

Travail de Bachelor 2024

Smart Virtual Characters using AI



Etudiant : Bastien Salamin

Professeur : Antoine Widmer

RÉSUMÉ

Ce travail de Bachelor porte sur le développement de personnages virtuels intelligents qui utilisent l'intelligence artificielle dans un jeu de simulation. La méthodologie employée comprend d'abord une définition des algorithmes usuels pour construire ce type de personnage, suivie d'une analyse des solutions existantes destinées à un tel développement via un état de l'art, et enfin d'une implémentation pratique d'un prototype. L'objectif principal de ce travail est l'élaboration d'un personnage capable de percevoir son environnement, d'interagir avec le joueur et d'exécuter des actions de manière autonome.

Le scénario dans lequel ce personnage évolue se base sur une simulation d'hôpital réalisée durant mon cursus en Informatique de Gestion, sous le moteur de jeu Unity. L'implémentation du prototype a nécessité au préalable une mise à jour de cette simulation ainsi que l'installation de Convai, solution sélectionnée à la suite de l'analyse de l'état de l'art. Ce document décrit le processus complet de la construction du personnage intelligent, depuis la plateforme en ligne de Convai jusqu'à son importation et sa configuration sur Unity.

Le résultat final est un avatar capable d'échanger de manière intelligente, de se mouvoir dans l'environnement de jeu et d'entreprendre des actions selon les ordres donnés par le joueur. De plus, le prototype démontre que l'intégration de modèles d'intelligence artificielle similaires à ChatGPT permet de simuler des interactions humaines crédibles. Ce travail illustre un équilibre délicat entre la rédaction et le développement du prototype, et met en lumière les problèmes rencontrés, les résultats obtenus et les aspects perfectibles de la solution sélectionnée.

Mots-clés : Intelligence artificielle, personnage non-jouable, réalité virtuelle, Unity.

AVANT-PROPOS

Dans le cadre de ma formation en Informatique de Gestion à la Haute École Supérieure de Suisse Occidentale (HES-SO), il m'a été demandé de réaliser un travail de Bachelor pour valider la fin de mes études. Ce travail inclut notamment la rédaction d'un rapport écrit sur un thème en lien avec mon cursus, un exercice qui représente une charge de travail conséquente en plus d'être un défi nouveau.

La sélection du thème du travail de Bachelor se fait par un processus où chaque étudiant doit indiquer ses cinq préférences, classées dans l'ordre, parmi une liste de thèmes fournie par la HES-SO. Le sujet « Smart Virtual Characters using AI », proposé par le professeur Antoine Widmer, m'a été attribué et correspondait à mon premier choix. La description du thème, qui consiste à explorer la création de personnages virtuels intelligents, a aussitôt suscité mon intérêt. En effet, cela m'offrait l'opportunité de participer d'une certaine manière à un développement d'un jeu vidéo, ce qui constitue un rêve d'enfance pour moi.

De plus, durant ma formation à la HES-SO, j'ai eu l'occasion de travailler sur un projet qui consistait au développement d'un prototype de jeu destiné à la réalité virtuelle. Ce projet, intitulé « Sim3D », m'a permis d'acquérir de premières bases dans l'utilisation du moteur de jeu Unity. Lorsque j'ai appris que mon sujet de Bachelor serait celui que j'avais placé en première position, j'étais heureux non seulement par la chance que j'avais eue dans le tirage au sort de mes propositions, mais aussi par cette nouvelle occasion de nourrir davantage mon rêve d'enfance.

REMERCIEMENTS

Je souhaite exprimer ma gratitude envers les personnes qui ont contribué, de quelque manière que ce soit, à la réalisation de ce document. Je remercie en premier lieu mon professeur responsable, Monsieur Antoine Widmer, et son collègue, Monsieur Jeff Zufferey, pour leurs conseils avisés et leur disponibilité tout au long de la durée de ce travail de Bachelor.

Je remercie également mes camarades de classe avec qui j'ai travaillé sur l'élaboration du projet Sim3D durant le semestre d'automne 2023, sans quoi le prototype réalisé au cours de ce travail n'aurait jamais pu voir le jour sous sa forme actuelle : Céline Vialard, Elias Borrajo et Simon de Blasi. Merci également à Monsieur Loïc Berthod, le client à qui ce projet était destiné, pour nous avoir donné cette opportunité.

Enfin, je tiens à exprimer ma gratitude envers toute ma famille pour leur soutien inconditionnel et émotionnel, ainsi que pour avoir accepté de relire ce document.

TABLE DES MATIÈRES

LISTE DES FIGURES	VII
LISTE DES ABRÉVIATIONS.....	X
INTRODUCTION	1
TERMINOLOGIE	1
UTILISATION DE L'INTELLIGENCE ARTIFICIELLE	2
1. OBJECTIFS & PROBLÉMATIQUES	3
1.1. PROGRAMMATION DES PNJS.....	3
1.1.1. <i>Finite-State Machines</i>	3
1.1.2. <i>Behavior Trees</i>	3
1.1.3. <i>Utility AI</i>	4
1.1.4. <i>Goal Oriented Action Planning</i>	4
1.1.5. <i>Chatbot</i>	4
1.1.6. <i>Considérations supplémentaires</i>	6
1.2. SCÉNARIO	6
1.2.1. <i>Unity</i>	6
1.2.2. <i>Sim3D</i>	8
1.2.3. <i>Ready Player Me</i>	10
1.3. OBJECTIFS.....	11
1.4. PROBLÉMATIQUES	11
2. ÉTAT DE L'ART	13
2.1. LISTE DES SOLUTIONS	13
2.1.1. <i>Convai</i>	13
2.1.2. <i>Inworld</i>	15
2.1.3. <i>Charisma.ai</i>	16
2.1.4. <i>Grail</i>	18
2.1.5. <i>Unity Machine Learning Agents</i>	19
2.1.6. <i>Total AI</i>	21
2.1.7. <i>IBRAIN</i>	21
2.2. MATRICE DE DÉCISIONS	22
2.2.1. <i>Suivi</i>	24
2.2.2. <i>Prise en main</i>	24
2.2.3. <i>Intelligence artificielle</i>	25
2.2.4. <i>Prix</i>	25

2.2.5.	Optionnel	25
2.2.6.	Analyse du résultat	26
2.2.7.	Remarques supplémentaires	27
3.	IMPLÉMENTATION DE LA SOLUTION	29
3.1.	PLATEFORME CONVAI	29
3.1.1.	Démarrage.....	29
3.1.2.	Description du personnage.....	30
3.1.3.	Langage et parole.....	31
3.1.4.	Banque de connaissances.....	32
3.1.5.	Style et personnalité.....	33
3.1.6.	État d'esprit.....	34
3.1.7.	Mémoire	34
3.1.8.	Modération.....	35
3.1.9.	Actions.....	36
3.1.10.	Design narratif.....	36
3.1.11.	Paramètres	39
3.1.12.	Configuration de l'avatar.....	39
3.1.13.	Conversation avec le personnage.....	40
3.1.14.	Compte	41
3.2.	PRÉPARATION DU PROJET SIM3D	42
3.2.1.	Mise à jour et importation des ressources.....	42
3.2.2.	Corrections des animations de la scène Sim3D	52
3.2.3.	XR Device Simulator.....	54
3.2.4.	Réimportation des références TextMesh Pro.....	55
3.3.	DÉVELOPPEMENT DU PNJ AVEC CONVAI.....	56
3.3.1.	Importation du personnage Convai.....	57
3.3.2.	Intégration de l'interface de Convai	59
3.3.3.	Activation de la synchronisation labiale	61
3.3.4.	Intégration du Design narratif.....	62
3.3.5.	Test des réponses	67
3.3.6.	Déplacement du personnage en jeu	68
3.3.7.	Assignment des actions	71
3.3.8.	Animation du personnage	75
3.3.9.	Test de la scène.....	77
3.4.	BILAN	79

3.5.	CONSTRUCTION DE LA SCÈNE	83
4.	GESTION DU PROJET	86
4.1.	PRÉPARATION.....	86
4.2.	SPRINT 0	86
4.2.1.	<i>Product Backlog</i>	87
4.2.2.	<i>Definition of Done</i>	88
4.3.	PAUSE INTER-SPRINT	88
4.4.	SPRINT 1	88
4.5.	SPRINT 2	90
4.6.	VÉLOCITÉ.....	91
4.7.	FINALISATION	91
	CONCLUSION	93
	DÉCLARATION DE L'AUTEUR	94
	BIBLIOGRAPHIE	95
	ANNEXE I : DONNÉE DU TRAVAIL DE BACHELOR	97
	ANNEXE II : PRODUCT BACKLOG	99
	ANNEXE III : JOURNAL DE BORD.....	100
	ANNEXE IV : DÉTAIL DU SPRINT 0.....	102
	ANNEXE V : DÉTAIL DU SPRINT 1.....	103
	ANNEXE VI : DÉTAIL DU SPRINT 2.....	104
	ANNEXE VII : TRANSCRIPTION D'UN DIALOGUE AVEC L'INFIRMIÈRE CONVAI	105

LISTE DES FIGURES

Figure 1 : Exemple de FSM pour un PNJ ayant le rôle d’une infirmière	3
Figure 2 : Exemple d’un Behavior Tree avec une séquence simple d’actions	4
Figure 3 : Deux écrans illustrant le chatbot de la Poste Suisse	5
Figure 4 : Exemple de structure d’une scène Unity	7
Figure 5 : Exemple de GameObject Unity avec divers composants	7
Figure 6 : Scène du jeu Sim3D vu par le joueur.....	9
Figure 7 : Rendu de l’avatar de l’infirmière.....	10
Figure 8 : Image promotionnelle de Convai	13
Figure 9 : Aperçu du plan gratuit proposé par Convai	14
Figure 10 : Image promotionnelle d’Inworld	15
Figure 11 : Capture d’écran d’un exemple de graphe narratif de Charisma.ai	17
Figure 12 : Logo de la solution Grail.....	18
Figure 13 : Diverses scènes de tests démontrant les possibilités d’usage des ML-Agents d’Unity .	20
Figure 14 : Logo de Total AI	21
Figure 15 : Illustration des comportements que les PNJs développés par IBRAIN peuvent prendre	22
Figure 16 : Matrice de décisions	23
Figure 17 : Tableau de bord de Convai	29
Figure 18 : Onglet « Character Description »	30
Figure 19 : Onglet « Language And Speech »	31
Figure 20 : Onglet « Knowledge Bank »	32
Figure 21 : Onglet « Personality & Style »	33
Figure 22 : Onglet « State Of Mind »	34
Figure 23 : Onglet « Memory »	35
Figure 24 : Onglet « Moderation & Guardrails »	35
Figure 25 : Onglet « Actions »	36
Figure 26 : Exemple d’une section du Design narratif	37
Figure 27 : Exemple d’un déclencheur du Design narratif	38
Figure 28 : Graphe narratif du personnage de l’infirmière	38
Figure 29 : Onglet « Settings »	39
Figure 30 : Onglet « Configure Avatar »	40

Figure 31 : Fenêtre de discussion du personnage sur Convai	41
Figure 32 : Informations du compte utilisateur	41
Figure 33 : Informations de facturation.....	42
Figure 34 : Modules installés de la version 2022.3.33 d'Unity sur Unity Hub	43
Figure 35 : Avertissement concernant le changement de version d'Unity	43
Figure 36 : Avertissement à l'ouverture du projet	44
Figure 37 : Avertissement concernant la mise à jour de scripts	44
Figure 38 : Aperçu de Convai sur l'Unity Asset Store.....	45
Figure 39 : Fenêtre Package Manager avec le filtre My Assets.....	45
Figure 40 : Liste des dépendances nécessitant une mise à jour pour Convai.....	46
Figure 41 : Liste des ressources de Convai	46
Figure 42 : Liste de conflits après l'installation de Convai sur Sim3D	47
Figure 43 : Avertissement en cas d'ouverture d'un projet Unity contenant des conflits	47
Figure 44 : Les quatre conflits après réouverture du projet	47
Figure 45 : Support officiel de glTFast par Unity	48
Figure 46 : Fenêtre Package Manager avec le filtre In Project	48
Figure 47 : Erreur liée au référencement de l'ancienne version de glTFast.....	49
Figure 48 : Ressources Ready Player Me à désinstaller	49
Figure 49 : Aperçu de Ready Player Me sur l'Unity Asset Store	49
Figure 50 : Ressource Ready Player Me visible depuis le Package Manager	50
Figure 51 : Liste des dépendances nécessitant une mise à jour pour Ready Player Me	50
Figure 52 : Liste des ressources de Ready Player Me.....	50
Figure 53 : Conflits de référencement après la mise à jour de Ready Player Me	51
Figure 54 : Exemple de Metadata changé en Grpc.Core.Metadata.....	51
Figure 55 : Exemple d'AudioConfig changé en Service.AudioConfig	52
Figure 56 : Fenêtre de bienvenue lors de la première utilisation de Convai sur Unity.....	52
Figure 57 : Emplacement de la scène de Sim3D	52
Figure 58 : Emplacement de l'élément Avatar du GameObject Patient avant la mise à jour de Ready Player Me	53
Figure 59 : Emplacement de l'élément Avatar du GameObject Patient après la mise à jour de Ready Player Me	53
Figure 60 : Ressource XR Interaction Toolkit visible depuis le Package Manager.....	54
Figure 61 : Ressource XR Device Simulator	54

Figure 62 : Emplacement du Prefab XR Device Simulator dans la scène de jeu et dans les fichiers du projet	55
Figure 63 : Copie de la scène de Sim3D pour le prototype du Bachelor	55
Figure 64 : Interfaces graphiques de Convai en jeu	56
Figure 65 : Emplacement du dossier Library dans le projet Sim3D	56
Figure 66 : Fenêtre demandant l'entrée de la clé API de Convai	57
Figure 67 : Fenêtre demandant l'entrée de l'ID du personnage Convai	57
Figure 68 : Liste de GameObjects de la scène après l'importation du personnage Convai	58
Figure 69 : L'infirmière suivant le joueur du regard	58
Figure 70 : Aperçu du contenu du Prefab Convai Essentials	59
Figure 71 : GameObject EventSystem	59
Figure 72 : Scripts ConvaiNPCManager et ConvaiGRPCAPI du GameObject Main Camera	60
Figure 73 : Configuration des contrôles du GameObject Convai Input Manager	60
Figure 74 : Conversation avec le personnage de l'infirmière	61
Figure 75 : Configuration de la synchronisation labiale du personnage	62
Figure 76 : Aperçu des modifications du script ConvaiOVR Lipsync	62
Figure 77 : Les sections du Design narratif récupérées sur Unity	62
Figure 78 : Les déclencheurs du Design narratif récupérés sur Unity	63
Figure 79 : Script HideObject	63
Figure 80 : GameObject AccueilTrigger	64
Figure 81 : GameObject Renderer_Head du joueur	65
Figure 82 : Méthode OnTriggerEnter du script NarrativeDesignTrigger	66
Figure 83 : L'infirmière initiant la conversation lorsque le joueur s'avance	66
Figure 84 : Composant Nav Mesh Agent du personnage	68
Figure 85 : GameObject HopitalChambre incorporant le script NavMeshSurface	69
Figure 86 : Exemple de GameObject incorporant le script NavMeshModifier	70
Figure 87 : Fenêtre Navigation d'Unity	70
Figure 88 : Aperçu de la surface de déplacement de l'infirmière dans la scène de jeu	71
Figure 89 : Composant Convai Actions Handler du personnage	71
Figure 90 : GameObject Convai Interactables	72
Figure 91 : Modification de la méthode ParseAction du script ConvaiInteractablesData	73
Figure 92 : Ajout d'actions supplémentaires à l'énumération ActionChoice	74
Figure 93 : Ajout à la méthode DoAction de cases correspondant aux actions supplémentaires ..	74

Figure 94 : Code de l'action personnalisée Poser.....	75
Figure 95 : Composant Animator du personnage.....	75
Figure 96 : Fenêtre Animator du personnage de l'infirmière	76
Figure 97 : Position des zones où l'infirmière devra s'y diriger pour réaliser ses actions	77
Figure 98 : Condition pour déterminer la destination du personnage selon l'objet à prendre	78
Figure 99 : Schéma des fonctionnalités disponibles sous Sim3D.....	81
Figure 100 : Schéma des fonctionnalités disponibles avec Convai	82
Figure 101 : Première partie de la hiérarchie de la scène HospitalRoomConvai	83
Figure 102 : Seconde partie de la hiérarchie de la scène HospitalRoomConvai	85
Figure 103 : Burn down chart du sprint 0	87
Figure 104 : Burn down chart du sprint 1	88
Figure 105 : User stories sélectionnées pour le sprint 1	89
Figure 106 : Burn down chart du sprint 2	90
Figure 107 : User stories sélectionnées pour le sprint 2	90
Figure 108 : Graphique de la vélocité du projet.....	91

LISTE DES ABRÉVIATIONS

Abréviation	Libellé
HES-SO	Haute École Supérieure de Suisse Occidentale
IA	Intelligence Artificielle
PNJ	Personnage Non-Jouable
NPC	Non-Player Character
LLM	Large Language Model
FSM	Finite-State Machine
GOAP	Goal Oriented Action Planning
ID	Identifiant
API	Application Programming Interface

INTRODUCTION

L'avènement de l'Intelligence Artificielle (IA) et de la réalité virtuelle a révolutionné le paysage technologique, offrant par la même occasion de nouveaux outils attractifs pour les développeurs d'applications de jeu. En effet, des possibilités innovantes quant à la création de personnages virtuels sont ainsi offertes, avec des cas d'utilisations variés. Que ce soit pour répondre à des besoins éducatifs, offrir une immersion accrue dans le cadre d'un jeu vidéo, ou simplement pour un support client interactif, les possibilités sont nombreuses et les intérêts grandissants.

« Smart Virtual Characters using AI » signifie littéralement en français « personnages virtuels intelligents utilisant l'Intelligence Artificielle ». Ce travail de Bachelor porte donc sur le développement de personnages numériques capables aussi bien d'interagir de manière autonome et en temps réel avec leur environnement, que de répondre de manière pertinente aux questions posées par le joueur et ainsi tenir une conversation humaine crédible.

Le but est d'obtenir un rapport détaillé sur les solutions disponibles pour la réalisation d'un tel projet, afin d'aider par la suite les équipes de recherche à choisir les meilleurs moyens pour construire et intégrer ces personnages dans une scène de jeu destinée à la réalité virtuelle et, à terme, pouvoir les réutiliser dans le cadre de nouveaux mandats. Tous les détails supplémentaires se trouvent dans l'annexe I « Données du Travail de Bachelor », en fin de document. Toujours selon cette annexe, ce travail doit regrouper dans une première partie les connaissances et solutions existantes sur ce sujet par le biais d'un état de l'art. Puis, dans une seconde partie, une implémentation pratique doit être réalisée à la suite de l'analyse des connaissances recueillies.

Ainsi, pour développer le thème de ce travail de Bachelor, ce rapport détaille d'abord les objectifs et les problématiques à résoudre lors de l'implémentation avec l'aide d'un scénario initial. Ensuite, un état de l'art présente différentes solutions permettant de faciliter le développement, comparées à l'aide d'une matrice de décisions. Enfin, la réalisation pratique détaille toutes les étapes que j'ai effectuées, avec l'utilisation de la solution choisie après analyse de la matrice, pour parvenir à un prototype fonctionnel. Pour terminer, des explications sur la gestion de ce travail sont fournies avant de procéder à la conclusion.

TERMINOLOGIE

Avant d'approfondir le sujet, il est important de définir dès maintenant un terme essentiel en lien avec le thème développé et qui réapparaîtra tout au long de ce rapport. Cela vise principalement à faciliter la lecture de ce dernier et à offrir une meilleure compréhension.

Comme ce travail consiste à développer des personnages d'un jeu qui ne sont pas contrôlés par le joueur, l'industrie du jeu vidéo appelle plus communément ces acteurs des « Personnages Non-

Jouables » (PNJ), l'équivalent anglais étant « Non-Player Character » (NPC). Leurs rôles peuvent varier d'un jeu à l'autre et aussi au sein d'un même jeu, qu'il s'agisse d'ennemis à affronter ou d'aides pour permettre de progresser dans l'aventure. L'un des buts principaux de ces personnages est d'apporter un semblant de vie et d'activité, voire d'immersion, évitant ainsi au joueur de se sentir seul dans sa partie (Rabin, 2017).

En parallèle, il est également utile de comprendre comment l'IA utilisée pour les PNJs des jeux vidéo diffère des autres formes d'IA telles que les « Large Language Models » (LLM) comme ChatGPT, ces derniers s'étant popularisés dans le monde entier. En effet, bien que les deux possèdent le nom d'IA, elles se distinguent par leur conception et leur fonction. Dans les jeux vidéo, l'IA des PNJs est conçue pour donner à ces derniers un aspect plus ou moins réaliste, en suivant des comportements prédéfinis par les développeurs et spécifiques au contexte du jeu. Cependant, la programmation de ces personnages fait qu'ils ne peuvent sortir au-delà de la routine que les programmeurs leur ont attribuée.

En revanche, une IA comme ChatGPT traite le langage naturel et permet d'avoir des conversations sur un large éventail de sujet, produisant ainsi une variété de réponses plus ou moins pertinentes. Cela utilise des techniques plus avancées pour comprendre et générer ensuite du texte de manière fluide. Ainsi, l'IA d'un jeu vidéo est spécialisée et contextuelle mais limitée, tandis qu'une IA comme ChatGPT est générale et conversationnelle, avec un domaine de connaissance quasi-infini.

UTILISATION DE L'INTELLIGENCE ARTIFICIELLE

Lors de la rédaction de ce rapport, j'ai eu recours à l'IA, plus précisément aux modèles GPT-4o et GPT-3.5 de ChatGPT, fournis par OpenAI, pour m'aider à la rédaction de certains paragraphes. J'ai utilisé ces outils notamment pour vérifier l'orthographe et la grammaire, ainsi que pour reformuler certains termes et améliorer la fluidité de mon texte. Parfois, lorsque des mots me manquaient, l'IA m'a aidé à les trouver. De plus, pour décrire l'implémentation du prototype, j'ai principalement rédigé des points clés que j'ai ensuite demandé à ChatGPT de développer en texte cohérent et structuré, afin de m'en inspirer pour la rédaction. Bien sûr, j'ai relu, corrigé et réécrit chaque réponse que l'IA m'a fournie, afin d'être sûr que le texte transmette effectivement l'idée que j'avais en tête. Tout ceci dans le but d'obtenir des paragraphes lisibles, rédigés dans un français correct et professionnel.

1. OBJECTIFS & PROBLÉMATIQUES

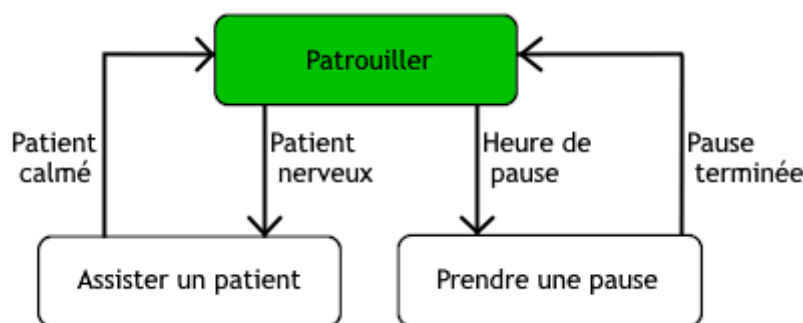
1.1. PROGRAMMATION DES PNJS

D'après Dawe, Gargolinski, Dicken, Humphreys et Mark (2013), il y a plusieurs algorithmes qu'il est possible d'implémenter pour gérer les comportements des PNJs. Celles qu'ils ont décrites et qui ont été les plus ressorties dans mes recherches sur la manière de programmer cette intelligence sont les suivantes, du plus simple aux plus complexes : les « Finite-State Machines » (FSM), les « Behavior Trees », « Utility AI » et « Goal Oriented Action Planning » (GOAP).

1.1.1. FINITE-STATE MACHINES

Un FSM consiste en une définition de plusieurs états, avec une transition pour passer d'un état à un autre, suivant le comportement que l'on veut donner à notre personnage face à certaines situations. Un état correspond à un comportement spécifique, et les transitions s'effectuent si certaines conditions sont rencontrées. La figure 1 illustre une représentation simplifiée d'un FSM, avec les états sous forme de rectangle, et les transitions avec leur condition sous forme de flèche. L'état initial, « Patrouiller », est représenté en vert, et est l'état dans lequel notre personnage va démarrer au lancement du jeu. Bien que facile à mettre en place, un FSM peut vite devenir compliqué à corriger si l'on commence à avoir plusieurs états à gérer.

Figure 1 : Exemple de FSM pour un PNJ ayant le rôle d'une infirmière

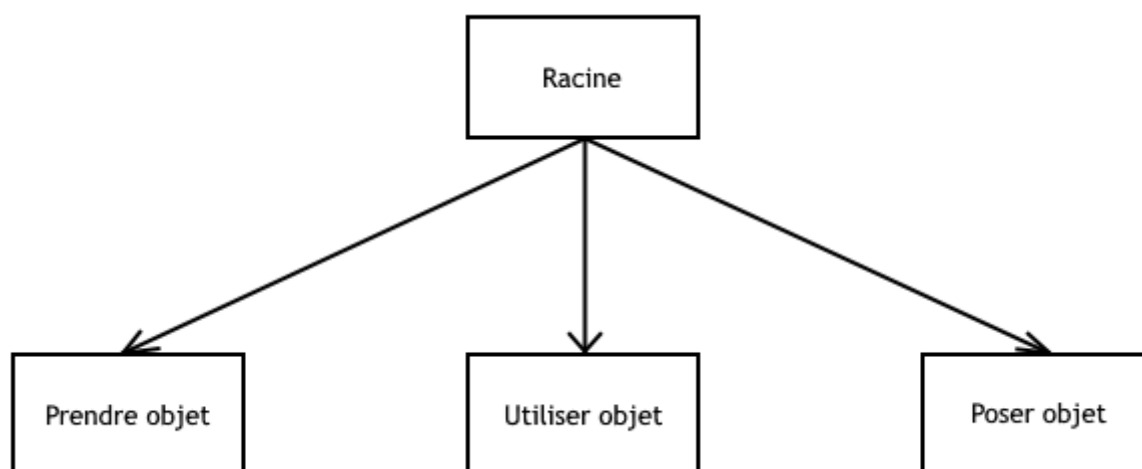


Source : données de l'auteur

1.1.2. BEHAVIOR TREES

Un Behavior Tree se compose d'un nœud racine suivi de plusieurs nœuds enfants, chacun de ces derniers représentant une action que le PNJ peut effectuer. Les nœuds enfants peuvent eux-mêmes avoir des sous-nœuds, créant ainsi cette structure en forme d'arbre. Chaque nœud est évalué séquentiellement, en partant toujours du nœud racine et en progressant à travers les branches de haut en bas et de gauche à droite. Si la condition est remplie, l'action du nœud actuellement évalué est exécutée, avant de recommencer le parcours de l'arbre pour évaluer les suivants. Il n'y a plus cet aspect de transitions, ce qui permet une construction plus flexible des comportements du personnage. Cependant, cette évaluation séquentielle peut causer des temps d'exécution plus longs, notamment avec de grands arbres. La figure 2 montre un exemple simplifié de Behavior Tree.

Figure 2 : Exemple d'un Behavior Tree avec une séquence simple d'actions



Source : données de l'auteur

1.1.3. UTILITY AI

Utility AI est un système qui permet au PNJ de déterminer la meilleure action à réaliser parmi celles disponibles face à une situation donnée. Pour effectuer ce choix, chaque action est associée à un score, en général un nombre décimal compris entre 0 et 1. Celle ayant le score le plus élevé est ainsi sélectionnée. De plus, ce score peut varier en fonction de la situation dans laquelle se trouve le PNJ, certaines actions étant plus pertinentes à exécuter que d'autres. Cette approche rend alors le comportement des personnages plus dynamique et flexible. Cependant, ce type de système nécessite un paramétrage minutieux et une grande variété d'actions possibles en jeu pour être efficace.

1.1.4. GOAL ORIENTED ACTION PLANNING

GOAP est un planificateur. La situation actuelle du jeu est fournie au PNJ, de même que la situation souhaitée, devenant ainsi l'objectif que le personnage doit atteindre. Pour y parvenir, une liste de toutes les actions possibles lui est fournie, avec leurs conséquences sur le jeu, et les conditions nécessaires pour pouvoir les exécuter. Le personnage va alors, de manière autonome, choisir et enchaîner logiquement les actions qui vont lui permettre d'atteindre son objectif. Avec le GOAP, les PNJs donnent ainsi l'impression de réfléchir par eux-mêmes, puisque ce sont eux qui créent leurs plans pour parvenir à leurs buts.

1.1.5. CHATBOT

Même s'il ne fait pas partie des algorithmes destinés à définir les comportements d'un PNJ, il est intéressant de définir ce qu'est un « chatbot ». Agglomération des mots « chatter » et « robot », un chatbot est un programme informatique permettant de simuler une conversation pouvant être basée sur du texte ou de la voix. Contrairement aux quatre algorithmes décrits précédemment, un chatbot

ne vise pas à intégrer des comportements à un personnage virtuel afin qu'il réalise des actions, mais plutôt pour qu'il donne des réponses de manière appropriée face aux entrées de l'utilisateur.

Les chatbots peuvent être programmés manuellement afin qu'ils puissent suivre un arbre de décision, donnant ainsi des réponses préconstruites en fonction de mots-clés spécifiques. Cependant, cette approche limite les sujets dont le chatbot peut traiter, un problème lorsqu'un utilisateur pose des questions trop ciblées ou hors-sujet. Une autre approche consiste à alimenter le chatbot à l'aide d'une IA, afin de lui permettre de répondre à un plus large éventail de questions, ChatGPT étant un exemple récent et avancé d'une telle technologie.

Figure 3 : Deux écrans illustrant le chatbot de la Poste Suisse



Source : <https://www.post.ch/fr/aide-et-contact>

Les applications concrètes de chatbots se retrouvent plutôt dans les supports clients sur les sites web d'entreprises, comme celui de la Poste Suisse illustré en figure 3. Siri, l'assistant virtuel des appareils d'Apple, est un autre exemple de chatbot utilisant la reconnaissance vocale. Bien que cette technologie ne soit pas couramment utilisée dans les jeux vidéo, elle reste intéressante pour donner aux PNJs la capacité d'échanger avec le joueur, en particulier lorsqu'un LLM est utilisé pour permettre des conversations plus élaborées.

1.1.6. CONSIDÉRATIONS SUPPLÉMENTAIRES

Chacun des algorithmes présentés est valide pour donner un comportement au PNJ, certains faisant plus de sens que d'autres suivant le genre du jeu. Enfin, de récentes études ont cherchées à démontrer qu'en plus de donner un moyen de communication, il est possible d'utiliser les LLMs tels que ChatGPT et parvenir, en usant de leur capacité conversationnelle et avec une première instruction préparée, à simuler des comportements humains pour ces personnages virtuels sans avoir à le coder entièrement, comme l'ont prouvé Park, O'Brien, Cai, Morris, Liang et Bernstein (2023) avec leur jeu de simulation d'un village de 25 habitants.

Cependant, il leur a quand même fallu construire un système de mémoire pour que leurs personnages puissent agir de manière crédible. Et comme toute expérience, certains problèmes ont fait surface, comme un langage jugé trop formel lors d'une discussion familiale entre deux PNJs, ou des oublis concernant l'heure de fermeture des magasins faisant qu'un personnage s'y rendait malgré tout, entre autres.

1.2. SCÉNARIO

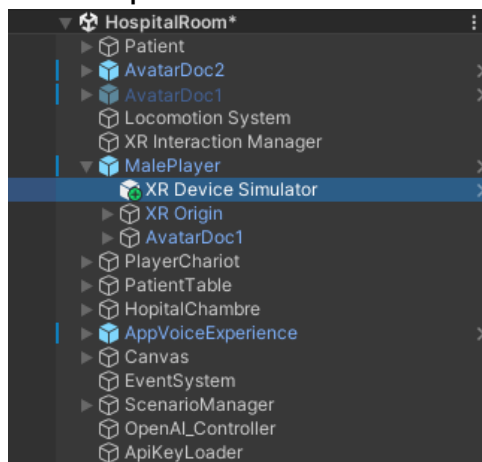
Après ce premier aperçu des types d'IA disponibles, il est temps de définir les objectifs et problématiques de ce travail de Bachelor. Comme mentionné en introduction, le but est d'obtenir un prototype de PNJ intelligent, capable de prendre conscience de son environnement et du joueur afin de pouvoir échanger avec ce dernier et de réaliser certaines actions de manière autonome et en temps réel.

Pour s'assurer que les objectifs, et surtout les problématiques, soient ciblés et utiles aux équipes de recherche, j'ai estimé important de définir un scénario de départ afin d'avoir un contexte plutôt précis et ainsi commencer avec un premier rôle et personnalité pour ce personnage. Après discussion avec mon professeur responsable, M. Antoine Widmer, le prototype se base sur un projet élaboré avec le moteur de jeu Unity et sur lequel j'ai déjà précédemment travaillé dans le cadre d'un autre module de cours au sein de ma formation à la HES-SO : « Sim3D ».

1.2.1. UNITY

Avant de rentrer dans l'explication de ce qu'est Sim3D, il est important de définir ce qu'est Unity. Selon Juliani et al. (2020, pp. 8-9), Unity est une plateforme de développement avec une interface graphique, un moteur de rendu et un moteur physique, permettant ainsi de réaliser des scènes animées en 3D. Son utilisation est principalement adoptée par l'industrie du jeu vidéo. Il est possible de créer aussi bien des jeux destinés aux téléphones portables qu'à la réalité virtuelle, Unity n'étant pas limité à un genre de jeu spécifique. Le principal langage de programmation à connaître pour travailler avec cette plateforme est le C#.

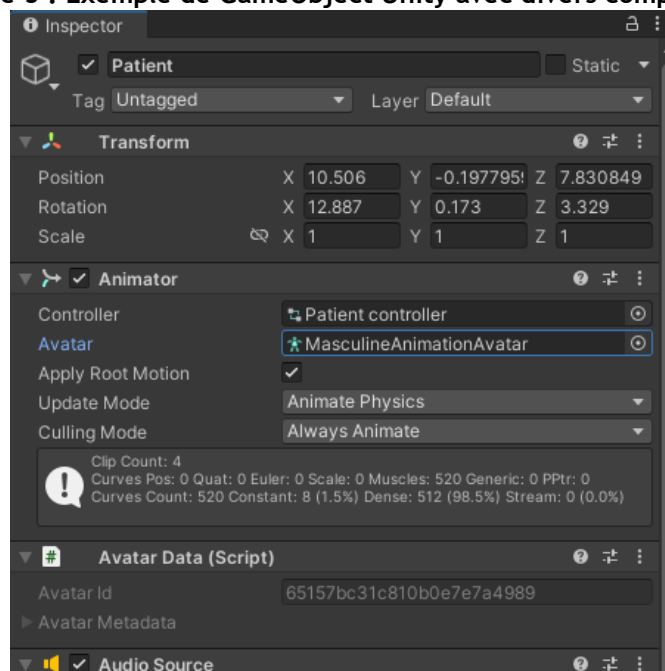
Figure 4 : Exemple de structure d'une scène Unity



Source : données de l'auteur

Un projet Unity se compose de plusieurs ressources nommées « Assets », la scène de jeu en étant un exemple et constituant aussi l'élément principal à la construction de l'environnement. Chaque élément à l'intérieur d'une scène est appelé un « GameObject », et est représenté par un cube dans l'interface graphique, comme illustré dans la figure 4. Sur cette image, la scène « HospitalRoom » sert d'élément racine, et ses GameObjects sont ordonnées à la suite, eux-mêmes pouvant en contenir d'autres, donnant ainsi une structure en forme d'arbre. Ces GameObjects peuvent être des modèles 3D, des éléments pour éclairer la scène, des caméras montrant l'écran du joueur, ou bien des scripts qui implémentent certaines fonctionnalités. Chaque GameObject peut avoir plusieurs composants, le premier étant à chaque fois « Transform », soit la position initiale de l'élément dans la scène. La figure 5 est un exemple de GameObject.

Figure 5 : Exemple de GameObject Unity avec divers composants



Source : données de l'auteur

1.2.2. Sim3D

Dans le cadre du module de cours 646-2 intitulé « Projet Agile », l'objectif était de mettre en pratique la méthode de travail en équipe connue sous le nom de « Scrum ». Pour ce faire, il a fallu choisir un projet parmi ceux proposés, tous étant des problématiques concrètes à résoudre et dont les clients n'étaient pas nécessairement des professeurs de la filière d'Informatique de Gestion. Après avoir examiné les descriptions de chaque projet, mon groupe a sélectionné celui nommé Sim3D, le seul qui se distinguait des autres vu qu'il suggérait un travail sur un jeu vidéo.

Proposé par la Haute Ecole de Santé de la HES-SO, Sim3D vise à fournir une simulation de soins infirmiers avec l'utilisation de la réalité virtuelle. Une base de projet sur Unity a été mise à disposition, consistant en une scène de chambre d'hôpital équipée de toutes les fournitures nécessaires. À partir de là, il a fallu essayer d'intégrer les fonctionnalités souhaitées par le client. Les principales exigences étaient les suivantes :

- Permettre au joueur de se déplacer dans la chambre, de prendre les outils et les médicaments et de pouvoir les utiliser sur le patient ;
- Intégrer des avatars pour représenter le joueur et le patient, avec des animations pour rendre la simulation plus réaliste ;
- Gérer le multijoueur ;
- Communiquer avec le patient en intégrant des réponses intelligentes via ChatGPT ;
- Développer un scénario par une liste d'objectifs à accomplir pour permettre une progression dans le jeu ;

Pour effectuer ce travail, il a fallu apprendre à utiliser Unity et les casques de réalité virtuelle. En effet, comme ces technologies ne possédaient aucun module de cours dédié dans mon cursus, une autre partie importante de ce projet était de se familiariser avec ces nouveaux éléments, même si la programmation en langage C# avait déjà fait l'objet de cours consacrés à son apprentissage.

Durant le semestre, les avancées se faisaient pas à pas au rythme des « sprints », des cycles courts de développement, jusqu'à aboutir à un prototype fonctionnel. La figure 6 montre le point de vue de l'utilisateur au lancement d'une partie. Les avatars du joueur et du patient ont été intégrés, avec des animations de base pour ce dernier, ainsi que la synchronisation des mouvements entre les manettes du joueur et les mains de son avatar, la possibilité de se déplacer et de saisir des objets, la faculté d'utiliser les outils du médecin sur certaines parties du corps du patient, ainsi qu'un affichage textuel de l'objectif actuel et ceux déjà accomplis. L'aspect multijoueur nécessitait plus de temps que celui à disposition et, avec l'accord du client, il n'a pas été implémenté.

Figure 6 : Scène du jeu Sim3D vu par le joueur



Source : données de l'auteur

Concernant la communication entre le joueur et le patient, cela a été construit ainsi : pour chaque objectif du jeu, le joueur doit échanger avec ce personnage. Il a fallu créer deux instances de ChatGPT pour gérer ces aspects. Lorsque le joueur active le micro à l'aide des manettes fournies avec le casque de réalité virtuelle et parle, ce qu'il dit se transforme en texte par le biais d'une reconnaissance vocale, qui est ensuite transmis à la première instance. Cela a pour but de vérifier que ce qui a été prononcé par le joueur correspond à une forme de demande attendue pour réaliser l'objectif en cours. La seconde instance représente le patient et, en lui ayant décrit au préalable son rôle dans le jeu, répond au joueur de manière intelligente, en convertissant le texte généré par ChatGPT en audio par le biais d'une synthèse vocale.

Par exemple, l'objectif initial du jeu consiste à se présenter au patient en tant que médecin avec nom et prénom. La première instance vérifie si ce qu'a dit le joueur correspond à une forme de présentation, tandis que la seconde émet la réponse du patient en réaction à ce qu'a dit le joueur, sans se soucier de la première instance. La progression du scénario est déterminée par la première instance, qui évalue chaque phrase prononcée et décide si cela correspond à l'objectif attendu ou non.

Le scénario du jeu se déroule ainsi : d'abord le joueur se présente. Ensuite, il demande au patient comment est-ce qu'il se sent. Puis, il peut ramasser les différents outils, qui sont au nombre de quatre : le thermomètre, le tensiomètre, le stéthoscope et le pulsomètre, pour vérifier l'état du patient. Chaque appareil nécessite d'être appliqué sur des parties du corps différentes. Mais pour que ces outils puissent afficher leurs mesures, le joueur doit dire quel appareil il s'apprête à utiliser

et demander l'autorisation au patient de faire le relevé. Après avoir effectué les quatre mesures, le joueur doit poser une boîte de médicament sur la table du patient, et informer ce dernier du traitement préconisé avant de conclure la conversation.

Ayant été responsable de la gestion du déplacement dans la scène de jeu entre autres aspects, j'ai choisi que le joueur serait représenté par un homme médecin. Cependant, bien que ce n'était pas une priorité, j'avais déjà envisagé l'apparence pour un avatar féminin, au cas où mon groupe aurait trouvé le temps d'intégrer une option de choix du sexe pour le joueur avant de démarrer la partie. Mais comme cette fonctionnalité était facultative pour le client, j'ai décidé de réutiliser cet avatar féminin en le plaçant dans la scène de jeu en tant qu'infirmière, même si son rôle allait rester décoratif, comme il est possible de le constater dans la figure 6.

Cependant, comme ce travail de Bachelor a pour but de créer un modèle de PNJ intelligent, que j'ai exprimé le besoin d'avoir un certain contexte afin de construire le prototype attendu, et pour éviter de partir entièrement de zéro avec une scène vide, il a été convenu de reprendre le projet Sim3D. Plus précisément, l'infirmière, représentée en figure 7 et qui servait de simple figurante, sera le cobaye de mon travail. Son rôle évoluera pour assister le joueur dans ses tâches médicales, devenant ainsi un personnage proactif.

Figure 7 : Rendu de l'avatar de l'infirmière



Source : <https://readyplayer.me/fr/hub/avatars>

1.2.3. READY PLAYER ME

À titre d'information complémentaire, les avatars utilisés dans le projet Sim3D ont été créés grâce à la plateforme « Ready Player Me ». Cet outil permet aux développeurs de disposer de personnages personnalisables et prêts à animer, avec une facilité d'intégration dans divers projets, que ce soit sur Unity ou autre. En plus de pouvoir utiliser ces modèles, il est possible d'intégrer aussi l'interface de création du personnage directement dans le jeu. En outre, les utilisateurs ayant un compte joueur sur Ready Player Me peuvent réutiliser les avatars qu'ils ont créés dans toutes les

applications qui intègrent cet outil. C'est une excellente ressource pour obtenir rapidement des modèles 3D prêts à l'emploi, surtout lorsqu'il n'est pas possible de les modéliser soi-même, comme ce fut le cas avec Sim3D.

1.3. OBJECTIFS

Maintenant que le scénario est défini avec une base de personnage sur lequel travailler, i.e. l'infirmière qui fait office de figurante dans le projet Sim3D, il est possible de regrouper pour une meilleure vue d'ensemble les objectifs à atteindre par ce travail. Voici une liste d'éléments que j'ai estimé importants à accomplir lors du développement du prototype :

- Créer un PNJ dans une scène de jeu destinée pour de la réalité virtuelle ;
- Concevoir un PNJ capable de reconnaître son environnement, y compris les obstacles, et d'identifier les éléments avec lesquels il peut interagir ou non en fonction de son rôle et de la scène ;
- Développer un PNJ autonome qui s'adapte en temps réel aux actions du joueur, avec une personnalité qu'il est possible de changer pour diversifier l'expérience de jeu ;
- Implémenter un PNJ qui puisse écouter le joueur, suivre ses instructions et donner des réponses pertinentes, tout en respectant son rôle et le contexte du jeu ;
- Fournir une explication précise de chaque étape lors de la construction du prototype, après avoir choisi la solution qui aura été jugée la plus pertinente à utiliser dans le cadre de ce travail de Bachelor ;

1.4. PROBLÉMATIQUES

Cette section aborde les différents aspects à prendre en compte lors de l'implémentation du PNJ. Ils peuvent être techniques, mais ils incluent également des réflexions sur la poursuite du développement du personnage, pour aller au-delà de la portée définie dans ce travail de Bachelor.

Certaines problématiques sont écrites par rapport au scénario de Sim3D et au rôle de l'infirmière, tous deux servant de points de départ pour réaliser le prototype. Cependant, il est possible de penser dès maintenant aux autres personnalités et rôles que ces PNJs peuvent adopter. De plus, quelques-uns de ces points étaient déjà sujets à réflexion lors de la réalisation du projet Sim3D. Dès lors, voici la liste des problématiques que j'ai identifiées :

- Définir les objets pouvant être saisis et gérer la compréhension de l'objectif en cours pour permettre au personnage d'agir en conséquence. Le PNJ ne devra pas prendre des objets utilisés par le joueur, ni lui donner un objet inutile ;
- Maintenir la mémoire du PNJ pour que le personnage ne sorte pas de son rôle tout le long de la partie ;

- Le dialogue du PNJ doit s'adapter en fonction des actions réalisées par le joueur durant la partie, afin de garder une interaction réaliste ;
 - En plus de cela, la mémoire du personnage devra être réinitialisée lors d'une nouvelle partie. Il ne doit pas se rappeler des actions effectuées par le joueur lors d'une ancienne partie ;
- Générer des réponses vocales en français, appropriées et cohérentes en fonction des propos du joueur, du contexte et du rôle du personnage ;
- Si le PNJ agit en tant que soutien au joueur, il doit éviter de le déranger constamment et ne pas interférer de manière agressive au déroulement du scénario ;
 - Cependant, il est possible d'imaginer un comportement différent en fonction de la personnalité que l'on veut donner au PNJ, en plus de varier également les actions possibles ;
 - Des changements sont à prévoir dans les cas où plusieurs PNJs sont présents, en particulier leurs rôles s'ils ne peuvent pas être similaires. Cela pourrait se faire dans le cadre d'un scénario différent de Sim3D ;
- Obtenir une fluidité des animations entre chaque action effectuée pour maintenir un certain réalisme ;
 - Il est aussi utile de penser à la synchronisation labiale lorsque le PNJ communique oralement, pour que ses lèvres s'animent en fonction des mots prononcés ;
- Reconnaître de manière fiable et efficace les commandes vocales du joueur pour réaliser l'action attendue. L'interprétation des instructions doit se faire correctement afin que le PNJ ne fasse pas une autre action que celle qui lui a été demandée ;
- Prendre en compte le coût potentiel associé à la solution sélectionnée. Comme il sera démontré au chapitre suivant, certaines d'entre elles limitent leurs fonctionnalités sans un apport financier préalable ;

Ces points englobent une variété de défis techniques et conceptuels indispensables pour créer un personnage réaliste et intelligent. Ils soulignent les éléments essentiels à prendre en compte, allant de la gestion de la mémoire et des interactions du PNJ, à la reconnaissance et génération vocale. Il est important de considérer ces aspects non seulement lors de la création du prototype, mais également pour anticiper les besoins ultérieurs du projet. En suivant ces points, le personnage de l'infirmière sera non seulement pratique, mais aussi immersif et adaptable à diverses situations, tout en préservant la faisabilité du projet pour de futures améliorations.

2. ÉTAT DE L'ART

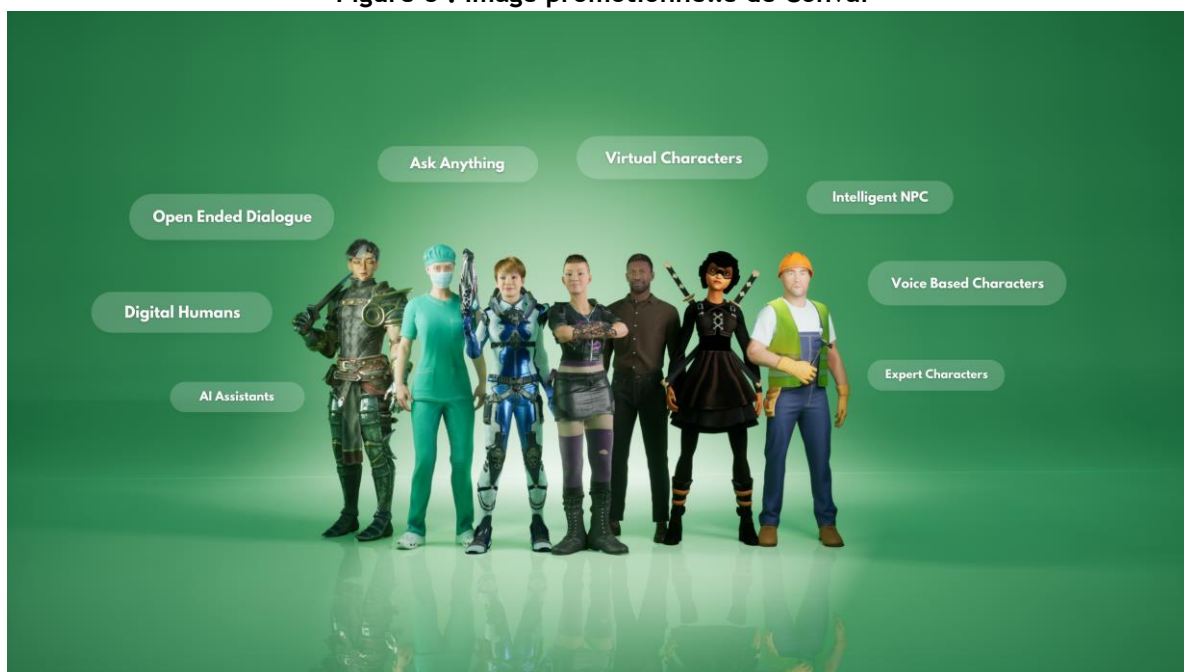
2.1. LISTE DES SOLUTIONS

Bien qu'il soit possible de coder entièrement à la main un prototype qui réponde aux attentes voulues de ce sujet de Bachelor, il existe fort heureusement différentes solutions permettant de gagner considérablement du temps. Comme mentionné dans le chapitre d'introduction, avec la démocratisation de ChatGPT, diverses entreprises ont saisi cette opportunité pour créer et mettre à disposition des développeurs différentes ressources, gratuites ou payantes, ayant recours à l'intelligence artificielle.

Ce sous-chapitre met en lumière ces solutions, trouvées au cours de mes recherches en ligne. Pour chacune d'entre elles, une description est fournie, avec les fonctionnalités proposées, la date de la dernière mise à jour, si de la documentation ou des tutoriels sont disponibles, les tarifs éventuels ainsi que d'autres éléments qui peuvent être importants à prendre en compte. Toutes ces solutions sont garanties par leurs créateurs de fonctionner sur le moteur de jeu Unity, ce dernier étant le principal critère de sélection.

2.1.1. CONVAI

Figure 8 : Image promotionnelle de Convai



Source : <https://convai.com/blog/introducing-convai>

Convai est une solution très récente qui offre la possibilité de créer des PNJs intelligents ayant une histoire ou des origines personnalisées, et de pouvoir ensuite communiquer avec eux, comme le montre l'image promotionnelle en figure 8. Utilisant la technologie des LLMs, les personnages créés peuvent ainsi répondre au joueur avec une synchronisation labiale et même exécuter des actions sur

commande. La première mise en ligne de cette solution sur l'« Unity Asset Store », le magasin de ressources conçu pour Unity, a eu lieu l'an passé en 2023. Quant à la dernière mise à jour, elle date du 21 juin 2024 au moment de l'écriture de ce document. De plus, les revues des utilisateurs sur l'Unity Asset Store sont régulières et semblent pour la plupart positives.

La création du personnage se fait directement sur leur site officiel (<https://convai.com/>) après s'y être connecté, avant de pouvoir l'importer dans Unity. Convai intègre les avatars Ready Player Me, mais il est tout à fait possible d'utiliser ses propres modèles, à condition qu'ils soient correctement configurés. De plus, une documentation explique comment paramétrer le PNJ, que ce soit à sa création sur le site ou après son importation sur la plateforme de développement. Des exemples, des tutoriels vidéo et même des scènes de tests sont à disposition, ainsi qu'un support pour résoudre les problèmes éventuels, avec en plus une communauté accessible via l'application de messagerie nommée « Discord ». Selon les revues des utilisateurs mentionnées précédemment, le support semble répondre assez rapidement. Enfin, la page du magasin précise une comptabilité uniquement à partir de la version 2022.3.0 d'Unity.

Figure 9 : Aperçu du plan gratuit proposé par Convai

Plans	
Free	
For starting out and learning to use Convai APIs	
\$0 / month	
Get Started >	
Daily free interaction quota ⓘ	100
Monthly interaction quota ⓘ	—
Interaction costs beyond quota ⓘ	—
Character concurrency ⓘ	1
Knowledge size ⓘ	1MB
Alpha feature access ⓘ	—
Moderation filter controls/adjustment ⓘ	—

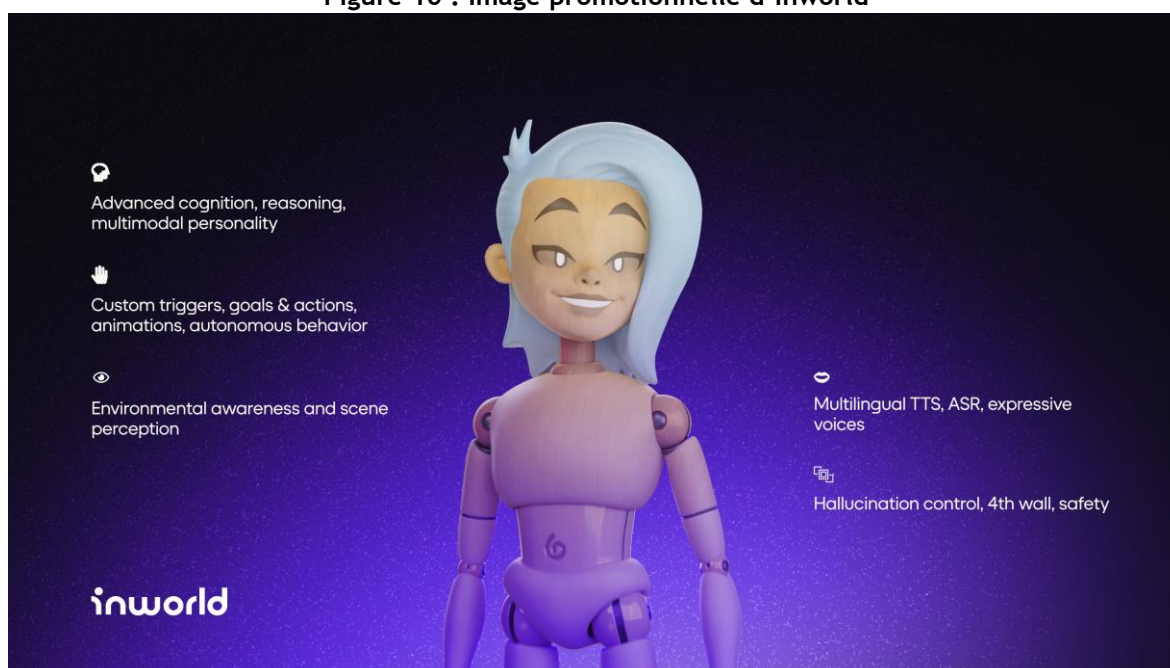
Source : <https://convai.com/pricing>

Néanmoins, une page de tarification présente sur le site de Convai indique qu'une utilisation gratuite de leur solution est limitée. En effet, selon la figure 9, il y a un quota journalier d'interactions qu'il est impossible de dépasser, une capacité limitée pour la banque de connaissances du PNJ, et une restriction de concurrence. C'est-à-dire qu'une seule personne à la

fois peut interagir avec l'instance de Convai. Par exemple, si le prototype est développé avec cette solution et que deux joueurs l'utilisent simultanément sur leur propre casque et parlent à l'infirmière, seul un des deux utilisateurs recevra une réponse. Malgré tout, Convai semble être une option appropriée vu les fonctionnalités proposées, notamment pour les actions que les PNJs peuvent accomplir.

2.1.2. INWORLD

Figure 10 : Image promotionnelle d'Inworld



Source : <https://assetstore.unity.com/packages/tools/behavior-ai/ai-npc-engine-v3-dialogue-behavior-for-unity-inworld-229406>

Inworld est également une solution récente, créée avant Convai, qui a déjà établi plusieurs partenariats avec des sociétés et présente des cas d'utilisation dans certains jeux. Inworld permet la création de personnages intelligents avec lesquels on peut communiquer, en étant configurés au préalable à l'aide de plusieurs paramètres, comme le montre la figure 10, afin d'obtenir des personnages aussi réalistes que possible. Inworld emploie la technologie des LLMs ainsi que ses propres modèles propriétaires pour construire ses PNJs. La première version de cette solution a été mise en ligne sur l'Unity Asset Store en 2022 et elle est régulièrement mise à jour, la dernière en date étant le 10 juin 2024 à l'écriture de ce travail de Bachelor.

À la manière de Convai, il faut créer un compte sur la partie studio d'Inworld (<https://studio.inworld.ai/showcase>), et c'est principalement sur cette plateforme que la création et le paramétrage du personnage s'effectuent, avant de pouvoir l'importer sur Unity. Une documentation détaillée est fournie, accompagnée d'images et de vidéos explicatives, ce qui permet une personnalisation poussée du personnage. Les avatars Ready Player Me sont intégrés,

mais ils bénéficient d'un support réduit au profit des avatars natifs d'Inworld. En revanche, il est possible d'utiliser nos propres modèles de personnages ou même d'intégrer des voix personnalisées, avec des indications claires dans la documentation pour y parvenir. Inworld prétend que sa solution peut s'utiliser sans avoir recours à du code, et il est vrai que la plupart des paramètres sur leur plateforme demande un texte descriptif, un exemple étant la motivation du personnage. Mais cela n'empêche pas la documentation de fournir aussi les détails nécessaires à l'utilisation de leur solution sur la plateforme de développement employée, avec des tutoriels et même des scènes de tests pour mieux comprendre les fonctionnalités disponibles.

La version 2022.3.10 d'Unity est recommandée, mais la solution est rétro-compatible tant que l'environnement de développement est sur Windows. Une page de licences indique que même si toutes les fonctionnalités d'Inworld sont offertes, un quota d'interactions mensuel et un plafond de revenus générés inférieurs à 50'000 dollars ne doivent pas être dépassés pour continuer à utiliser leur produit gratuitement. Enfin, bien que les avis des utilisateurs sur l'Unity Asset Store soient moins nombreux, ils semblent montrer des retours plutôt positifs. De plus, Inworld dispose aussi d'une communauté accessible via l'application Discord.

Cependant, même si la quantité de paramètres disponibles sur la plateforme studio d'Inworld sont nombreuses pour personnaliser en détail le personnage, cela peut très vite submerger les utilisateurs s'essayant à la solution pour la première fois. De plus, seul l'anglais est supporté, que ce soit pour la voix du PNJ ou pour ses paramétrages, qui nécessitent alors uniquement du texte rédigé dans cette langue. Comme il est possible de le constater dans leur documentation, il est apparent que les personnages ne peuvent pas réaliser d'actions une fois en jeu, à part réagir à l'environnement par leurs paroles et leurs gestes, contrairement aux PNJs d'Inworld qui peuvent accomplir des tâches lorsqu'on leur en donne l'ordre. Ainsi, Inworld apparaît comme une solution idéale pour créer un personnage intelligent avec qui échanger de manière réaliste et approfondie, tant que la discussion se déroule en anglais, mais elle ne semble pas vraiment apte à aller au-delà de cette fonctionnalité.

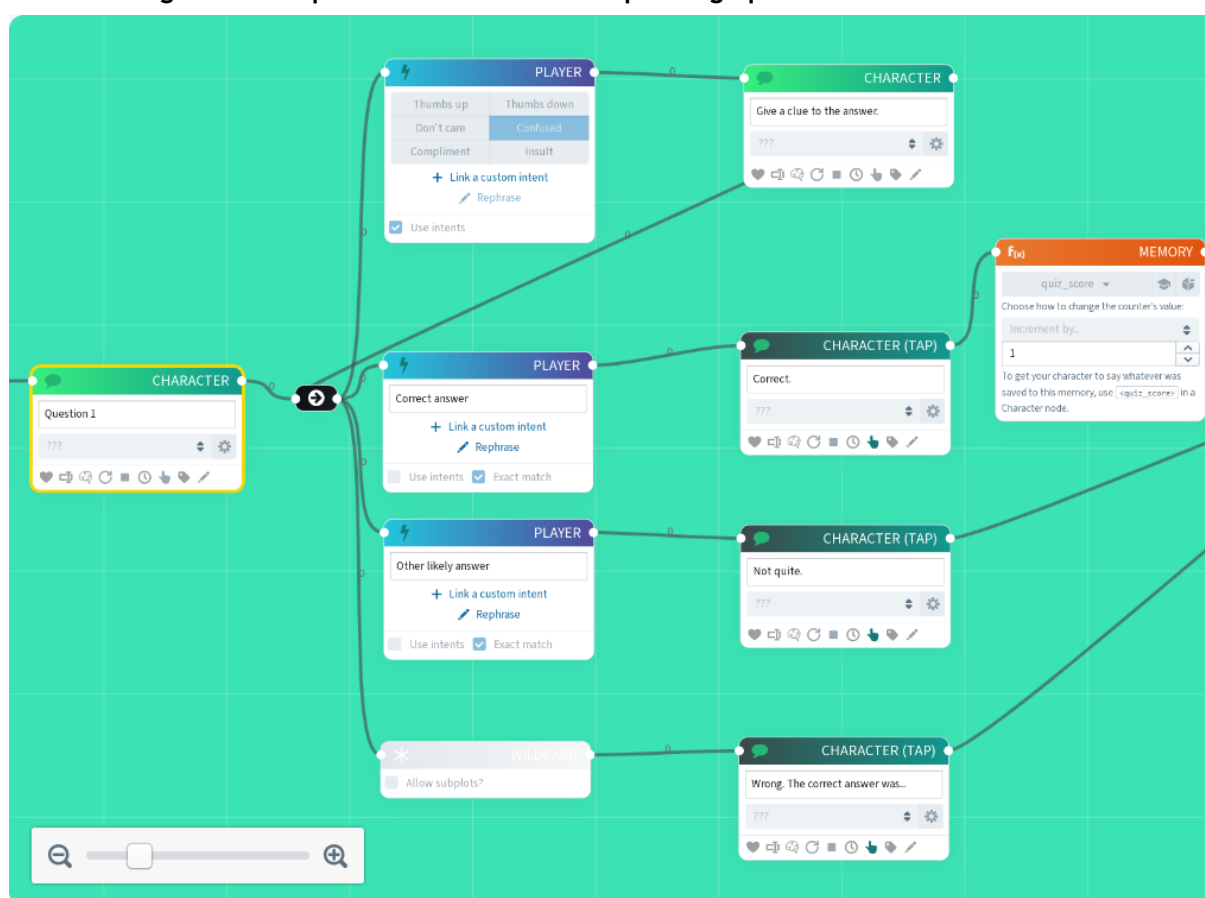
2.1.3. CHARISMA.AI

Charisma.ai est une solution axée sur la création de personnages et de récits interactifs, utilisant l'intelligence artificielle et des graphes narratifs similaires à des arbres de décision, un exemple étant illustré en figure 11. Cette approche permet de contrôler précisément tous les aspects du personnage et le déroulement de l'histoire, évitant ainsi des comportements indésirables de l'IA. Cette solution est particulièrement utile pour le développement de jeux narratifs où le contrôle du scénario est crucial.

Comme présenté sur son site internet (<https://charisma.ai/>), Charisma.ai a établi des partenariats avec plusieurs industries, tels que Warner Bros dans le cinéma et la British

Broadcasting Corporation dans le journalisme. La construction du personnage se fait directement sur leur plateforme en ligne, avec une documentation détaillée et des tutoriels vidéo facilitant la prise en main pour la création du graphe narratif et son application dans le moteur de jeu sélectionné. Les PNJs créés peuvent alors effectuer des actions et échanger avec le joueur, les rendant véritablement interactifs. Une communauté active sur Discord et un support disponible offrent une assistance supplémentaire aux développeurs.

Figure 11 : Capture d'écran d'un exemple de graphe narratif de Charisma.ai



Source : <https://charisma.ai/docs/examples/4-question-quiz>

Cependant, Charisma.ai est une solution payante. Un mois d'essai gratuit est offert avec 50'000 crédits, lesquels se réduisent à chaque interaction avec le personnage. Après l'essai ou l'épuisement des crédits, il est nécessaire d'en acheter de nouveaux. Les tarifs commencent à 5 dollars pour obtenir de nouveau 50'000 crédits, et les suivants augmentent rapidement pour passer à 20, 100, puis 500 dollars. Selon l'utilisation de cette solution, comme lors d'une phase de test par exemple, les crédits peuvent alors s'épuiser très vite. De plus, la page de tarification de Charisma.ai indique que l'essai gratuit est conçu pour être utilisé de manière limitée, ce qui peut poser un problème pour le prototype de ce travail de Bachelor, qui va nécessiter de nombreuses interactions avec le personnage de l'infirmière.

Charisma.ai n'est pas disponible sur l'Unity Asset Store, mais sur GitHub, une plateforme de développement logiciel. Il nécessite l'installation de deux ressources distinctes pour Unity : le « Plug 'n' Play » et le « Software Development Kit ». Leurs dernières mises à jour datent du 29 mai 2024 au moment de l'écriture de ce rapport, le premier étant sorti en 0.1.3 et le second en 0.2.1, ce qui suggère qu'ils sont encore en phases de test. Enfin, la version d'Unity recommandée est à partir de la 2021.3.

Plusieurs langues sont supportées par Charisma.ai, dont le français. Cependant, il est difficile de savoir si les voix disponibles pour les personnages le sont également. Bien que des PNJs soient fournis avec la solution pour Unity, il n'existe pas de documentation sur la manière de paramétrer des modèles personnalisés pour qu'ils fonctionnent avec Charisma.ai. En conclusion, cette solution semble être davantage orientée vers un développement basé sur la narration, pour offrir des expériences riches en histoire, mais le modèle économique pourrait ne pas être approprié pour les besoins du prototype de ce Bachelor.

2.1.4. GRAIL

Figure 12 : Logo de la solution Grail



Source : <https://assetstore.unity.com/packages/tools/behavior-ai/grail-ai-framework-tiered-pricing-258038>

Grail est aussi une solution récente, mise en ligne pour la première fois sur l'Unity Asset Store le 26 juin 2023. Cependant, elle n'a été mise à jour que le 25 juillet 2023. Grail, dont le logo est visible en figure 12, vise à faciliter la construction de l'intelligence artificielle des PNJs, peu importe les plateformes de développement utilisées. Elle incorpore quatre algorithmes d'IA, dont les FSMs, Utility AI et le GOAP. De plus, cette solution vise à faciliter la gestion de plusieurs PNJs en même temps et offre un outil pour vérifier et corriger d'éventuelles erreurs de comportement lors de la phase de test du jeu.

Selon le site internet officiel de Grail (<https://grail.com.pl/>), la solution a été intégrée dans trois jeux indépendants, tous disponibles sur la plateforme de distribution de jeux en ligne « Steam ». De plus, une documentation détaillée est disponible, ainsi qu'une scène de test pour

obtenir un premier aperçu des capacités de l'outil. Pour utiliser Grail, la version 2021.3.9 d'Unity est recommandée.

Toutefois, même si les étapes sont données en détail dans la documentation pour modifier certains aspects de la solution, on peut reprocher l'absence de vidéos tutoriels ou explicatives pour mieux comprendre le fonctionnement du produit. Cela donne l'impression qu'il est nécessaire de connaître certains aspects du moteur de jeu utilisé, ainsi qu'un certain apprentissage de Grail pour exploiter pleinement son potentiel. De plus, cette solution n'intègre pas la possibilité de tenir des conversations avec les PNJs créés.

Bien que la version téléchargeable sur l'Unity Asset Store soit gratuite, débloquer toutes les fonctionnalités de Grail nécessite l'acquisition d'une licence. Elle est offerte pour un usage académique, mais devient très vite coûteuse dans le cas d'une utilisation commerciale. Enfin, même si un forum existe sur le site officiel de la solution, il n'y a que très peu de publications, et on peut regretter l'absence de revues utilisateurs sur l'Unity Asset Store. Tous ces éléments donnent l'impression que Grail est plus spécifiquement destiné à des développeurs de jeux vidéo ayant déjà de l'expérience et des compétences dans la création d'IA pour les PNJs.

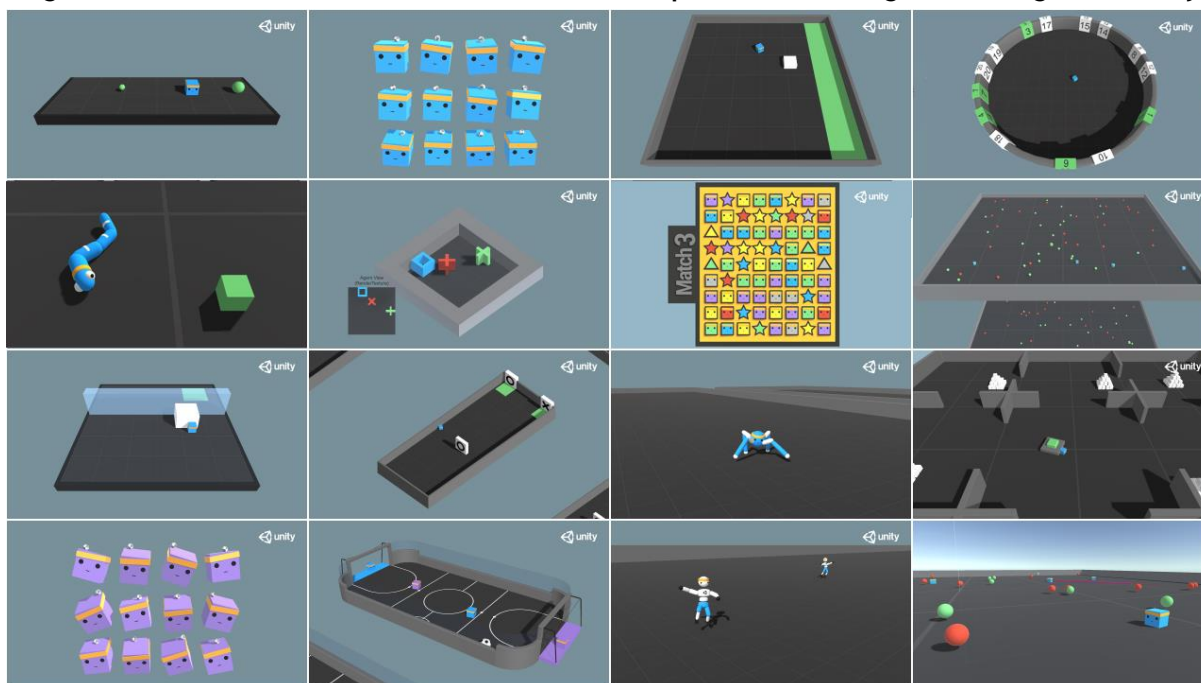
2.1.5. UNITY MACHINE LEARNING AGENTS

Les agents d'apprentissage automatique d'Unity, ou « Unity Machine Learning Agents » en anglais, abrégé en « ML-Agents », sont des outils développés et mis à disposition par Unity. Ayant déjà été mise en ligne avec une première version de test en 2017, cette solution a officiellement été publiée en 2020, et sa dernière mise à jour remonte au 9 octobre 2023 sur GitHub. Cet outil open-source permet de créer des personnages intelligents grâce à des techniques d'apprentissage automatique. Le code source est disponible gratuitement à l'utilisation, à condition d'effectuer les référencements nécessaires, notamment dans le cadre d'une recherche scientifique. Pour utiliser cette solution avec la dernière mise à jour, il est recommandé d'avoir la version 2023.2 ou supérieure d'Unity, même si la version minimum supportée est à partir de la 2022.3.

Les ML-Agents d'Unity utilisent un système de récompenses pour l'apprentissage de ses personnages, leur permettant d'acquérir des comportements totalement autonomes pour exploiter de manière optimale un tel système s'il est bien configuré. En effet, un objectif est donné au PNJ, et chaque action qui s'en rapproche est récompensée par un score positif, tandis que les obstacles rencontrés ou comportements inappropriés sont sanctionnés par des points négatifs. En configurant des environnements d'entraînement de plus en plus difficiles, le personnage finit par prendre les meilleures décisions face à une situation donnée. Cela est particulièrement utile pour gérer des problèmes complexes, comme trouver le chemin le plus court ou même animer le corps du personnage de manière réaliste, pour ce dernier par exemple afin d'éviter les chutes ou se redresser de manière optimale dans le cas contraire. Tout cela est possible grâce à une perception

avancée de l'environnement par le PNJ et évite aux développeurs de devoir coder ces comportements complexes.

Figure 13 : Diverses scènes de tests démontrant les possibilités d'usage des ML-Agents d'Unity



Source : <https://unity-technologies.github.io/ml-agents/Learning-Environment-Examples/>

Pour apprendre à utiliser cet outil, Unity possède un cours de tutoriel sur leur plateforme « Unity Learn », guidant pas-à-pas chaque étape du processus d'implémentation pour parvenir à créer de premiers personnages intelligents. Des scènes de tests sont également disponibles, dont certaines mettent en lumière la coordination entre plusieurs personnages pour réussir un objectif donné, démontrant une multitude de scénarios d'application, comme il est possible d'observer à la figure 13. Bien qu'un site officiel de documentation soit disponible, il peut y être assez difficile à naviguer dans un premier temps. De plus, il est compliqué de trouver des avis d'internautes, et certains commentaires en ligne laissent suggérer que la documentation officielle, notamment la procédure d'installation, pourrait ne pas être à jour.

L'entraînement des personnages peut prendre du temps, surtout si les objectifs sont difficiles et si une faible marge d'erreur est recherchée. Étant donné que cette solution vise à construire les comportements des PNJs, il n'y a pas de fonctionnalités qui permettent d'échanger par oral avec eux. Enfin, l'utilisation des ML-Agents d'Unity nécessite l'installation de logiciels externes comme Python, ce dernier étant indispensable pour la phase d'entraînement. Cela peut constituer un obstacle pour ceux n'ayant pas de connaissances préalables dans ce langage de programmation, surtout lorsqu'il n'y a pas eu de cours dédiés durant ma formation à la HES-SO. Néanmoins, cette solution reste intéressante à explorer pour toute personne souhaitant expérimenter l'apprentissage de comportements autonomes sur Unity.

2.1.6. TOTAL AI

Figure 14 : Logo de Total AI



Source : <https://github.com/TotalAI/TotalAI/blob/master/TotalAI/Editor/Images/TotalAILogo.png>

Total AI, dont le logo est illustré à la figure 14, est une solution similaire à Grail en termes de fonctionnalités proposées. En effet, tout comme ce dernier, Total AI vise à faciliter la construction de l'IA des PNJs, en se concentrant uniquement sur le moteur de jeu Unity. Trois algorithmes d'intelligence sont actuellement implémentés : les FSMs, Utility AI et le GOAP. De plus, un modèle d'agents d'apprentissage automatique, basé sur la solution d'Unity, est en cours de développement. Total AI se concentre sur la construction des comportements des PNJs, et n'est donc pas conçu pour converser avec eux.

Cette solution est open-source et elle est mise à disposition gratuitement pour quiconque souhaitant l'utiliser. Sa première publication sur GitHub date de la fin de l'année 2020, mais elle n'a pas reçu de mise à jour depuis fin janvier 2021. Total AI se considère toujours comme en phase d'alpha. Il peut donc potentiellement être instable et comporter des erreurs de programmation.

Cependant, cela n'empêche pas Total AI de disposer d'une bonne documentation, bien que partiellement incomplète, ainsi que d'une vidéo d'installation faisant office de tutoriel pour une première implémentation. Des scènes de démonstration sont également fournies pour donner un aperçu des capacités offertes par cette solution. De plus, la version d'Unity utilisée dans cette vidéo d'installation est la 2020.1.17.

Malgré ces différents points, on peut regretter que le potentiel prometteur de Total AI semble laissé à l'abandon, compte tenu de l'absence de mises à jour. Bien qu'il existe une communauté sur Discord qu'il est possible de rejoindre, il est difficile de savoir si elle est encore active à ce jour.

2.1.7. IBRAIN

IBRAIN est une solution payante développée par l'entreprise « Vit Labs Mobile », ayant le pseudo « VIT BIT » sur l'Unity Asset Store. Destiné à faciliter la création de PNJs intelligents et conscients de leur environnement, IBRAIN a été initialement publié le 25 avril 2018, et sa dernière mise à jour remonte au 14 juin 2018. Bien qu'il soit l'un des outils les plus anciens parmi ceux listés, il peut être utile pour les développeurs cherchant une première base d'IA pour leurs PNJs. IBRAIN se concentrant uniquement sur le développement de comportements, il n'intègre donc pas de fonctionnalités pour discuter avec les personnages créés.

Figure 15 : Illustration des comportements que les PNJs développés par IBRAIN peuvent prendre



Source : <https://assetstore.unity.com/packages/tools/behavior-ai/ibrain-npc-character-a-i-116128>

Selon la vidéo de présentation présente sur sa page, les personnages développés avec cette solution adaptent leurs comportements en temps réel, en fonction des paramètres donnés et de l'environnement, comme illustré en figure 15. De plus, des exemples de personnages sont inclus afin de tester les fonctionnalités disponibles. La solution est compatible avec les versions 2017.1.13 et ultérieures d'Unity. Malheureusement, la documentation n'est accessible qu'après l'achat de cet outil, et il n'est pas possible de le tester au préalable. Toutefois, le prix ne s'élève qu'à environ 5 dollars, et le créateur d'IBRAIN est joignable par courriel pour offrir un support en cas de besoin.

Malgré le faible nombre d'avis utilisateurs, ceux disponibles sont positifs et certains sont même plus récents que la dernière mise à jour de la solution. Cependant, le fait qu'il n'ait plus eu de suivi depuis 2018 peut susciter des inquiétudes quant à la stabilité et la modernité de l'outil par rapport aux concurrents cités précédemment. En outre, la description succincte, le manque de vidéos tutoriels et l'impossibilité d'accéder à la documentation avant l'achat rendent difficile l'évaluation des capacités offertes par IBRAIN. Néanmoins, il peut être une option envisageable pour ceux qui souhaitent débiter dans l'implémentation d'une IA pour leurs PNJs.

2.2. MATRICE DE DÉCISIONS

Après ce bref aperçu des différentes solutions disponibles, il est nécessaire de les comparer afin de déterminer laquelle répondra le mieux aux besoins de ce travail de Bachelor. Pour ce faire, j'ai défini plusieurs critères importants à prendre en compte pour choisir la meilleure solution. La figure 16 présente la matrice de décisions résultant de mes recherches et de l'analyse de ces différents critères.

Figure 16 : Matrice de décisions

	Convai		Inworld		Charisma.ai		Grail		Unity ML-Agents		Total AI		IBRAIN	
	Evaluation	Normalisation	Evaluation	Normalisation	Evaluation	Normalisation	Evaluation	Normalisation	Evaluation	Normalisation	Evaluation	Normalisation	Evaluation	Normalisation
Suivi	Mis à jour récurrent	3	9	3	9	3	9	2	6	2	6	1	3	3
	Communauté active	3	6	3	6	3	6	4	2	4	2	4	2	4
Prise en main														
	Tutoriels	3	9	3	9	2	6	2	6	3	9	2	6	1
	Documentation	3	9	3	9	3	9	3	9	2	6	2	6	6
Intelligence artificielle	Version Unity 2021.3.27f1	1	1	3	3	3	3	3	3	1	3	3	3	3
	Navigation intelligente	3	6	1	3	2	6	3	9	3	9	3	9	9
Prix	Interaction avec l'environnement	3	9	1	3	3	9	3	9	3	9	3	9	6
	Open-source	1	2	1	2	1	2	1	2	3	6	3	6	1
Optionnel	Quota illimité	1	1	1	1	1	1	3	3	3	3	3	3	3
	Dialogue en français	2	3	6	1	2	4	1	2	1	2	1	2	2
Total	Avis des utilisateurs	1	3	2	2	1	1	1	1	1	1	1	1	2
		61	49	56	54	56	52	43						

Poids : de 1 à 3, 1 = optionnel ; 2 = bien ; 3 = indispensable
Évaluation : de 1 à 3, 1 = ne remplit pas le critère ; 2 = peut remplir le critère ; 3 = remplit exactement le critère

Source : données de l'auteur

Chaque critère d'évaluation possède un poids initial allant de 1 à 3, définissant ainsi son importance dans le cadre de cette comparaison. Conformément à l'échelle présente dans la figure 16, les poids sont attribués ainsi : 1 correspond à un critère optionnel, 2 signifie qu'il est utile, et 3 indique que le critère est indispensable pour la réalisation du prototype.

En plus du poids, les solutions sont évaluées individuellement pour chaque critère dans la colonne en italique « Évaluation ». Une seconde échelle, allant également de 1 à 3, précise l'attribution des points : 1 signifie que la solution ne remplit pas le critère, 2 indique qu'elle le remplit partiellement, et 3 qu'elle le satisfait pleinement.

La colonne « Normalisation » est calculée en multipliant l'évaluation de chaque critère par son poids correspondant, donnant ainsi sa valeur réelle. Par exemple, la catégorie « Tutoriels » possède une valeur de 3, et Convai pour ce critère a une évaluation de 3. La cellule de normalisation pour cette catégorie et cette solution contient alors le produit de ces deux nombres : 9. Ainsi, le total des points d'une solution est calculé en additionnant toutes ses nouvelles évaluations après normalisation.

2.2.1. SUIVI

La catégorie « Suivi » évalue l'actualité de la solution, ce qui est essentiel pour garantir sa pertinence et son efficacité. La fréquence des mises à jour forme le premier critère. Les solutions ayant l'évaluation maximale signifie qu'elles ont été mises à jour dans le courant de l'année de la rédaction de ce travail de Bachelor. Des mises à jour récentes démontrent que le produit est toujours d'actualité, et est donc un critère important, représenté ainsi avec un poids de 3.

L'existence d'une communauté active autour de la solution est le second critère de cette catégorie. Ce soutien peut être particulièrement utile en cas de problèmes rencontrés lors de l'utilisation de l'outil, et c'est pourquoi il a le poids de 2. Un support réactif ou l'accès à une plateforme de messagerie où les utilisateurs peuvent s'entraider sont des aspects qui contribuent à obtenir une meilleure note pour ce critère.

2.2.2. PRISE EN MAIN

Cette catégorie évalue les divers éléments qui contribuent à faciliter l'utilisation de la solution. Ainsi, le premier critère est la disponibilité de tutoriels, qu'ils soient vidéo ou écrits, permettant une prise en main initiale de l'outil étape par étape. En plus de cela, il faut tenir compte de la documentation, ce qui constitue le second critère. En effet, elle est essentielle pour comprendre le fonctionnement de la solution et d'y trouver des réponses en cas de questionnements. Ces deux aspects sont indispensables à l'utilisation de l'outil et possèdent alors chacun un poids de 3.

Un troisième critère détermine la compatibilité avec la version 2021.3.27 d'Unity, sur lequel Sim3D est construit. Une solution compatible évite la nécessité de mettre à jour ce projet et ses

dépendances, ce qui pourrait autrement nécessiter des ajustements de certaines fonctionnalités. Cependant, après discussion avec mon professeur responsable, ce critère ne possède qu'un poids de 1. Selon lui, et comme il sera démontré ensuite dans le chapitre suivant, les fonctionnalités de Sim3D étant relativement simples et se limitant à une seule scène, les problèmes liés à une mise à jour d'Unity ne devraient pas être trop difficiles à corriger.

2.2.3. INTELLIGENCE ARTIFICIELLE

Pour mieux définir l'IA nécessaire à la réalisation du prototype, et après avoir passé en revue les différents outils de la matrice, j'ai pu préciser les caractéristiques essentielles au personnage que la solution choisie devra fournir. L'intégration d'une navigation intelligente, permettant au PNJ de se déplacer efficacement dans la scène de jeu, constitue le premier critère. Cela inclut la capacité à éviter les obstacles et se diriger vers les objectifs, le tout en tenant compte de l'environnement. Une telle fonctionnalité est nécessaire pour que le personnage puisse accomplir les tâches qui lui sont données. Ce critère permet alors de filtrer les solutions qui n'implémentent pas de déplacements pour leurs PNJs.

Le second critère concerne l'interaction avec l'environnement, c'est-à-dire la possibilité au personnage d'interagir avec les objets présents dans la scène de jeu. Concrètement, cela signifie que le PNJ doit pouvoir saisir et utiliser les objets conformément au scénario pour accomplir les objectifs et progresser dans le jeu. L'intelligence artificielle étant une catégorie très importante pour la construction du prototype, chacun de ces critères possède un poids de 3.

2.2.4. PRIX

Cette catégorie vise à identifier les solutions disponibles gratuitement ou, du moins, celles qui ne présentent pas une limitation trop restrictive de leurs fonctionnalités. Le premier critère détermine si le code de l'outil est open-source. Selon M. Antoine Widmer, cet aspect est particulièrement intéressant dans le domaine de la recherche, car cela permet d'éviter des surprises liées à une augmentation des frais si une solution payante décide de modifier ses tarifs. En raison de son utilité, le poids de ce critère s'élève à 2.

Le critère suivant évalue si la solution n'est pas limitée par un quota d'utilisation. En général, celles ayant obtenu la note maximale pour la partie open-source ne sont pas soumises à des quotas, ce qui leur permet d'avoir également l'évaluation la plus élevée pour ce second critère. Afin de ne pas pénaliser les solutions n'étant pas open-source, le critère des quotas d'utilisation a un poids de 1.

2.2.5. OPTIONNEL

La catégorie optionnelle regroupe deux éléments intéressants à prendre en compte lors de la sélection de la solution, et qui ne rentrent pas dans les catégories précédentes. La capacité de

dialoguer en français avec le personnage virtuel compose le premier critère. Cette fonctionnalité met en avant les solutions qui utilisent des LLMs pour construire leurs PNJs, permettant ainsi de donner une dimension interactive au prototype, car le joueur aura la possibilité de communiquer avec l'infirmière. Bien que le sujet de ce Bachelor cherche l'implémentation de la logique derrière le PNJ, la possibilité de renforcer cet aspect par la communication entre le joueur et le personnage met ainsi en avant l'intelligence de ce dernier, enrichissant alors l'expérience utilisateur et apportant une valeur ajoutée significative au prototype. De plus, la précision de la langue française permet de filtrer les solutions limitées à l'anglais. Ce critère, bien que non indispensable, garde tout de même un poids de 2 pour souligner son utilité.

L'avis des utilisateurs forme le second critère, et est évalué notamment sur leur disponibilité, leur positivité, et surtout leur actualité. En général, l'évaluation se base principalement sur les avis des utilisateurs de l'Unity Asset Store, car il est possible d'y écrire directement des revues sur les produits. Elles sont alors facilement visibles et potentiellement plus pertinentes que des messages postés sur des forums non spécifiques à l'outil. Ces avis fournissent de précieuses informations concernant l'efficacité et la popularité d'une solution, mais restent un critère plutôt facultatif face aux autres catégories, ce qui est illustré par son poids de 1.

2.2.6. ANALYSE DU RÉSULTAT

Maintenant que les différentes catégories ont été décrites, il est possible d'observer la solution ayant obtenu la meilleure évaluation. Ainsi, celle avec le plus grand nombre de points sous la ligne « Total » est Convai.

En effet, la dernière version de cette solution est sortie très récemment, le 21 juin 2024, en plus d'être régulièrement mise à jour, un aspect que j'ai pu constater tout au long de l'écriture de mon travail de Bachelor. De plus, comme mentionnés dans le sous-chapitre qui lui est dédié, un support est disponible pour répondre en cas de problèmes, ainsi qu'une communauté accessible sur leur serveur Discord. Ces éléments font que Convai obtient l'évaluation maximale pour la catégorie « Suivi ».

Pour la partie « Prise en main », Convai dispose de tutoriels vidéo et d'une documentation détaillée, tant pour leur plateforme en ligne que pour son utilisation sur Unity, ce qui donne à ces deux critères le maximum de points. Seule la version d'Unity n'est pas respectée car, pour rappel, cette solution ne fonctionne qu'à partir de la 2022.3.0. Cela nécessitera donc de mettre à jour le projet Sim3D.

En ce qui concerne l'intelligence artificielle, Convai présente dans ses vidéos et sa documentation des personnages qui effectuent des actions sous la commande du joueur, mais seul le critère de la navigation intelligente n'est pas totalement rempli. En effet, le joueur peut demander au personnage de le suivre ou d'aller chercher un objet, mais sans avoir utilisé la solution

au préalable, il est impossible de savoir si le PNJ prend en compte l'environnement et ses obstacles pour les contourner. Cela donne à Convai la note de 3 pour le critère de l'interaction avec l'environnement, et la note de 2 pour la navigation intelligente.

Convai obtient l'évaluation minimale dans la catégorie du prix. Ce n'est pas une solution open-source et le code n'est pas mis à disposition pour modification, avec une partie de la construction du PNJ qui doit se faire sur leur plateforme en ligne. De plus, un quota journalier peut limiter son utilisation. Même si la limitation de concurrence ou le stockage réduit pour la banque de connaissances du personnage ne vont pas réellement impacter le prototype, seul le quota peut poser un problème, mais rien d'insurmontable contrairement aux autres solutions payantes.

Enfin, pour la catégorie optionnelle, Convai se concentre sur l'aspect de la communication avec le personnage, sans omettre les autres éléments que ce dernier peut accomplir, notamment les actions et les déplacements. Cette solution apparaît ainsi comme un outil plus multifonctionnel, plutôt que purement focalisé sur l'échange comme Inworld. De plus, Convai semble être la seule à posséder autant d'avis positifs et réguliers sur sa page de l'Unity Asset Store, ce qui inspire confiance. Convai reçoit donc le maximum de points pour cette dernière catégorie.

En conclusion, toutes ces évaluations démontrent que Convai est une solution moderne et certainement capable de répondre aux objectifs et problématiques définis au chapitre numéro 1 de ce rapport, ou du moins de permettre facilement la construction de l'intelligence du personnage de l'infirmière dans Sim3D.

2.2.7. REMARQUES SUPPLÉMENTAIRES

La construction de la matrice de décisions a nécessité plusieurs ajustements tout au long de l'écriture de ce rapport, tant au niveau des points accordés qu'à la définition des catégories.

Au départ, la catégorie sur l'IA ne contenait qu'un seul critère générique, plaçant ainsi Inworld en première position. Cependant, après une première utilisation de leur plateforme en ligne, des limitations non mentionnées dans leur site officiel ou dans leur documentation ont été identifiées. Notamment, cela a permis de confirmer qu'Inworld ne supporte pas la langue française. De plus, les vidéos de démonstration ou de tutoriels ont rapidement montré que les personnages ne pouvaient pas réaliser d'actions, si ce n'est d'échanger avec le joueur. En conséquence, l'évaluation d'Inworld a été revue à la baisse. Cela a conduit à préciser la catégorie sur l'intelligence artificielle avec l'ajout de ses deux critères actuels, ainsi qu'à affiner celui concernant le dialogue avec le PNJ pour y inclure le français.

En plus de ces changements, la catégorie du prix favorisait les outils gratuits, et ce sont alors les ML-Agents d'Unity qui se sont retrouvés en première position. Cette solution semblait idéale, étant donné qu'elle est open-source et soutenue par Unity. Cependant, l'absence d'interactions entre le

joueur et le personnage signifiait que l'infirmière créée avec cet outil serait devenue totalement autonome, entreprenant des actions de manière indépendante, tandis que le joueur serait relégué à un rôle de spectateur. De plus, l'installation de cet outil est particulièrement laborieuse, car elle nécessite de télécharger les versions 3.7 ou 3.8 de Python, toutes deux n'étant plus disponibles sur son site officiel. Elles sont indispensables pour utiliser une sortie des ML-Agents compatible avec la version d'Unity employée par Sim3D, évitant ainsi la nécessité de mettre à jour le moteur de jeu. Ces éléments ont permis de préciser les critères de prix et diminuer leur poids afin de moins pénaliser les solutions payantes. De plus, cette expérience a également fourni un argument supplémentaire pour réduire l'importance de la version d'Unity dans la matrice. Cependant, même après cette réévaluation, les ML-Agents d'Unity restent une option viable, se classant en deuxième position.

Charisma.ai a été découvert au cours des ajustements de la matrice de décisions, après que les autres solutions avaient déjà obtenu une évaluation initiale. Bien qu'elle soit payante, Charisma.ai maintient un score élevé en raison de ses fonctionnalités. Quant aux autres outils évalués, ils se révèlent particulièrement adaptés à la mise en application des algorithmes de PNJs présentés au chapitre 1. Contrairement à Convai, Inworld et Charisma.ai, qui se concentrent davantage sur un aspect chatbot, ou à Unity ML-Agents, orienté vers l'apprentissage de comportements, les solutions Grail, Total AI et potentiellement IBRAIN sont plus adaptées pour intégrer ces algorithmes. Cependant, ces dernières n'utilisent pas une forme d'intelligence artificielle comme les LLMs, contrairement aux premières solutions citées.

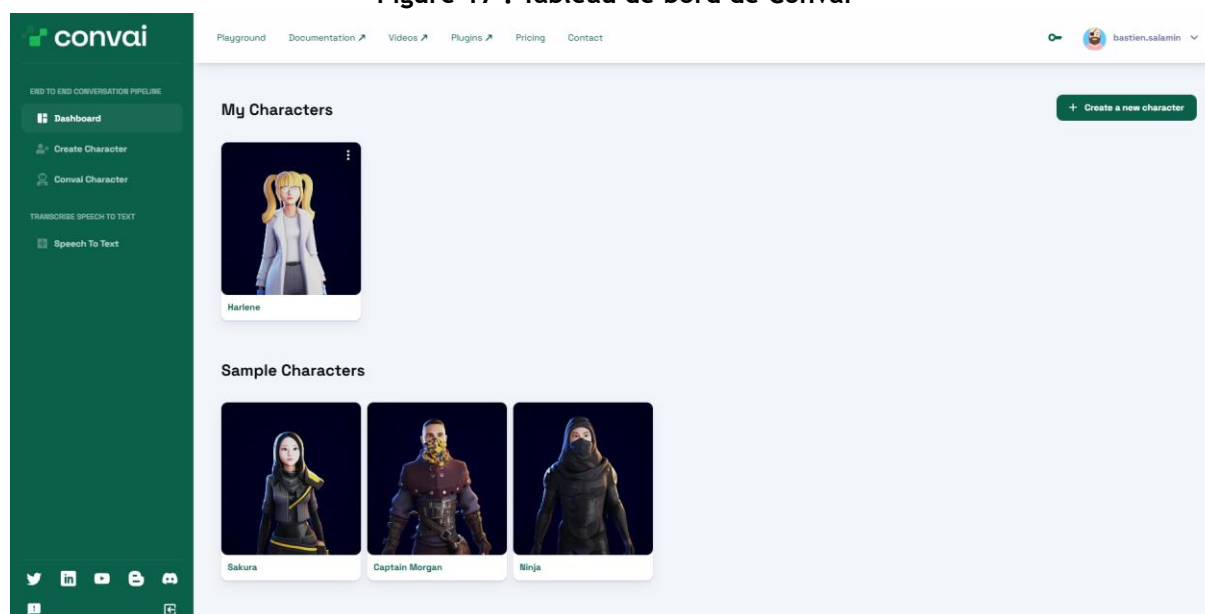
3. IMPLÉMENTATION DE LA SOLUTION

Après avoir analysé différentes solutions facilitant l'implémentation d'un personnage intelligent, Convai s'est distinguée par les fonctionnalités qu'elle propose. Cet outil permet notamment de créer des PNJs capables d'effectuer des actions, de se déplacer dans la scène de jeu et d'échanger avec le joueur, tout en offrant un support actif entre autres aspects. Ce chapitre se concentre sur l'implémentation pratique de cette solution dans le projet Sim3D. L'utilisation de la plateforme de Convai, la mise à jour de Sim3D, ainsi que l'installation et le développement du personnage seront les principales étapes abordées. L'objectif est de fournir une documentation détaillée pour comprendre mon intégration et mon utilisation de Convai, tout en présentant les résultats obtenus et les défis rencontrés, afin de pouvoir établir en fin de chapitre un bilan de cette solution. La documentation officielle de Convai (<https://docs.convai.com/api-docs>) est la principale source d'information employée tout au long de cette partie.

3.1. PLATEFORME CONVAI

3.1.1. DÉMARRAGE

Figure 17 : Tableau de bord de Convai



Source : <https://convai.com/pipeline/dashboard>

À la création d'un compte sur le site <https://convai.com/>, le tableau de bord s'affiche directement à l'utilisateur comme illustré en figure 17. Il est à noter que le site est entièrement en anglais. Convai met à disposition des PNJs préfabriqués, mais pour les besoins du prototype j'ai trouvé intéressant de commencer avec un nouveau personnage et de tester les fonctionnalités disponibles avec lui. En cliquant sur le bouton vert « + Create a new character » se trouvant juste en-dessous du nom d'utilisateur, cela mène directement sur les pages de création du personnage.

3.1.2. DESCRIPTION DU PERSONNAGE

La création du personnage est divisée en plusieurs catégories. La première se nomme « Character Description », et comporte quatre éléments essentiels : l'identifiant (ID) du PNJ, son nom, sa voix et sa description. La fenêtre de cette catégorie est illustrée en figure 18.

Figure 18 : Onglet « Character Description »

Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

L'identifiant du personnage est automatiquement attribué par Convai, le reste revient à l'utilisateur de remplir. Pour la création de l'infirmière, j'ai décidé de l'appeler « Harlene ». Son nom n'a pas vraiment d'importance pour l'implémentation du prototype, mais reste obligatoire pour créer le personnage sur Convai. La voix sélectionnée est l'une des voix féminines françaises proposées par Convai dans liste déroulante, après définition du langage qui s'effectue dans la catégorie suivante.

Enfin, en ce qui concerne la description du personnage, je suis d'abord parti avec un premier texte simple décrivant son rôle une fois en jeu, en précisant les objectifs attendus par le joueur, dépeint en tant que médecin, et demandés par le scénario de Sim3D, mais sans définir de manière détaillée sa personnalité. Il s'agit de la même description présente dans la figure 18 :

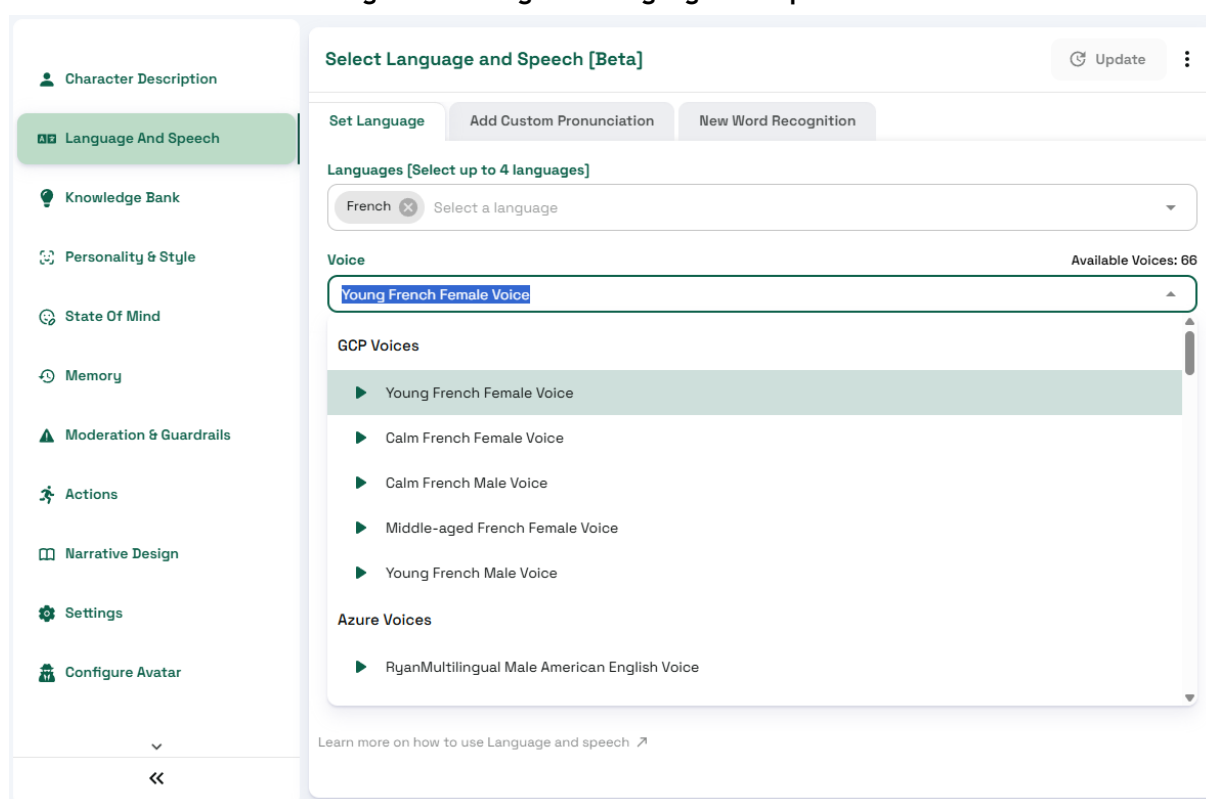
« Tu es une infirmière et tu te trouves actuellement dans la chambre du patient, consultant l'état vital de ce dernier en attendant que le médecin vienne faire l'examen régulier du patient. Tu es impatiente d'aider le médecin à effectuer l'examen de routine. Le médecin te demandera de lui donner les objets suivants, nécessaires à l'examen : le thermomètre, le pulsomètre, le tensiomètre et le stéthoscope. Tu lui donneras chaque objet lorsqu'on te le demandera et tu le remettras dans le chariot lorsque le médecin aura fini de l'utiliser. Après l'examen, le médecin te demandera de déposer un médicament sur la table du patient, et vous quitterez la chambre ensemble en souhaitant une bonne journée au patient. »

patient. Tu es impatiente d'aider le médecin à effectuer l'examen de routine. Le médecin te demandera de lui donner les objets suivants, nécessaires à l'examen : le thermomètre, le pulsomètre, le tensiomètre et le stéthoscope. Tu lui donneras chaque objet lorsqu'on te le demandera et tu le remettras dans le chariot lorsque le médecin aura fini de l'utiliser. Après l'examen, le médecin te demandera de déposer un médicament sur la table du patient, et vous quitterez la chambre ensemble en souhaitant une bonne journée au patient. »

Convai accepte du texte en français, ce qui facilite la rédaction de cette description. La limitation de 1'000 mots est assez élevée et peut permettre de détailler davantage le personnage en fonction des résultats obtenus en jeu, notamment si l'on veut donner une personnalité particulière au PNJ. Le bouton « Generate Backstory » génère du texte en anglais, mais j'ai préféré écrire la description manuellement et en français.

3.1.3. LANGUAGE ET PAROLE

Figure 19 : Onglet « Language And Speech »



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Cette catégorie, illustrée en figure 19, permet de définir la langue qui sera utilisée par le PNJ. Elle est constituée de trois onglets, « Set Language », « Add Custom Pronunciation » et « New Word Recognition ». Le premier onglet est le plus important, car il permet d'assigner au personnage un ou

plusieurs langages, jusqu'à quatre maximums. Cela va permettre ensuite de construire la liste de propositions des voix selon les langues sélectionnées. Cette liste de voix est la même que celle présente dans la première catégorie.

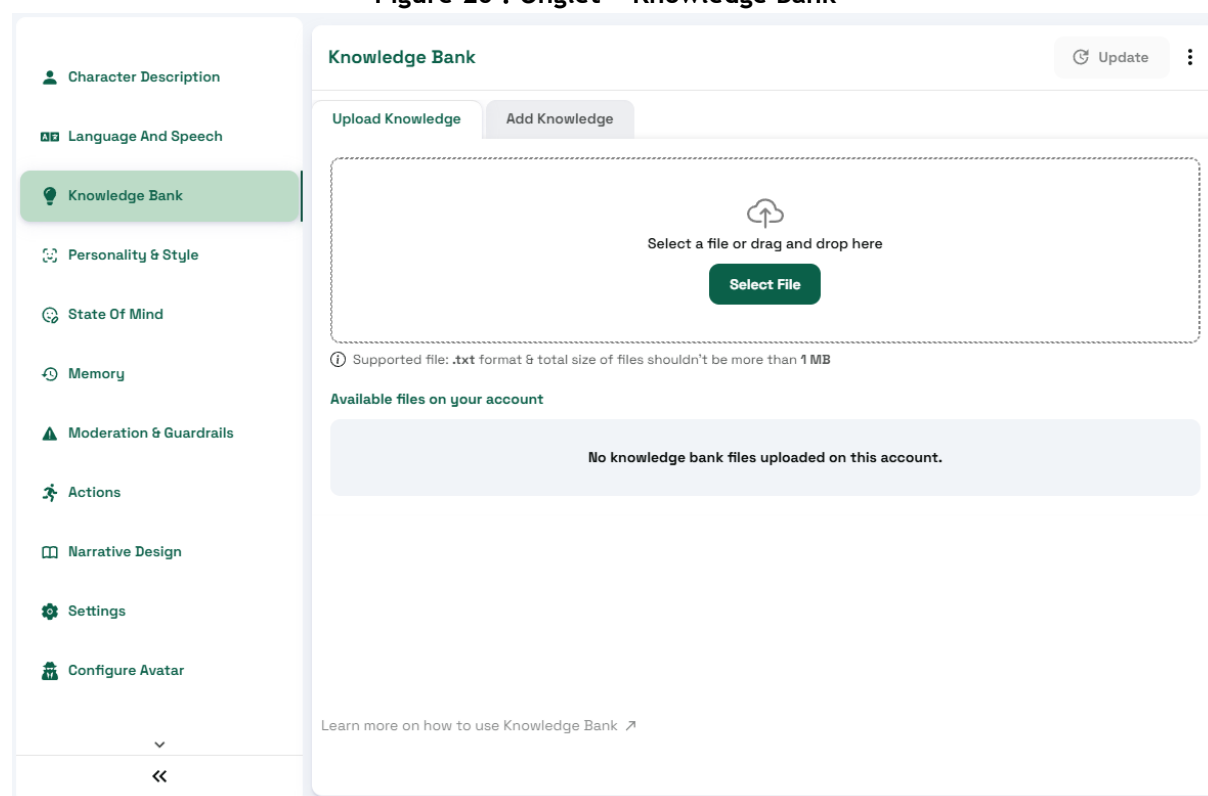
Les deux derniers onglets sont des fonctionnalités qui permettent de personnaliser la prononciation ou d'ajouter du vocabulaire au PNJ, mais elles sont uniquement disponibles pour l'anglais. De plus, comme ces éléments ne sont pas utiles au prototype, ils ne sont donc pas traités dans ce rapport.

3.1.4. BANQUE DE CONNAISSANCES

Afin de développer davantage les informations du personnage, Convai offre une banque de connaissances, utile pour enregistrer par exemple des consignes hospitalières spécifiques à suivre selon l'état du patient. Cette fonctionnalité permet donc de mieux ajuster les réponses du personnage selon les sujets abordés si ces derniers sont enregistrés dans cette banque.

Comme le montre la figure 20, deux onglets sont disponibles. Dans « Upload Knowledge », les utilisateurs peuvent ajouter du contenu textuel via des fichiers au format « .txt » uniquement. Ou bien, avec « Add Knowledge », il est possible d'écrire directement du texte sans limite de caractères, qu'il faudra ensuite sauvegarder, générant ainsi un nouveau fichier sur le compte.

Figure 20 : Onglet « Knowledge Bank »



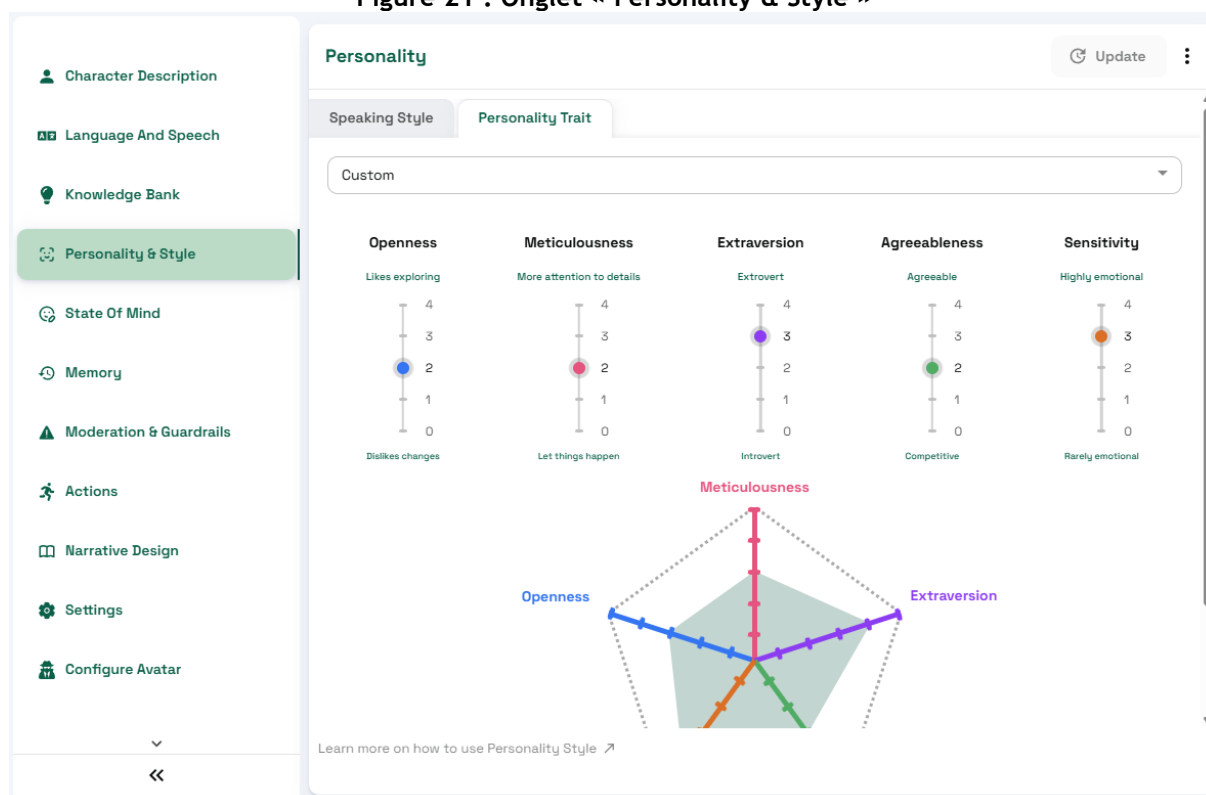
Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Étant donné que, pour le prototype actuel, la description du personnage est suffisamment concise et que je n'avais pas à disposition de consignes réelles d'un hôpital qui peuvent être données à un service infirmier, cette fonctionnalité proposée par Convai n'a pas été utilisée.

3.1.5. STYLE ET PERSONNALITÉ

Cette catégorie comporte deux onglets. Le premier, intitulé « Speaking Style », permet de choisir le mode d'expression du personnage parmi une liste de traits, incluant des styles tels que celui d'un pirate ou d'un cowboy. En plus de cela, il est possible d'écrire des expressions spécifiques que le PNJ utilisera dans ses réponses, afin de personnaliser davantage ses interactions. Ces fonctionnalités, bien qu'amusantes et utiles selon le type de personnage à créer, n'ont pas été modifiées afin de garder une infirmière neutre sans élément de langage particulier.

Figure 21 : Onglet « Personality & Style »

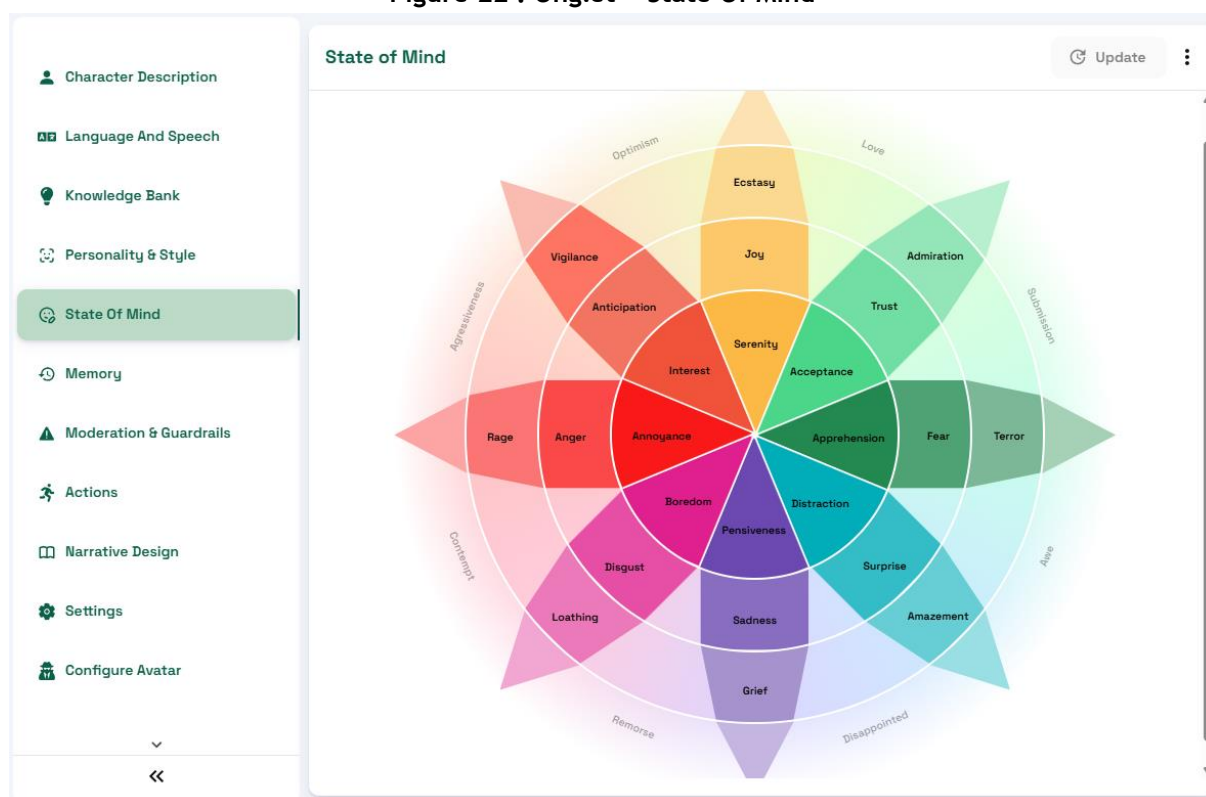


Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

L'onglet suivant « Personality Trait », illustré en figure 21, permet d'ajuster les réponses du PNJ en fonction de cinq tempéraments principaux. Il est possible de modifier ces traits soit en choisissant dans la liste déroulante l'une des personnalités prédéfinies par Convai, soit en ajustant manuellement les cinq échelles disponibles. Pour l'infirmière, j'ai choisi de garder des traits neutres, mais tout en ajoutant un point à l'extraversion et à la sensibilité, comme montré en figure 21, afin de donner au personnage un léger caractère sans en faire une personnalité trop marquée.

3.1.6. ÉTAT D'ESPRIT

Figure 22 : Onglet « State Of Mind »



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

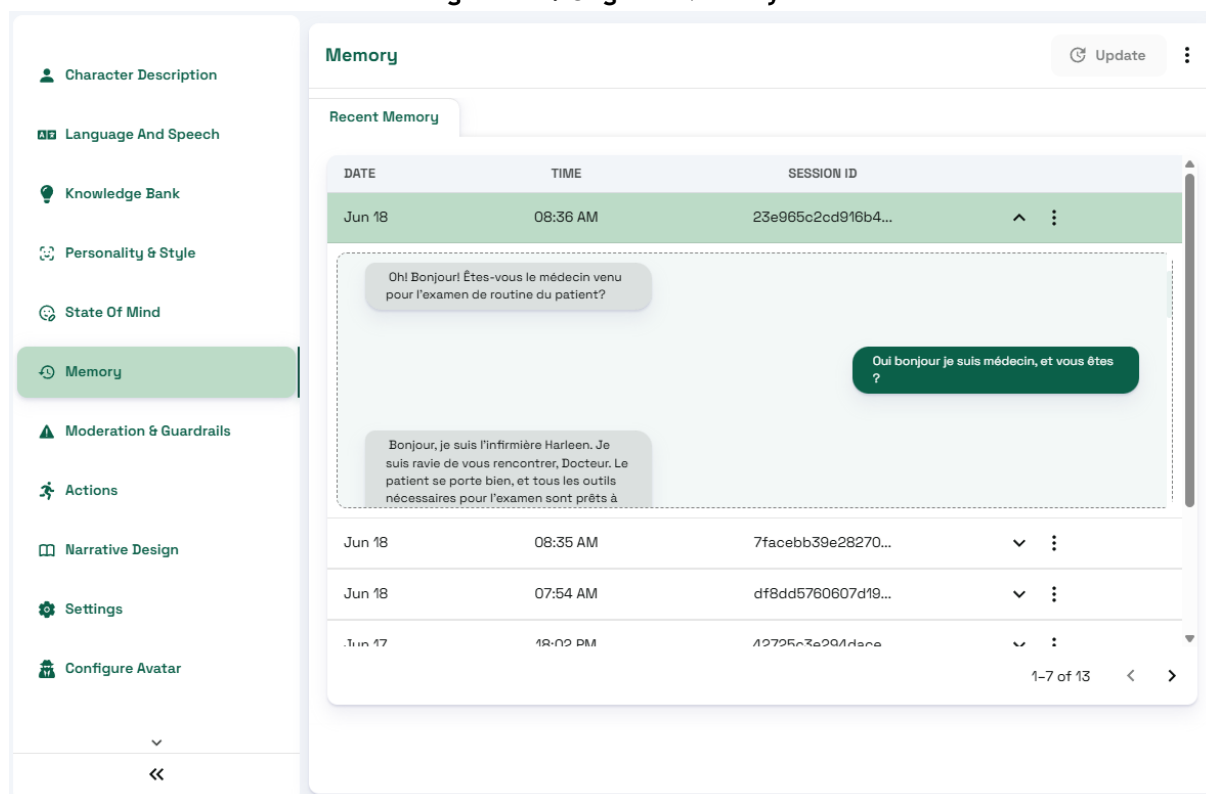
Cette catégorie, présentée en figure 22, permet d'observer l'état d'esprit actuel du personnage durant les discussions effectuées sur la plateforme de Convai, et de suivre les changements qui se déroulent au fil des échanges. Cette roue d'émotions est fournie à titre informatif et n'a pas d'utilisation directe sur Unity.

3.1.7. MÉMOIRE

L'historique des sessions de discussions réalisées avec le personnage est regroupé dans cette catégorie, comme l'illustre la figure 23. Chaque échange effectué sur la plateforme Convai ou en jeu est classé par date et par heure, et un identifiant de session lui est assigné. Lors de la consultation de l'une de ces discussions, il devient possible de télécharger ou de copier la conversation. Cependant, il est impossible de supprimer les précédents échanges avec le PNJ, alors que la documentation indique le contraire. J'ignore si cela est dû à l'utilisation d'un compte gratuit, ou si Convai a supprimé cette fonctionnalité.

Bien que cette page soit intitulée « Mémoire », les sujets de conversation abordés ne sont pas récupérés entre les sessions, ce qui signifie que le PNJ ne se souvient pas des interactions précédentes lorsque l'utilisateur initie une nouvelle session d'échange.

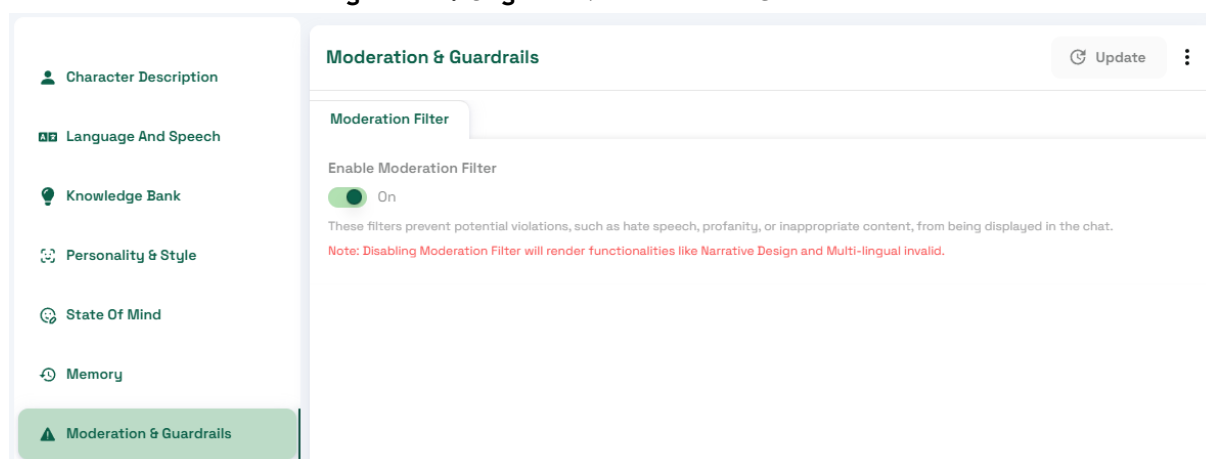
Figure 23 : Onglet « Memory »



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

3.1.8. MODÉRATION

Figure 24 : Onglet « Moderation & Guardrails »

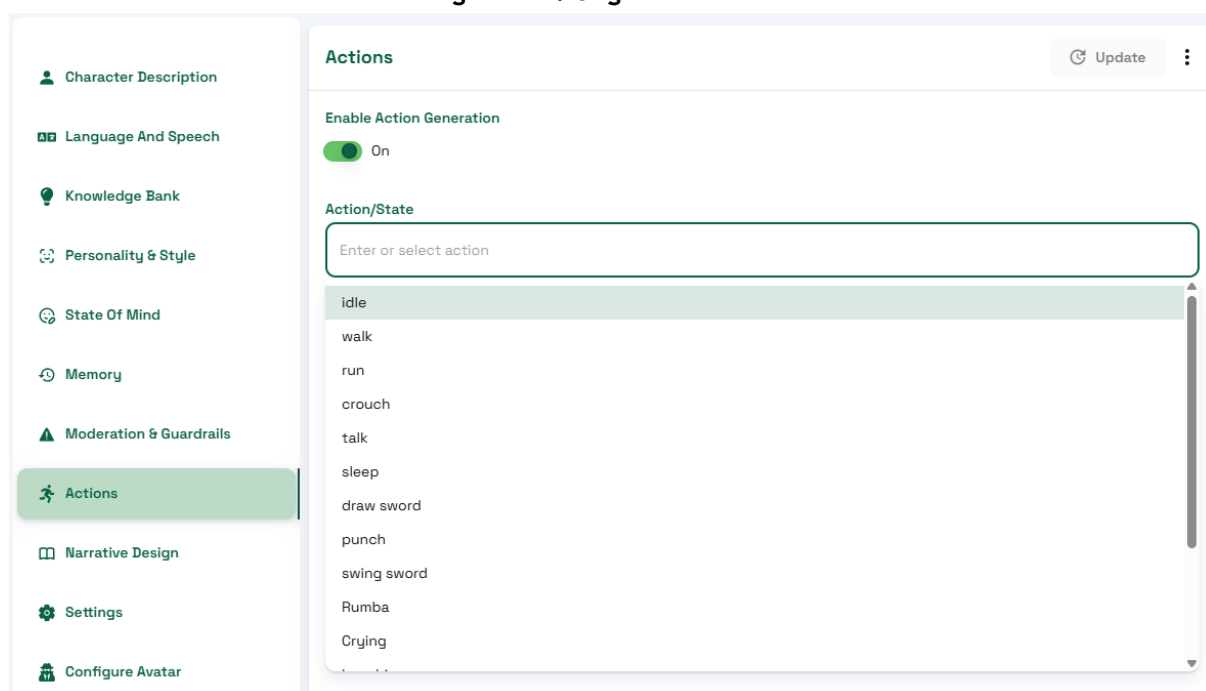


Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Cette catégorie permet d'activer ou de désactiver le filtre qui, comme mentionné sur la figure 24, bloque les vulgarités et les contenus inappropriés dans les réponses du PNJ. Activé par défaut, le désactiver enlève aussi le support multilingue ainsi que d'autres fonctionnalités. Pour les besoins du prototype, le filtre reste actif afin de permettre au personnage de parler français.

3.1.9. ACTIONS

Figure 25 : Onglet « Actions »



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Il est possible de demander au personnage d'effectuer une action pendant une conversation sur la plateforme Convai grâce à cette catégorie. L'option est désactivée par défaut, mais une fois activée, une liste des actions que l'on peut commander au PNJ d'exécuter s'affiche, permettant de sélectionner celles qui sont les plus pertinentes. Cette fonctionnalité est spécifique à la plateforme, et n'affecte pas les actions que le personnage pourra réaliser une fois importé sur Unity.

3.1.10. DESIGN NARRATIF

Une catégorie particulièrement intéressante lors de la création du personnage est le Design narratif. Selon la documentation fournie par Convai, cette fonctionnalité permet de construire une sorte d'arc narratif pour le personnage, similaire à un arbre de décisions. La construction du graphe s'effectue à l'aide de deux éléments principaux : les sections, colorées en noirs, et les déclencheurs, appelés « triggers » sur la plateforme, colorés en bleus.

Une section représente le but du personnage au cours de la discussion. Chaque section possède un titre, un identifiant et un objectif, et peut inclure une ou plusieurs décisions. L'objectif guide le PNJ sur ce qu'il doit dire à l'utilisateur durant cette étape, tandis que les décisions évaluent les réponses pour déterminer quand et quelle section suivante choisir. Ces éléments permettent ainsi de construire un scénario dans lequel progresser. Une section de démarrage doit être définie, et les sections de fins ne contiennent pas de décisions.

La figure 26 présente un aperçu d'une section de démarrage, indiquée par l'icône en forme d'étoile grisée. L'objectif précise en quelques mots ce que le PNJ doit faire durant cette étape. Dans cet exemple, il va accueillir l'utilisateur et lui demander s'il est le médecin venu pour réaliser l'examen de routine. La première décision qui figure dans l'illustration indique la prochaine section à suivre dans le cas où l'utilisateur est effectivement le docteur attendu.

Figure 26 : Exemple d'une section du Design narratif

The screenshot shows a web interface for narrative design. At the top, there's a header with a gear icon, the title 'Accueil', and a 'Section ID' (9e50b444-2d4a-11ef-bc59-42010a7be00e). Below the header, there are two main sections: 'Objective' and 'Decision 1'. The 'Objective' section contains two paragraphs of text. The 'Decision 1' section contains a single line of text and a 'Connects to' field with the value 'Etat du patient'. At the bottom right, there are 'Update' and 'Delete' buttons.

Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Un déclencheur, quant à lui, est un élément qui s'active lorsque certaines conditions sont remplies, permettant ainsi au personnage de réagir. Ces déclencheurs peuvent s'amorcer soit selon l'emplacement du PNJ ou du joueur dans l'environnement de jeu, soit si un certain temps s'est écoulé, soit si un évènement est survenu dans la scène. Cela permet ainsi de donner au personnage une dimension un peu plus réaliste. Un déclencheur est toujours suivi d'une section.

Chaque déclencheur possède un titre, un identifiant, un message indiquant l'évènement qui s'est produit, et la destination suivante. La figure 27 représente un exemple de déclencheur, qui s'active dans le cas où le joueur ne discute pas avec le personnage pendant un certain temps. Le message

indique au PNJ qu'il doit demander si tout va bien et s'il a besoin de préciser à nouveau quels sont les outils à disposition. La section suivante est ensuite précisée dans la destination. Un bouton « Invoke Trigger » permet de forcer l'activation du déclencheur lors d'une discussion effectuée sur la plateforme en ligne.

Figure 27 : Exemple d'un déclencheur du Design narratif

Absence de réponses

Trigger ID : 067ea7b8-2d4d-11ef-aa3e-42010a7be00e

Message

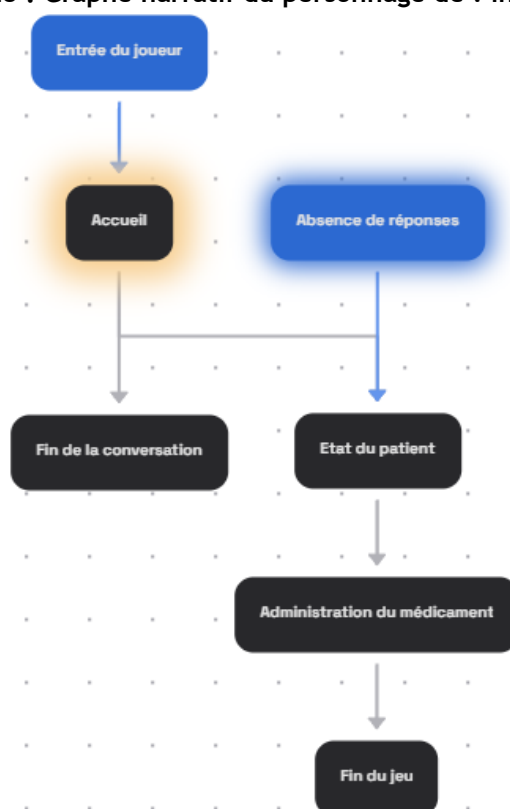
Demander au docteur si tout va bien, et s'il a besoin qu'on lui rappelle quels sont les c

Destination ID

Etat du patient

Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Figure 28 : Graphe narratif du personnage de l'infirmière



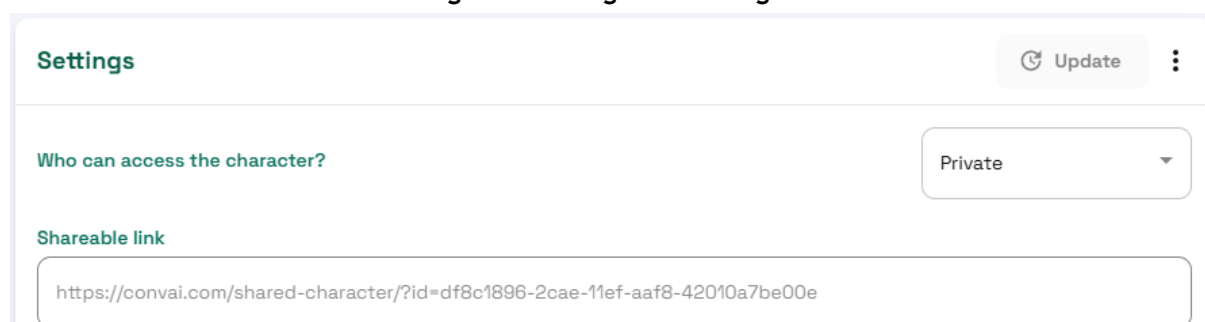
Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

Ces graphes narratifs sont très importants, car ils peuvent être importés sur Unity. Il peut donc être utile de construire un Design narratif plutôt complet pour faciliter l'intégration du scénario souhaité au personnage une fois en jeu. En figure 28 est illustré alors le graphique de l'infirmière, avec les grandes étapes de Sim3D représentées dans les sections, et certains déclencheurs pour activer des réactions au PNJ. La section de démarrage se prénomme « Accueil », et deux sections de fins sont présentes. Le premier concerne le cas où le joueur précise qu'il n'est pas médecin en début de jeu, ce qui met fin à la conversation. Tandis que le second se rapporte à la situation où les objectifs du scénario ont tous été accompli, mettant ainsi fin au jeu. Il est à noter cependant qu'il n'est pas possible de faire des boucles sur une même section, ce qui aurait été utile notamment pour celle de l'état du patient qui consiste à utiliser quatre outils, bien que différents, pour un même objectif.

3.1.11. PARAMÈTRES

Cette catégorie permet de déterminer si le personnage est public ou privé, afin que d'autres utilisateurs de Convai puissent tester la conversation avec lui. Bien qu'il soit configuré en privé dans mon cas, il est toujours possible de permettre à d'autres personnes d'échanger avec l'infirmière en utilisant le lien de partage disponible sur cette page de paramètres, visible dans la figure 29.

Figure 29 : Onglet « Settings »

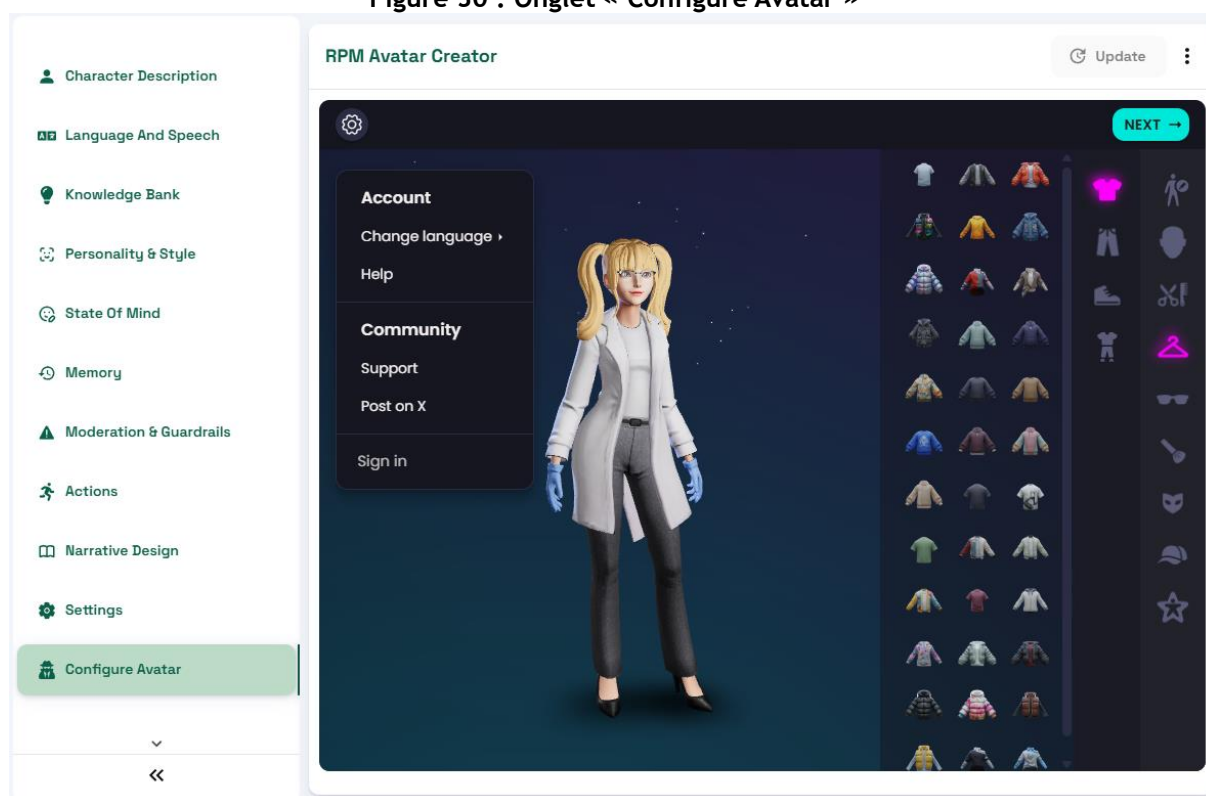


Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

3.1.12. CONFIGURATION DE L'AVATAR

Il est possible de modifier l'apparence du personnage à l'aide de cette dernière catégorie. Convai intègre les avatars Ready Player Me, ce qui signifie que la page de création du personnage, illustrée en figure 30, est la même que pour d'autres applications utilisant cet outil. Pour enregistrer l'avatar créé et pouvoir le réutiliser, que ce soit sur Convai ou ailleurs, il est nécessaire de se connecter avec un compte Ready Player Me. Dans mon cas, ayant auparavant conçu l'apparence de l'infirmière lors du développement de Sim3D, l'avatar est donc déjà présent sur mon compte personnel. Cependant, pour le récupérer, il est tout de même nécessaire de créer un premier avatar sur Convai afin que cette application puisse être associée à mon compte. Une fois cela fait, il est possible alors d'importer l'apparence de l'infirmière.

Figure 30 : Onglet « Configure Avatar »



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

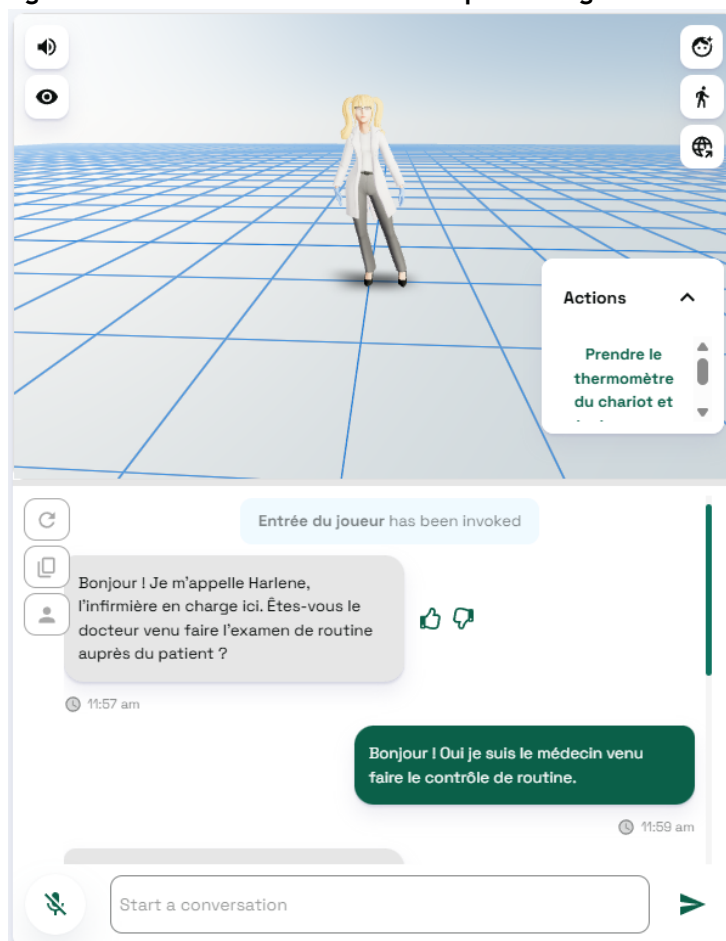
3.1.13. CONVERSATION AVEC LE PERSONNAGE

En plus de ces différentes catégories, une fenêtre de discussion avec le personnage est disponible en tout temps, afin de tester ses réponses et affiner sa configuration si besoin. Les échanges effectués sur cette fenêtre seront enregistrés dans la catégorie concernant la mémoire du PNJ. Comme illustré en figure 31, au-dessus de cette fenêtre de discussion est affiché l'apparence du personnage, et ce dernier réalise quelques animations selon les propos transmis à l'utilisateur. En plus d'être textuelles, les réponses du personnage se font aussi de manière vocale, avec la voix sélectionnée dans les premières catégories. Avant d'engager une conversation, Convai propose quelques suggestions pour démarrer la discussion, souvent en rapport avec la description du personnage, et la plupart du temps en anglais, même si ce dernier point n'empêche pas le PNJ de répondre en français.

Avec l'un des boutons « Invoke Trigger » du graphe narratif, un message apparaît dans la discussion, indiquant le titre du déclencheur avec la mention qu'il a été invoqué, comme il est possible de le voir en figure 31. Le personnage va répondre automatiquement en réaction à l'événement, de la même façon que si cela s'était déroulé en jeu. Le graphique de l'état d'esprit du personnage se met à jour pour indiquer ses émotions correspondant à sa dernière réponse. Enfin, si l'option des actions activées, l'utilisateur peut commander au personnage de réaliser celles qui

ont été spécifiées au préalable. De plus, il est aussi possible de voir que certaines actions sont suggérées au fil de la conversation, apparaissant dans une boîte textuelle blanches dénommées « Actions », visible également en figure 31.

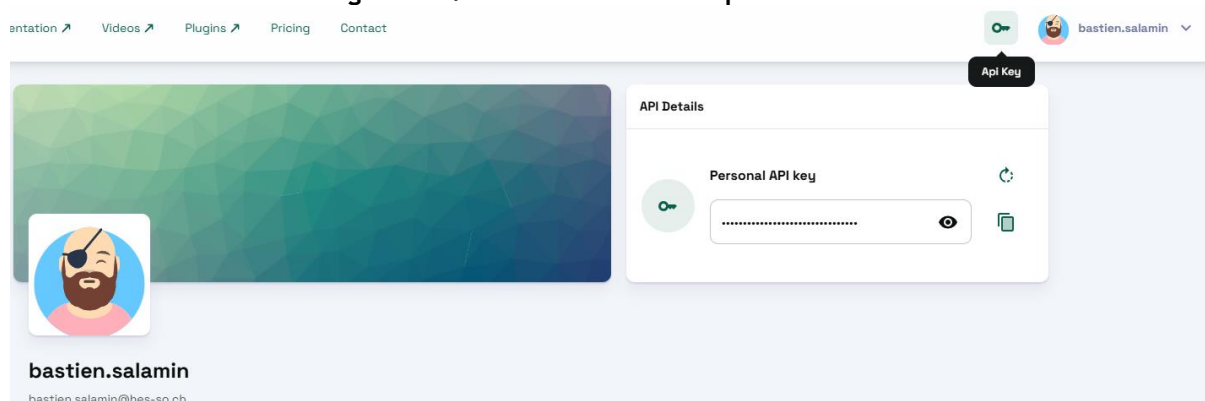
Figure 31 : Fenêtre de discussion du personnage sur Convai



Source : <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>

3.1.14. COMPTE

Figure 32 : Informations du compte utilisateur

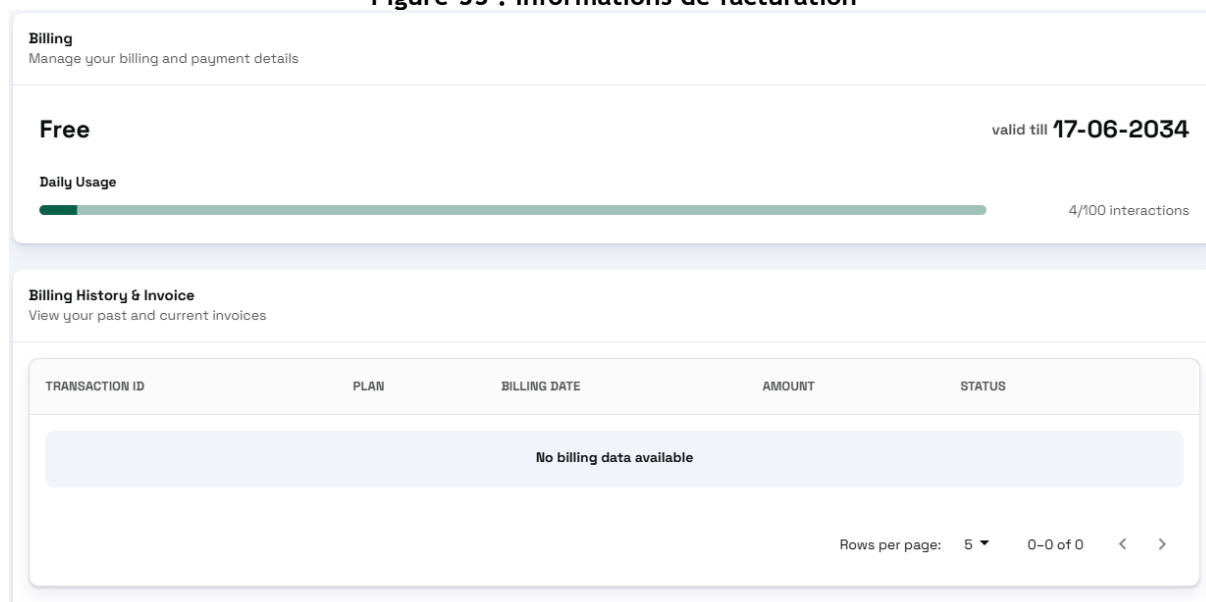


Source : <https://convai.com/management/profile>

À tout moment, cliquer sur le nom de l'utilisateur permet d'accéder aux informations du compte ou de facturation. Concernant le compte, on retrouve notamment la clé d'interface de programmation d'application, nommée « Application Programming Interface » (API), nécessaire pour activer les fonctionnalités de Convai sur Unity. Cette suite de caractères est aussi accessible à tout moment en cliquant sur l'icône en forme de clé, se trouvant à gauche du nom d'utilisateur sur l'interface de Convai, tel qu'indiqué en figure 32.

Les informations de facturation offrent, en plus d'un historique des paiements effectués sur la plateforme, la possibilité de suivre le quota journalier pour chaque interaction avec un personnage de Convai, que ce soit sur la plateforme ou sur Unity directement, comme illustré à la figure 33. Lors de la création de mon compte, il n'a pas été nécessaire d'entrer les informations d'une carte bancaire. Convai s'utilise directement après une inscription avec un courriel et un mot de passe.

Figure 33 : Informations de facturation



Source : <https://convai.com/management/billing>

3.2. PRÉPARATION DU PROJET SIM3D

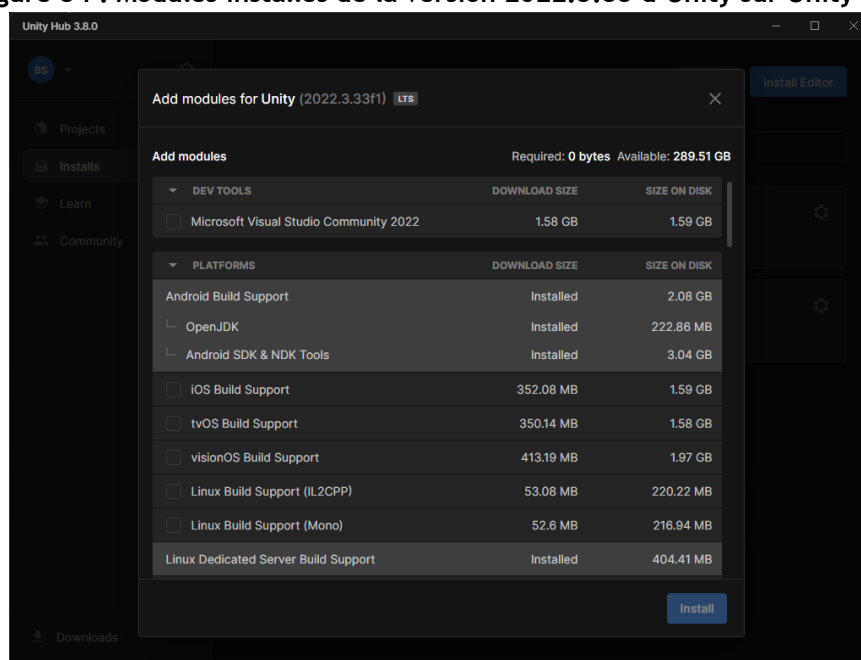
Après avoir passé en revue les différentes fonctionnalités de la plateforme en ligne de Convai et construit l'infirmière, l'intégration du personnage intelligent sur Unity est la prochaine étape à réaliser. Pour ce faire, il est nécessaire de mettre à jour la version du projet Sim3D et d'installer les ressources de Convai disponibles sur l'Unity Asset Store.

3.2.1. MISE À JOUR ET IMPORTATION DES RESSOURCES

Pour commencer, j'ai téléchargé à l'aide du programme « Unity Hub », ce dernier permettant la gestion de différentes installations d'Unity, la version 2022.3.33, la plus récente lors de l'écriture de ce rapport. Avant le lancement du téléchargement, il est nécessaire de cocher les mêmes

modules que ceux présents dans la version d'Unity employée par Sim3D, à savoir « Android Build Support », « Android SDK & NDK Tools », « OpenJDK » et « Linux Dedicated Server Build Support », comme le montre la figure 34. Cela permet ainsi d'éviter une première perte de fonctionnalité. Sur cette image, il est à noter que l'outil de développement « Visual Studio 2022 » n'a pas été installé, ayant déjà sur ma machine la version 2019 utilisée durant le projet Sim3D.

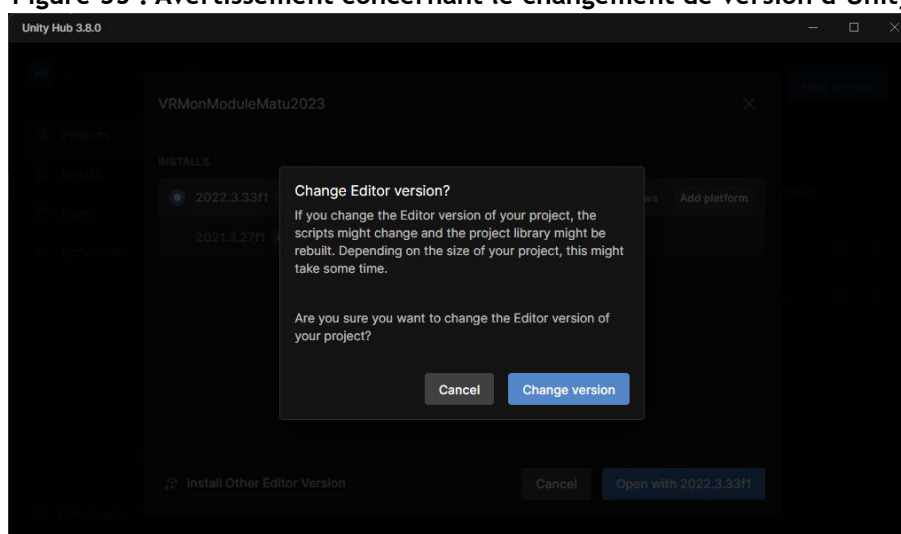
Figure 34 : Modules installés de la version 2022.3.33 d'Unity sur Unity Hub



Source : données de l'auteur

Il est possible ensuite d'assigner cette nouvelle version d'Unity au projet Sim3D. Une fenêtre avertit alors que des changements peuvent s'effectuer au niveau de certains scripts, et que l'ouverture peut prendre du temps, comme l'indique la figure 35.

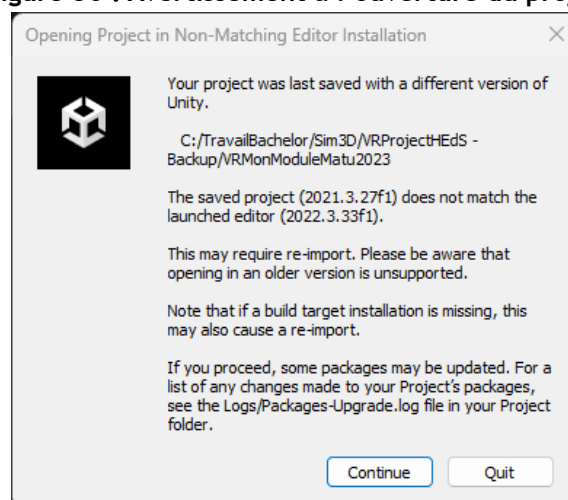
Figure 35 : Avertissement concernant le changement de version d'Unity



Source : données de l'auteur

À l'ouverture du projet avec cette nouvelle installation d'Unity, un second avertissement rappelle que les versions ne correspondent pas, et que cela nécessite un nouveau chargement des ressources, comme présenté dans la figure 36. Pour les besoins du prototype, il est nécessaire de continuer.

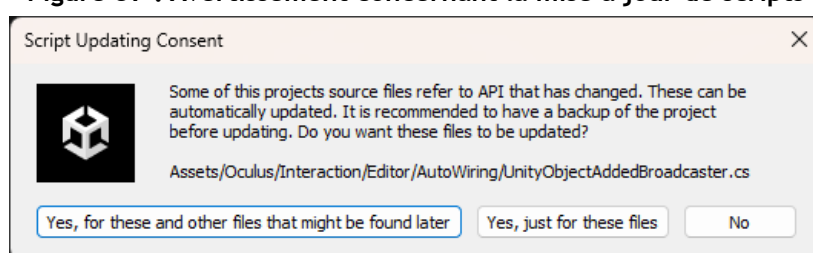
Figure 36 : Avertissement à l'ouverture du projet



Source : données de l'auteur

Durant le processus d'ouverture du projet, un nouvel avertissement, illustré en figure 37, demande un consentement avant de mettre à jour un fichier fourni par Unity et faisant appel à des APIs. Vu que cette opération semble nécessaire pour pouvoir ouvrir Sim3D avec la version 2022.3.33 d'Unity, j'ai choisi la première option, qui consiste à accepter la mise à jour pour ce fichier ainsi que pour tous les autres qui pourraient survenir au cours du processus.

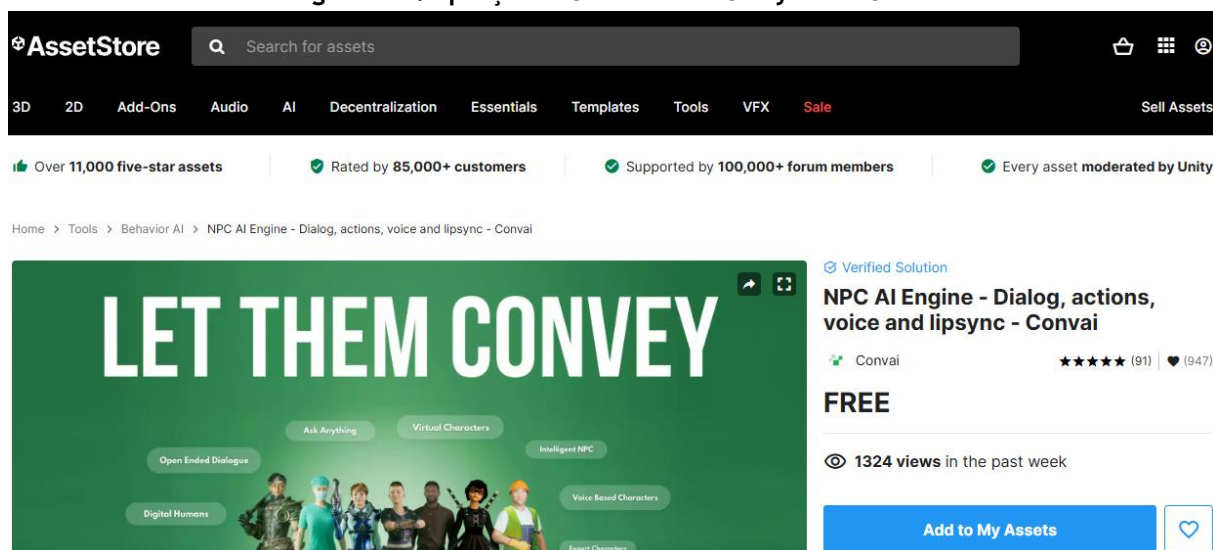
Figure 37 : Avertissement concernant la mise à jour de scripts



Source : données de l'auteur

Après une durée approximative d'une demi-heure, le projet Sim3D s'ouvre enfin, sans erreur particulière, que ce soit sur la scène principale du jeu ou dans la console de messages d'Unity. Il est alors possible d'installer Convai sans risque d'incompatibilité au niveau des versions. Pour ce faire, il faut se connecter à l'Unity Asset Store avec le même compte utilisateur employé sur l'Unity Hub. Sur la page Convai, il suffit ensuite de cliquer sur le bouton bleu « Add to My Assets », visible en figure 38, afin que les ressources à installer puissent être directement accessibles depuis l'éditeur Unity.

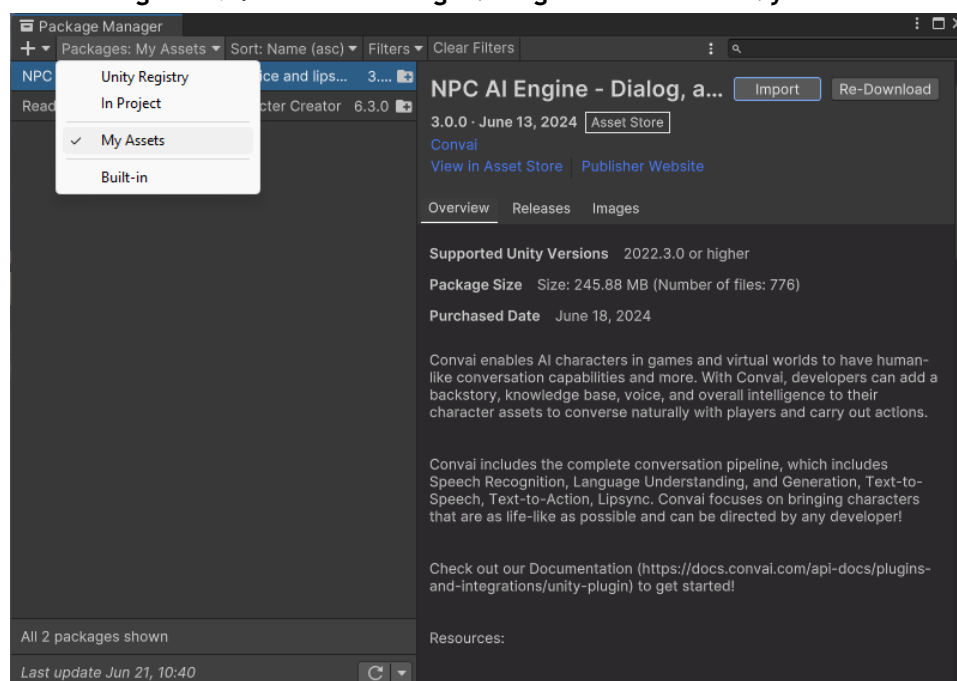
Figure 38 : Aperçu de Convai sur l'Unity Asset Store



Source : <https://assetstore.unity.com/packages/tools/behavior-ai/npc-ai-engine-dialog-actions-voice-and-lipsync-convai-235621>

Sur l'éditeur Unity, en ouvrant la fenêtre « Package Manager », il est nécessaire de filtrer les ressources selon « My Assets », comme le montre la figure 39. Ainsi, Convai apparaît et il est possible de le télécharger, puis de l'importer au projet.

Figure 39 : Fenêtre Package Manager avec le filtre My Assets



Source : données de l'auteur

Cependant, avant de pouvoir effectuer ce processus, Unity avertit que certaines dépendances nécessitent d'être mises à jour ou d'être installées, la liste étant illustrée en figure 40. Il faut alors accepter pour pouvoir procéder ensuite à l'importation de Convai.

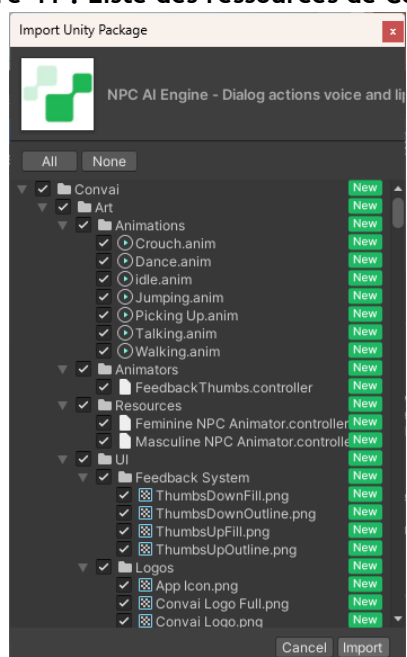
Figure 40 : Liste des dépendances nécessitant une mise à jour pour Convai



Source : données de l'auteur

Après cette installation s'affiche une fenêtre, visible en figure 41, montrant la liste des ressources de Convai qu'il est possible d'installer. De base, elles sont toutes cochées, alors il suffit juste d'importer.

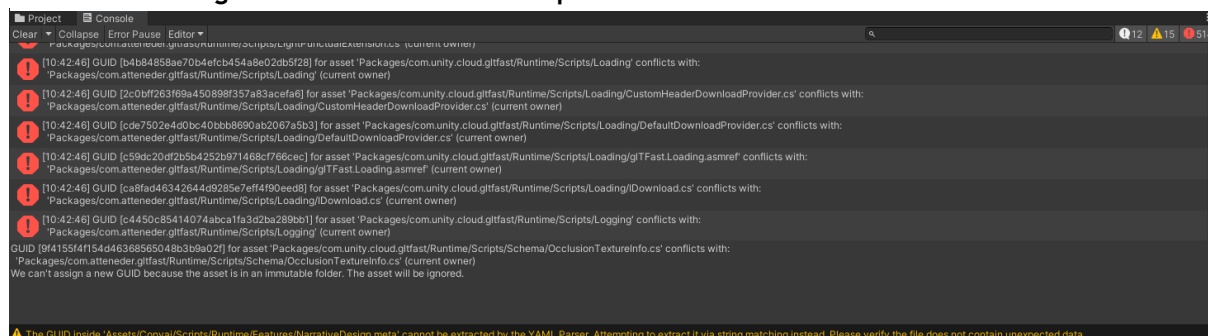
Figure 41 : Liste des ressources de Convai



Source : données de l'auteur

Cependant, un nombre élevé de conflits apparaissent suite à l'installation de Convai, comme illustré en figure 42.

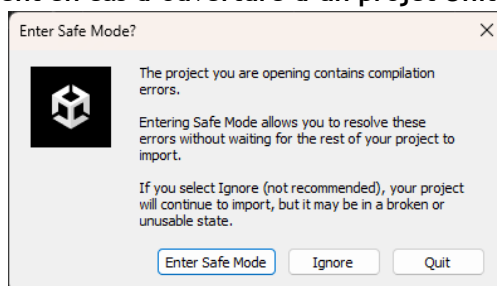
Figure 42 : Liste de conflits après l'installation de Convai sur Sim3D



Source : données de l'auteur

J'ai trouvé alors judicieux de fermer l'éditeur et d'ouvrir à nouveau le projet Sim3D, toujours avec la version 2022.3.33 d'Unity. Comme il est coutume en cas d'erreurs, Unity demande d'ouvrir le projet en mode sans échec, la fenêtre d'avertissement étant visible en figure 43.

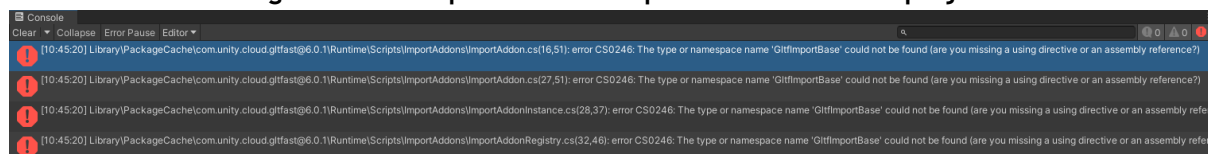
Figure 43 : Avertissement en cas d'ouverture d'un projet Unity contenant des conflits



Source : données de l'auteur

Avec cette ouverture en mode sans échec, la figure 44 montre que la console de messages d'Unity n'affiche désormais plus que quatre erreurs.

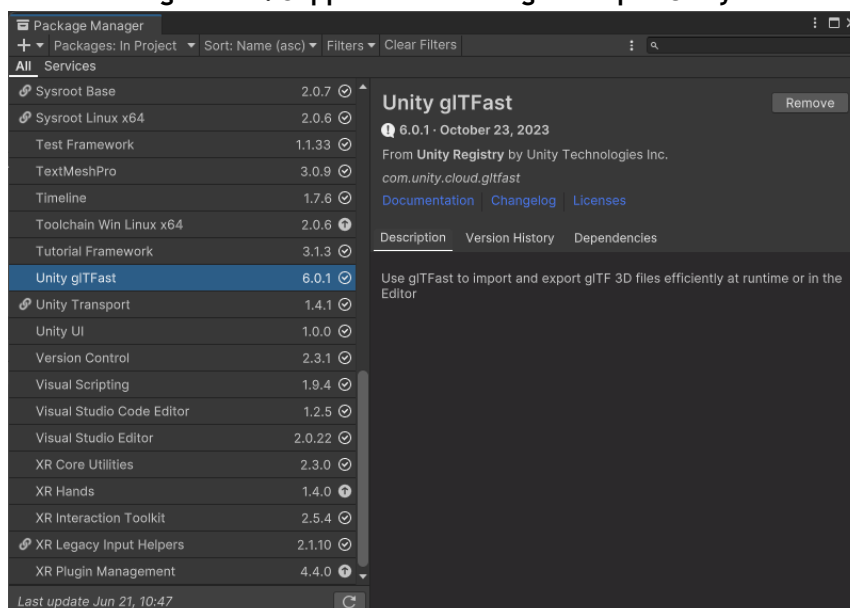
Figure 44 : Les quatre conflits après réouverture du projet



Source : données de l'auteur

Ces conflits indiquent que l'éditeur ne sait pas quelle référence de « GltfImportBase » utiliser. En effet, avec l'installation de la version 2022.3.33 d'Unity, cet élément fait partie d'une ressource nommée « glTFast », qui a été prise en charge officiellement par Unity depuis le 23 octobre 2023, comme indiqué en figure 45. Ce conflit apparaît car, lors de la réalisation du projet Sim3D, il a été nécessaire d'importer cette ressource manuellement, signifiant qu'à ce stade il y a deux instances de glTFast.

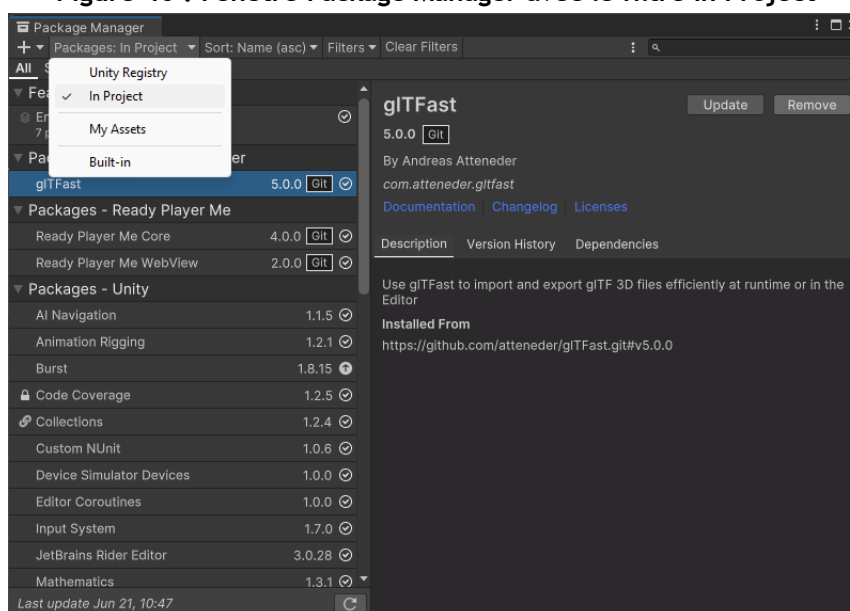
Figure 45 : Support officiel de glTFast par Unity



Source : données de l'auteur

Pour pouvoir résoudre ce conflit, il est alors nécessaire de désinstaller la ressource installée pendant le développement de Sim3D. Dans le Package Manager, il faut filtrer par « In Project », et enlever le glTFast présenté en figure 46.

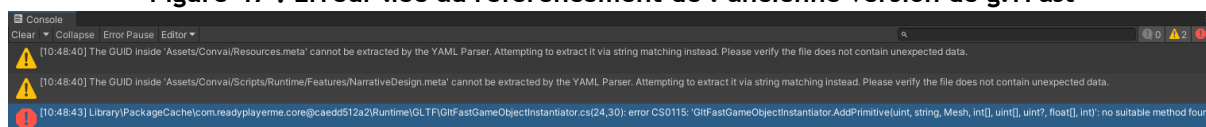
Figure 46 : Fenêtre Package Manager avec le filtre In Project



Source : données de l'auteur

Après cette première correction, une seule nouvelle erreur s'affiche dans la console Unity, comme illustré en figure 47. Ce conflit est en lien avec des ressources de Ready Player Me, installées elles aussi durant le développement de Sim3D, et qui font référence au glTFast tout juste supprimé.

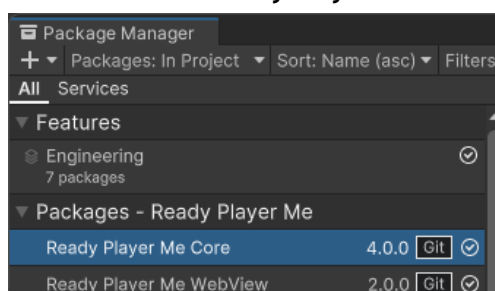
Figure 47 : Erreur liée au référencement de l'ancienne version de glTFast



Source : données de l'auteur

Afin de corriger cette erreur, il est nécessaire de procéder à la mise à jour de Ready Player Me. La première chose à faire est donc de supprimer les deux ressources Ready Player Me installées lors du développement de Sim3D, « Core et « WebView », illustrées en figure 48.

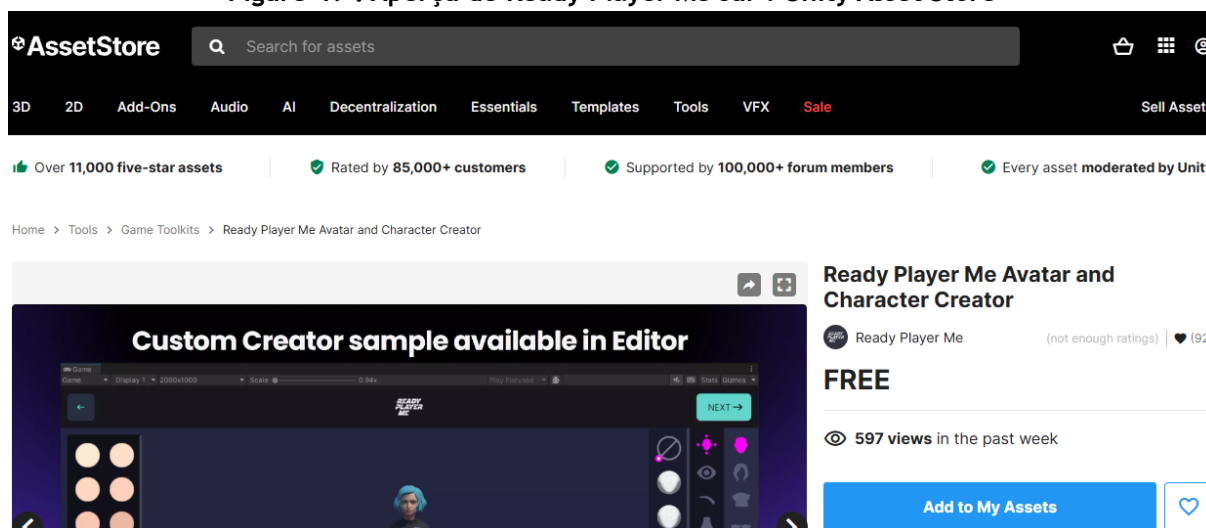
Figure 48 : Ressources Ready Player Me à désinstaller



Source : données de l'auteur

Ce processus va générer plusieurs erreurs qui vont s'afficher sur la console de messages d'Unity. Cela est normal, et pour les corriger, il faut procéder aux mêmes étapes qu'à l'installation de Convai. C'est-à-dire qu'il faut commencer par ajouter Ready Player Me à notre compte depuis l'Unity Asset Store, visible en figure 49, pour pouvoir le télécharger puis l'installer.

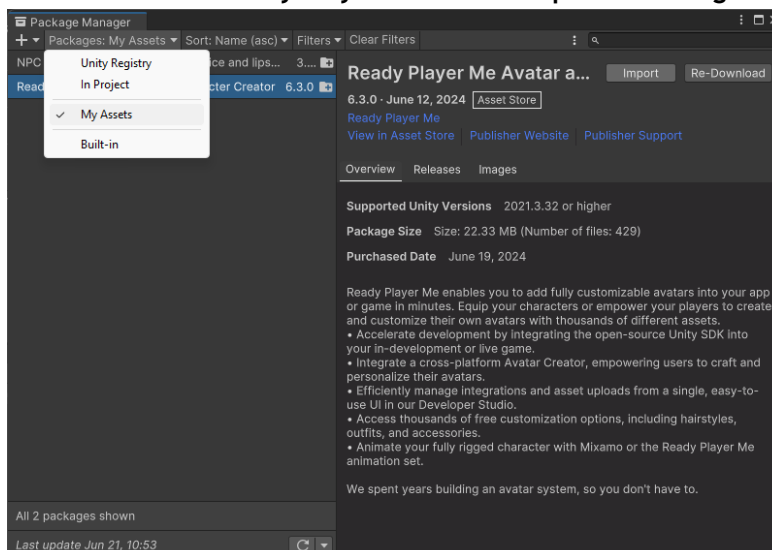
Figure 49 : Aperçu de Ready Player Me sur l'Unity Asset Store



Source : <https://assetstore.unity.com/packages/tools/game-toolkits/ready-player-me-avatar-and-character-creator-259814>

Ainsi, il est possible d'importer la dernière version de Ready Player Me, avec le même procédé que pour Convai, à savoir depuis le Package Manager sous My Assets, comme le montre la figure 50.

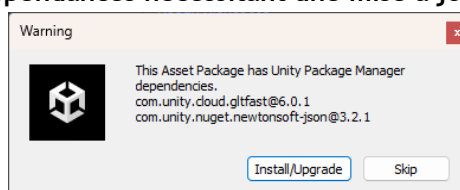
Figure 50 : Ressource Ready Player Me visible depuis le Package Manager



Source : données de l'auteur

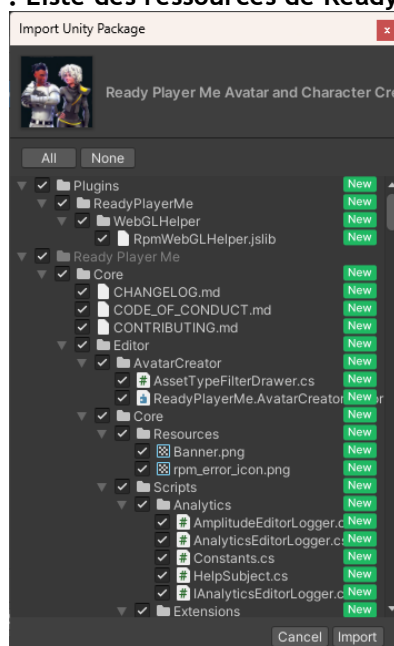
Tout comme Convai, avant de pouvoir importer Ready Player Me à proprement parler, deux dépendances nécessitent une mise à jour ou une installation qu'il faut accepter, comme illustré en figure 51.

Figure 51 : Liste des dépendances nécessitant une mise à jour pour Ready Player Me



Source : données de l'auteur

Figure 52 : Liste des ressources de Ready Player Me

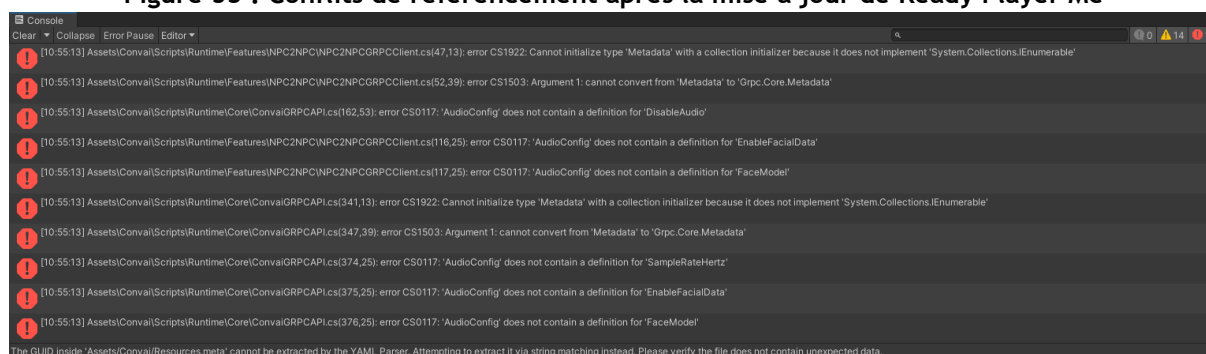


Source : données de l'auteur

Ensuite, il est possible d'importer Ready Player Me au projet. De la même manière que pour Convai, la figure 52 montre que toutes les ressources disponibles sont sélectionnées par défaut, il suffit alors de tout importer

Après cette correction, de nouvelles erreurs font leur apparition sur la console de messages d'Unity, illustré en figure 53. Cette fois, il s'agit simplement de mauvais référencements. Il faut donc éditer les deux classes d'objets de Convai qui ressortent en erreur : « NPC2NPCGRPCClient » et « ConvaiGRPCAPI ».

Figure 53 : Conflits de référencement après la mise à jour de Ready Player Me

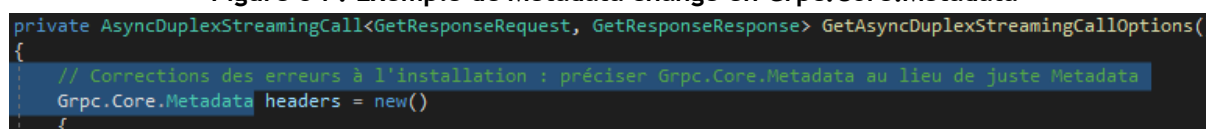


Source : données de l'auteur

Pour modifier ces deux classes d'objets, il faut les ouvrir sur Visual Studio, et effectuer deux changements mineurs en fonction de l'erreur rencontrée. En effet, NPC2NPCGRPCClient et ConvaiGRPCAPI ont tous deux des référencements construits de la même manière : le premier étant pour « Metadata », le second vers « AudioConfig ». Cependant, les erreurs qui surviennent sur la console indiquent qu'Unity ne sait pas à quel Metadata ni à quel AudioConfig ces classes font références. Il est donc nécessaire de les préciser.

Concernant l'élément Metadata, il faut le modifier pour « Grpc.Core.Metadata », comme l'illustre la figure 54. J'ai effectué ce changement à chaque fois qu'un Metadata surligné en rouge apparaissait dans l'une des deux classes, avec un commentaire précisant la modification réalisée.

Figure 54 : Exemple de Metadata changé en Grpc.Core.Metadata



Source : données de l'auteur

Pour ce qui est de l'élément AudioConfig, il s'agit du même principe. À chaque endroit parmi ces deux classes où cet élément était surligné en rouge, j'ai modifié le code défectueux afin de transformer les instantiations AudioConfig en « Service.AudioConfig », la figure 55 étant un exemple de ce changement. Comme pour Metadata, j'ai de plus rajouté un commentaire pour préciser la modification réalisée.

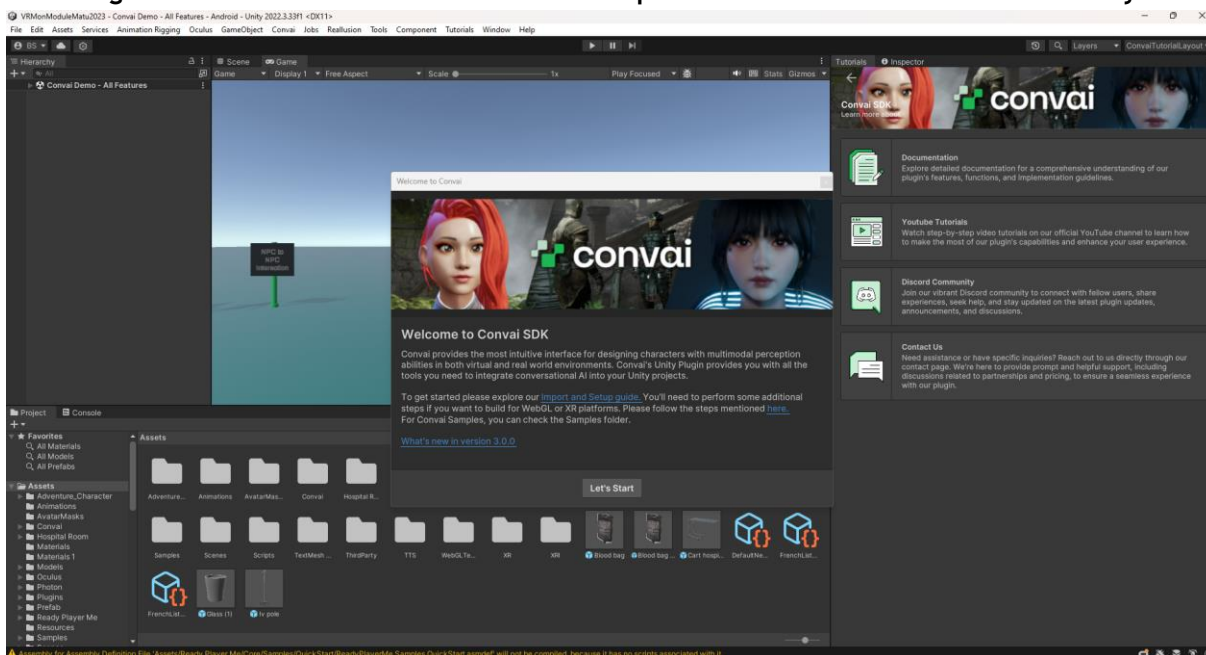
Figure 55 : Exemple d'AudioConfig changé en Service.AudioConfig

```
GetResponseConfig = new GetResponseRequest.Types.GetResponseConfig
{
    CharacterId = characterID,
    ApiKey = _apiKey,
    SessionId = sessionID,
    AudioConfig = new Service.AudioConfig // Corrections des erreurs à l'installation : préciser Service.AudioConfig au lieu de juste AudioConfig
}
```

Source : données de l'auteur

Une fois que les classes d'objets NPC2NPCGRPCClient et ConvaiGRPCAPI ne montrent plus d'erreurs et que les changements ont été enregistrés, l'éditeur d'Unity se met à jour et sort du mode sans échec. Ainsi, les conflits ont été corrigés et aucune erreur supplémentaire ne s'affiche dans la console de messages. Une scène de démonstration proposée par Convai s'ouvre automatiquement, avec une fenêtre souhaitant la bienvenue à l'utilisateur pour démarrer l'utilisation de ce nouvel outil, comme illustré en figure 56. Il est désormais possible d'intégrer les fonctionnalités de Convai au projet Sim3D.

Figure 56 : Fenêtre de bienvenue lors de la première utilisation de Convai sur Unity

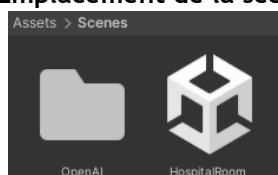


Source : données de l'auteur

3.2.2. CORRECTIONS DES ANIMATIONS DE LA SCÈNE SIM3D

La mise à jour de Ready Player Me a provoqué une perte de références au niveau des animations des personnages de la scène de jeu de Sim3D. Cette scène, appelée « HospitalRoom », est enregistrée dans le projet sous « Assets > Scenes », comme indiqué à la figure 57.

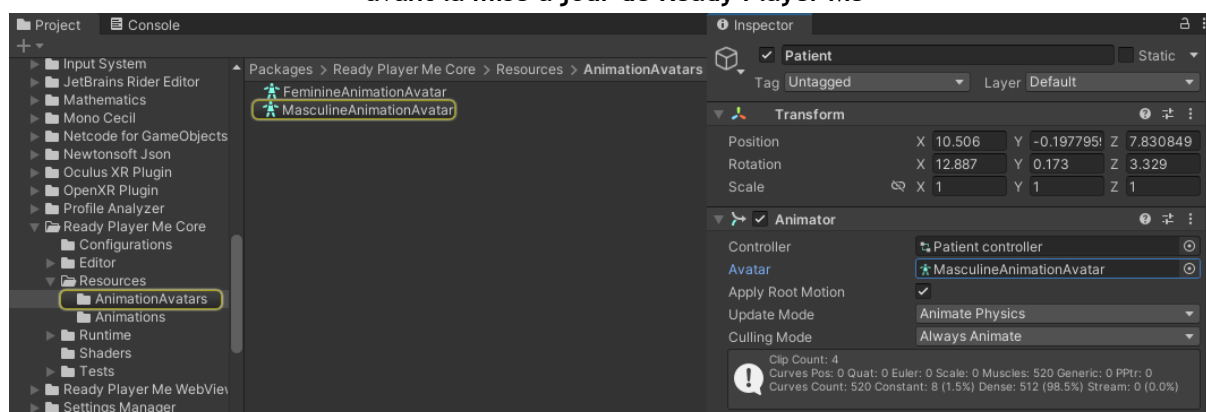
Figure 57 : Emplacement de la scène de Sim3D



Source : données de l'auteur

En effet, lorsque cette scène est exécutée en mode « Play » pour tester le jeu, les personnages restent figés sur place et ne s'animent pas. Pour corriger cela, il faut modifier les GameObjects de chaque avatar présent dans la scène. Chaque GameObject d'un personnage possède un composant « Animator », et ce dernier contient un élément « Avatar » ayant un « Missing (Avatar) » après la mise à jour de Ready Player Me. Auparavant, les avatars de la scène possédaient à cet emplacement un « MasculineAnimationAvatar » ou un « FeminineAnimationAvatar », la figure 58 représentant le cas du personnage du patient.

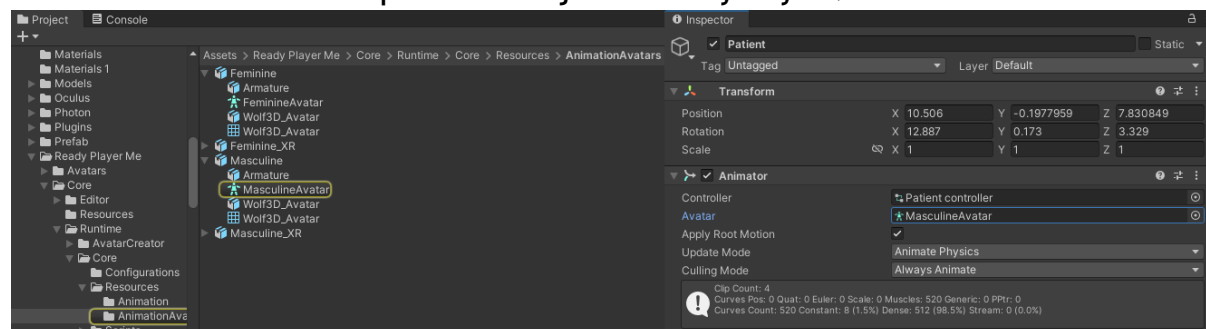
Figure 58 : Emplacement de l'élément Avatar du GameObject Patient avant la mise à jour de Ready Player Me



Source : données de l'auteur

Mais ces éléments ont été déplacés et renommés après l'installation de la dernière version de Ready Player Me. En utilisant la recherche d'éléments de l'onglet « Projet », le dossier « AnimationAvatars », qui se trouvait auparavant sous « Packages > Ready Player Me Core > Resources », a été déplacé vers « Assets > Ready Player Me > Core > Runtime > Core > Resources ». Aussi, les objets Avatar recherchés sont maintenant enregistrés dans des GameObjects préconfigurés que l'on appelle « Prefab » dans Unity, et renommés en « MasculineAvatar » et « FeminineAvatar ». En remplaçant les « Missing (Avatar) » des composants Animator de chaque personnage par ces éléments, comme celui du patient illustré en figure 59, les personnages exécutent de nouveau les animations qui leur ont été assignées pour Sim3D.

Figure 59 : Emplacement de l'élément Avatar du GameObject Patient après la mise à jour de Ready Player Me



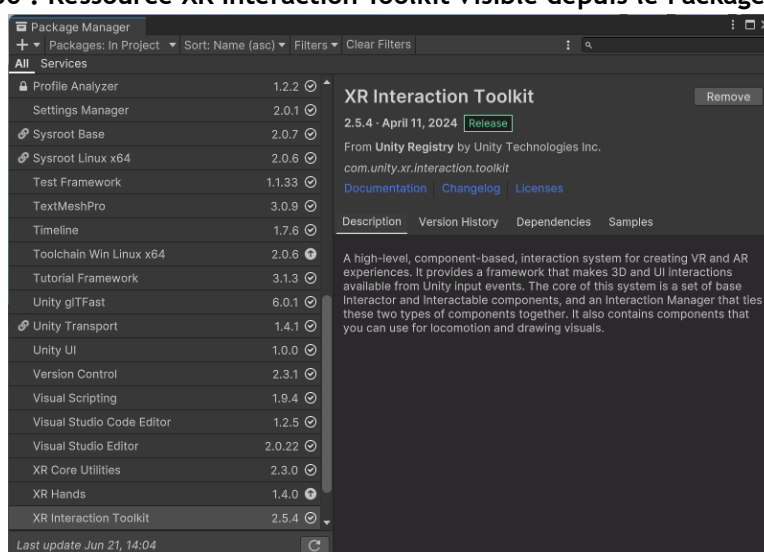
Source : données de l'auteur

3.2.3. XR DEVICE SIMULATOR

Comme le prototype de ce travail de Bachelor se concentre sur le développement du PNJ, je n'avais donc pas à disposition un casque de réalité virtuelle pour tester les actions du joueur du jeu Sim3D. Heureusement, Unity met à disposition une ressource permettant de simuler les mouvements du casque et ses manettes, nommée « XR Device Simulator ».

Pour cela, il faut à nouveau aller sur la fenêtre du Package Manager, et filtrer les ressources par In Project pour trouver celle nommée « XR Interaction Toolkit », comme illustré en figure 60.

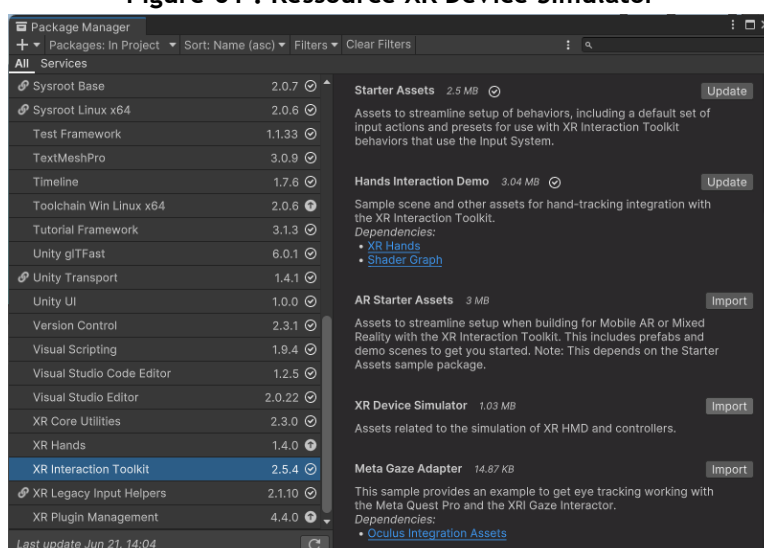
Figure 60 : Ressource XR Interaction Toolkit visible depuis le Package Manager



Source : données de l'auteur

XR Device Simulator se trouve au fond de la liste d'éléments de l'onglet « Samples », comme l'illustre la figure 61. Il ne reste plus qu'à l'importer au projet.

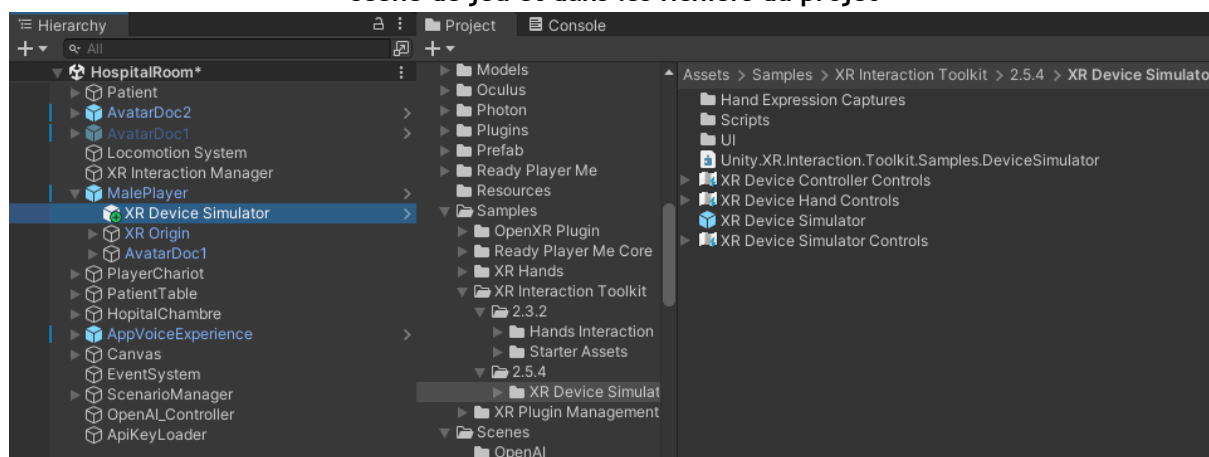
Figure 61 : Ressource XR Device Simulator



Source : données de l'auteur

Puis, il convient d'ajouter l'élément importé dans la scène de Sim3D. La figure 62 indique que le Prefab XR Device Simulator est ajouté au GameObject du joueur, « MalePlayer ». C'est aussi à cet emplacement que l'on retrouve l'élément intégrant les mouvements du casque de réalité virtuelle, le GameObject « XR Origin », et l'avatar du joueur sous « AvatarDoc1 ». Ainsi, la scène de Sim3D est correctement mise à jour sans avoir perdu ses fonctionnalités.

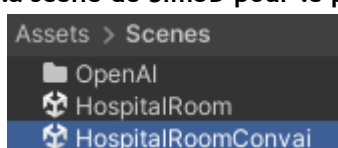
Figure 62 : Emplacement du Prefab XR Device Simulator dans la scène de jeu et dans les fichiers du projet



Source : données de l'auteur

Pour ne pas perdre la scène originale de Sim3D avec les changements qui pourraient survenir au cours de l'implémentation du prototype, il convient de la copier. Comme l'illustre la figure 63, j'ai choisi de nommer la scène intégrant le personnage Convai « HospitalRoomConvai », et de la laisser dans le même dossier où est enregistré celle de Sim3D.

Figure 63 : Copie de la scène de Sim3D pour le prototype du Bachelor



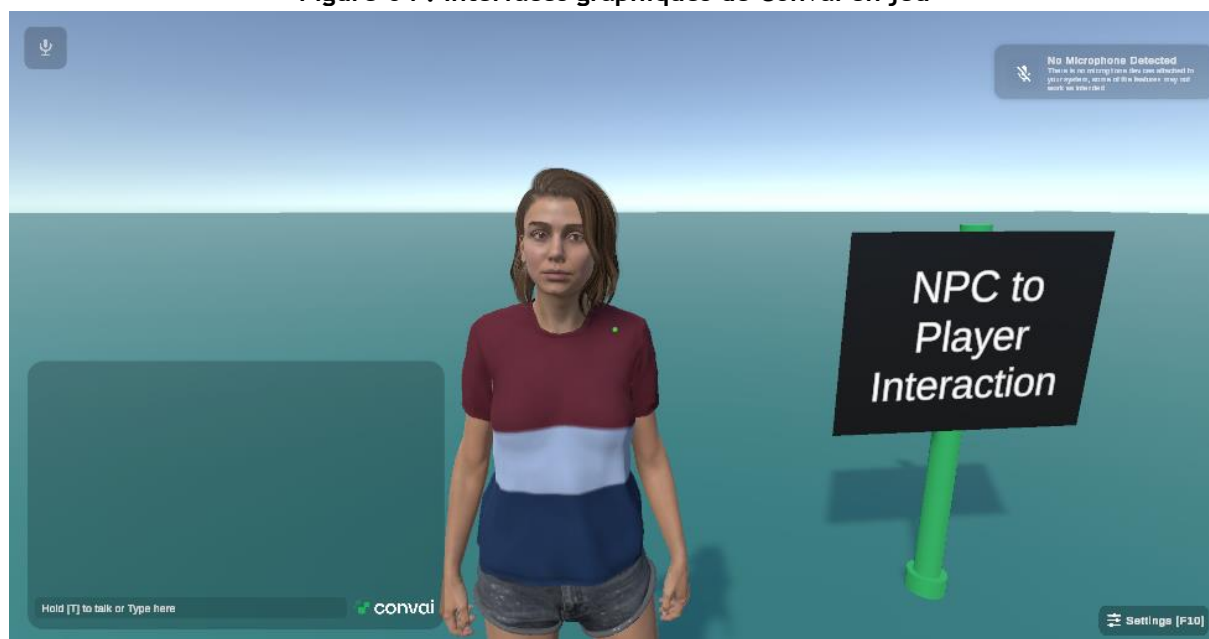
Source : données de l'auteur

3.2.4. RÉIMPORTATION DES RÉFÉRENCES TEXTMESH PRO

L'utilisation des scènes de démonstration fournies par Convai a montré que certaines références d'Unity n'étaient pas correctement actualisées après la mise à jour du moteur de jeu. Convai utilise les ressources de « TextMesh Pro » qui permet d'afficher du texte et, dans ce cas précis, d'afficher la fenêtre de conversation entre le PNJ intelligent et l'utilisateur, ainsi que d'autres interfaces graphiques utilitaires visibles en figure 64.

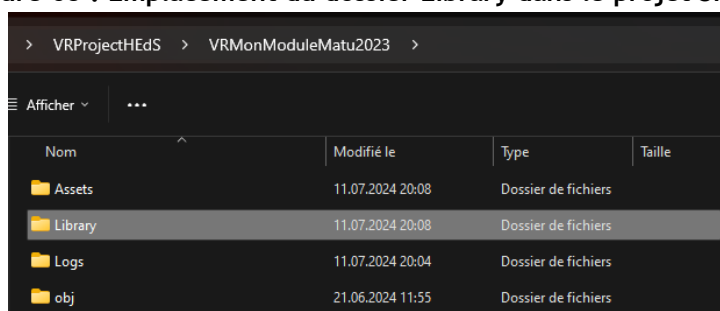
Afin de corriger ces référencements, selon le commentaire de l'utilisateur « lucbloom » (2020) sur le forum Unity, il faut supprimer le dossier « Library » du projet Unity, la figure 65 montrant l'emplacement de cet élément pour Sim3D dans l'explorateur de fichiers Windows.

Figure 64 : Interfaces graphiques de Convai en jeu



Source : données de l'auteur

Figure 65 : Emplacement du dossier Library dans le projet Sim3D



Source : données de l'auteur

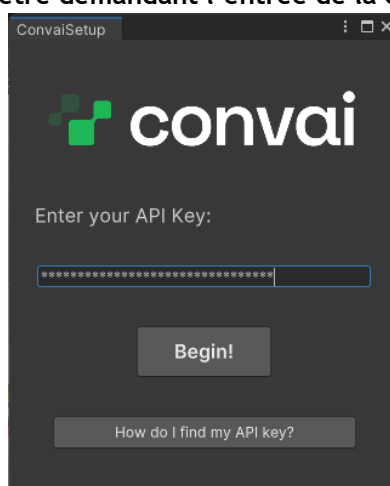
En ouvrant à nouveau le projet depuis l'Unity Hub, les référencements se remettront à jour pendant un certain temps, de la même façon que lorsque le projet a été ouvert pour la première fois avec la version 2022.3.33 d'Unity. Une fois le processus terminé, j'ai dû importer à nouveau les ressources de TextMesh Pro pour les mettre à jour, en allant depuis la barre de menu d'Unity, sous « Window > TextMeshPro > Import TMP Essential Resources ». Ainsi, les interfaces graphiques implémentées par Convai s'affichent correctement, et il est alors possible d'échanger avec les personnages.

3.3. DÉVELOPPEMENT DU PNJ AVEC CONVAI

Avant toute chose, il est nécessaire d'ajouter sur l'éditeur Unity la clé API de son compte Convai pour débloquer les fonctionnalités de ce dernier, dont notamment la génération des réponses du personnage. Pour ce faire, depuis la barre de menu d'Unity, il faut aller sous « Convai > Convai Setup ». Une fenêtre s'affiche, présentée en figure 66, demandant à l'utilisateur de rentrer sa clé

API. Comme mentionné au point 3.1.14, cette dernière se trouve sur la plateforme en ligne de Convai, après s’y être connecté avec son compte. Il suffit de copier cette suite de caractères et de l’insérer dans la zone de texte prévu à cet effet, puis de cliquer sur le bouton « Begin! ». Une fenêtre s’affiche ensuite, indiquant que la clé API a été chargée avec succès.

Figure 66 : Fenêtre demandant l’entrée de la clé API de Convai

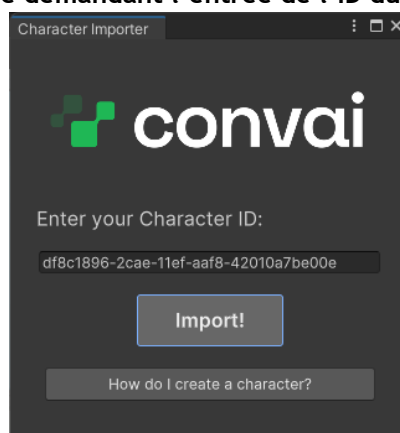


Source : données de l’auteur

3.3.1. IMPORTATION DU PERSONNAGE CONVAI

Pour importer l’infirmière créée sur la plateforme de Convai dans la scène HospitalRoomConvai, il faut aller dans la barre de menu Unity sous « Convai > Character Importer ». De la même manière que pour rentrer la clé API, une fenêtre s’affiche, demandant d’insérer l’identifiant du personnage afin de pouvoir l’importer, comme illustré en figure 67. Pour rappel, cet ID est automatiquement attribué par Convai lorsque l’on crée un PNJ sur leur plateforme, et il se trouve dans la partie concernant la description du personnage, comme expliqué au point 3.1.2 de ce document.

Figure 67 : Fenêtre demandant l’entrée de l’ID du personnage Convai



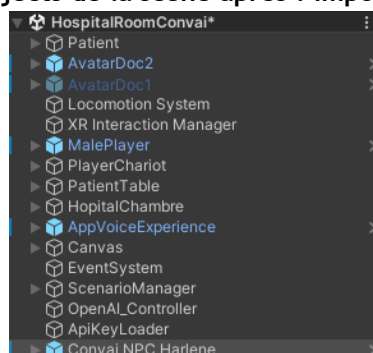
Source : données de l’auteur

Un Prefab du personnage de l’infirmière est automatiquement créé et ajouté dans la scène de jeu. Il est nommé « Convai NPC Harlene ». Mais comme il s’agit d’un élément qui va être modifié, il

convient de « déballer » ce GameObject en faisant un clic-droit dessus, puis « Prefab > Unpack Completely ». C'est une méthode qui sera utilisée aussi pour les autres Prefabs qui seront employés au cours de ce développement.

Il convient aussi de supprimer l'avatar initial de l'infirmière qui a été ajoutée durant la réalisation de Sim3D, afin de ne garder que le modèle de Convai. Cet avatar est enregistré sous le GameObject « AvatarDoc2 ». De plus, un AvatarDoc1 est aussi présent mais inactif, qui était censé être l'équivalent masculin de l'infirmière au cas où l'option de choix du sexe du joueur aurait été implémentée. Il convient de le supprimer de la scène aussi. La figure 68 montre ces deux GameObjects à enlever, ainsi que le Prefab de l'infirmière Convai après son importation.

Figure 68 : Liste de GameObjects de la scène après l'importation du personnage Convai



Source : données de l'auteur

En repositionnant le personnage Convai au même emplacement où se tenait l'infirmière de Sim3D, et en lançant le mode Play de la scène de jeu, il est possible de constater que le PNJ suit déjà automatiquement le joueur des yeux et se tourne vers lui, comme l'illustre la figure 69.

Figure 69 : L'infirmière suivant le joueur du regard

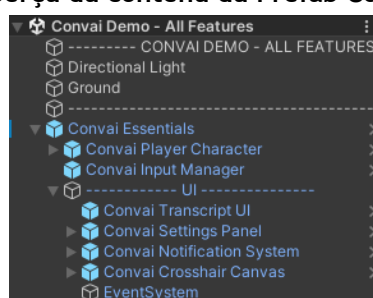


Source : données de l'auteur

3.3.2. INTÉGRATION DE L'INTERFACE DE CONVAI

Convai possède un Prefab qui couvre tous les éléments essentiels pour faire fonctionner l'échange avec un de leur personnage. Ce GameObject, nommé « Convai Essentials », se trouve dans « Assets > Convai > Prefabs > Base Scene Essentials ». Le contenu de ce Prefab est illustré en figure 70. Il est donc important de le rajouter dans la scène du prototype. Cependant, il n'est pas nécessaire de conserver tous ses éléments, étant donné que les fonctionnalités de quelques-uns sont déjà présentes d'une certaine manière dans la scène de Sim3D.

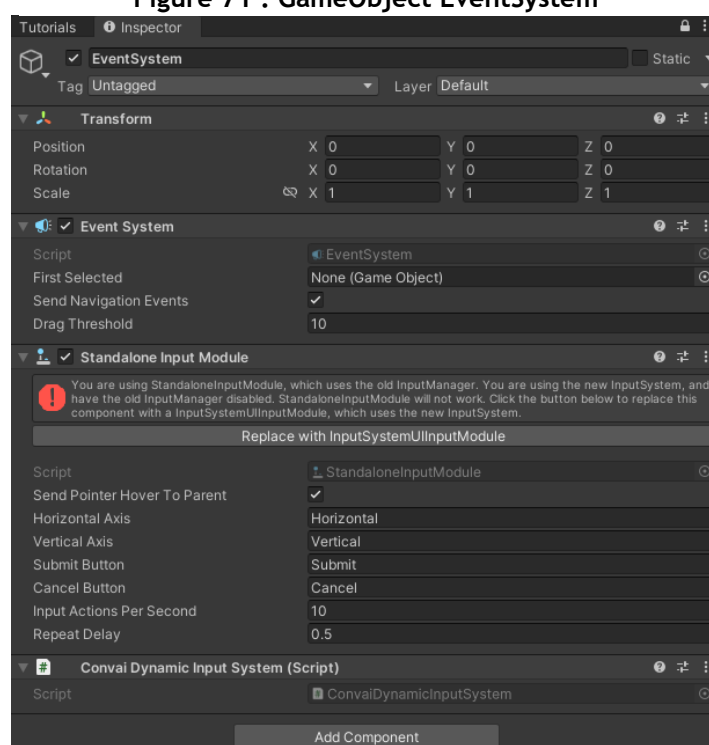
Figure 70 : Aperçu du contenu du Prefab Convai Essentials



Source : données de l'auteur

Ainsi, après avoir déballé complètement ce Prefab, le premier élément à enlever est le GameObject « Convai Player Character ». En effet, le personnage du joueur est déjà présent dans la scène de Sim3D sous le GameObject MalePlayer, comme mentionné au point 3.2.3 de ce document, lors de la configuration du XR Device Simulator.

Figure 71 : GameObject EventSystem

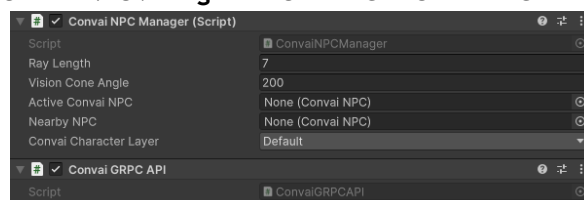


Source : données de l'auteur

Le second élément à supprimer est le GameObject « EventSystem », lui aussi déjà présent dans la scène de jeu. Il faut cependant rajouter à celui déjà existant le script « ConvaiDynamicInputSystem », disponible sous « Assets > Convai > Scripts > Runtime > Addons > Player ». La figure 71 montre le GameObject EventSystem créé durant le projet Sim3D, avec ce nouveau script dans ses composants.

Pour récupérer les fonctionnalités du GameObject Convai Player Character qui a été supprimé, il faut alors ajouter deux scripts au MalePlayer, plus précisément au GameObject « Main Camera », situé dans « Camera Offset », lui-même contenu dans XR Origin. Ces scripts, illustrés en figure 72, sont « ConvaiNPCManager » et « ConvaiGRPCAPI », tous deux disponibles dans « Assets > Convai > Scripts > Runtime > Core ». Cela va permettre respectivement de détecter quel PNJ le joueur observe et d'effectuer l'échange avec le serveur Convai pour générer les réponses du personnage.

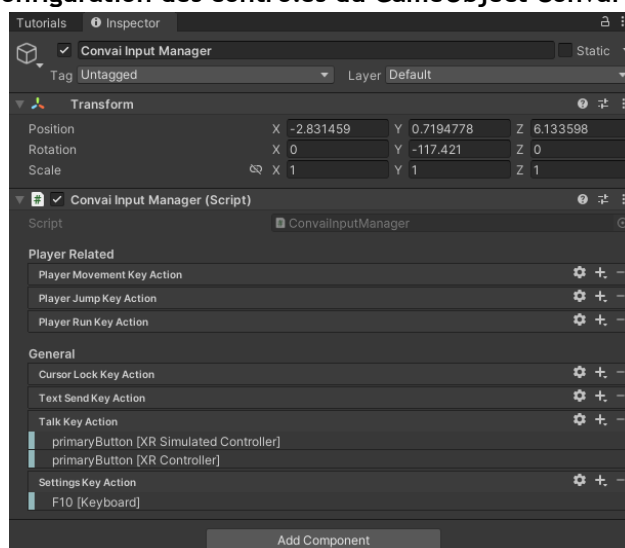
Figure 72 : Scripts ConvaiNPCManager et ConvaiGRPCAPI du GameObject Main Camera



Source : données de l'auteur

Enfin, il faut encore déplacer le GameObject « Convai Input Manager » à l'intérieur du MalePlayer, et le reparamétrer. En effet, tout ce qui concerne le déplacement est déjà intégré dans Sim3D. Les seuls éléments à conserver sont l'activation du micro de Convai et l'ouverture des paramètres, même si cette dernière n'est pas indispensable. La figure 73 montre ces changements afin de pouvoir fonctionner avec les manettes du casque de réalité virtuelle, à l'exception du bouton de configuration, qui a été laissée sur la touche du clavier « F10 » par défaut.

Figure 73 : Configuration des contrôles du GameObject Convai Input Manager



Source : données de l'auteur

Pour finir, en ce qui concerne la partie interface graphique de Convai, il n'y a rien à reparamétrer, si ce n'est le réticule vert pour indiquer la direction du regard du joueur. Ce dernier est utile pour les scènes de démonstration de Convai, vu qu'ils implémentent parfois plusieurs personnages, et que cela permet de préciser de manière graphique quel personnage le joueur est actuellement en train de regarder pour déterminer avec lequel il va converser. Comme le prototype implémente uniquement le personnage de l'infirmière, j'ai changé le réticule pour qu'il soit transparent afin de ne pas perdre son utilisation, mais pour l'enlever de l'écran du joueur.

Avec tous ces changements, il est ainsi possible de dialoguer avec l'infirmière dans la scène de jeu, la figure 74 étant une illustration d'un premier échange à ce stade du prototype.

Figure 74 : Conversation avec le personnage de l'infirmière



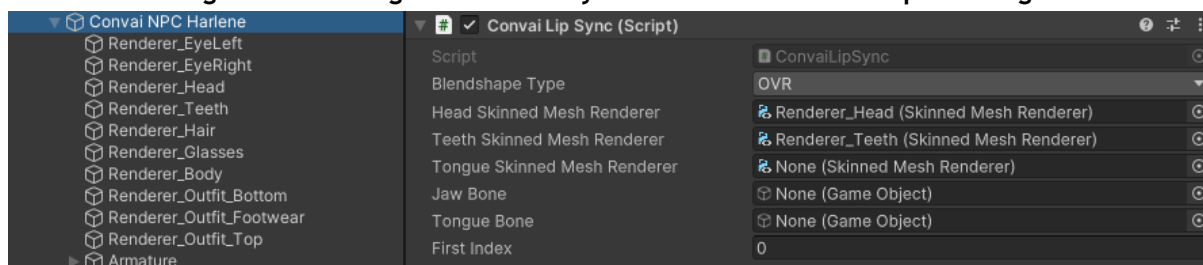
Source : données de l'auteur

3.3.3. ACTIVATION DE LA SYNCHRONISATION LABIALE

Selon la documentation de Convai, l'activation de la synchronisation labiale du personnage s'effectue depuis son GameObject. Pour le prototype il s'agit de Convai NPC Harlene. Il y a alors un composant nommé « ConvaiNPC », et un bouton « Add Components » permet de rajouter des fonctionnalités supplémentaires au PNJ, dont le suivi du regard, activé par défaut à l'importation du personnage comme observé au point 3.3.1, et la synchronisation labiale. Il suffit alors de cocher « Lip Sync » dans cette fenêtre et de confirmer avec le bouton « Apply Changes ».

Ainsi, un nouveau composant s'ajoute au GameObject du personnage, nommé « Convai Lip Sync (Script) ». Toujours selon la documentation, pour les avatars Ready Player Me, il faut ensuite glisser deux enfants du GameObject du personnage, soit « Renderer_Head » et « Renderer_Teeth » dans les sections correspondantes pour activer la synchronisation labiale, comme le montre la figure 75. Pour la section « Blendshape Type », il faut choisir « OVR » pour les avatars Ready Player Me.

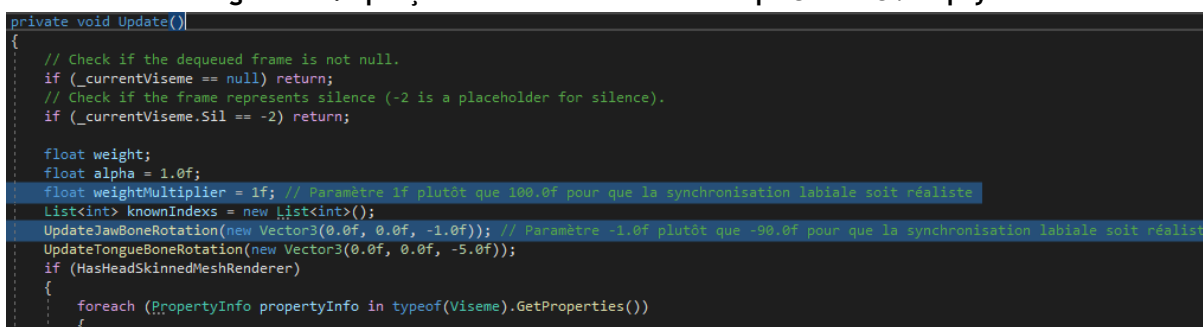
Figure 75 : Configuration de la synchronisation labiale du personnage



Source : données de l'auteur

Cependant, il faut encore paramétrer un script concernant le « Blendshape » sélectionné, comme l'illustre la vidéo de Sarge (2023). Dans le cas du prototype, le script à modifier se nomme « ConvaiOVRlipsync », se trouvant dans « Assets > Convai > Scripts > Runtime > Features > LipSync > Types ». Dans ce script, et selon la vidéo mentionnée précédemment, certaines valeurs doivent être réduites pour que les mouvements des lèvres du personnage paraissent réalistes durant l'échange. J'ai donc reparamétré toutes les valeurs supérieures à 10f et les ai réduits à 1f (ou -1f s'il s'agissait d'une valeur inférieure à -10f). La figure 76 illustre deux exemples de ces changements pour régler la synchronisation labiale.

Figure 76 : Aperçu des modifications du script ConvaiOVRlipsync

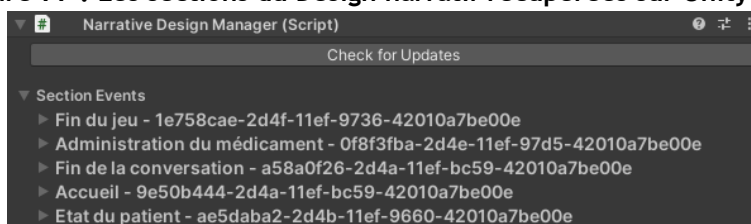


Source : données de l'auteur

3.3.4. INTÉGRATION DU DESIGN NARRATIF

De la même manière que pour la synchronisation labiale, l'intégration du Design narratif nécessite d'activer au préalable une fonctionnalité supplémentaire au composant ConvaiNPC du personnage. Cette fois, c'est l'élément « Narrative Design Manager » à cocher. À l'application des changements, un nouveau composant s'ajoute au personnage, nommé « Narrative Design Manager (Script) ». La figure 77 montre que les sections définies au point 3.1.10 ont été récupérées.

Figure 77 : Les sections du Design narratif récupérées sur Unity

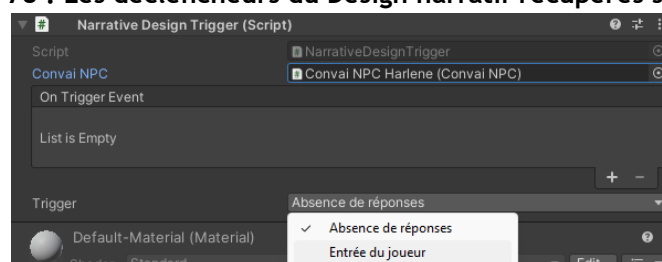


Source : données de l'auteur

Durant une conversation avec le PNJ sur Unity, les sections s'embranchent et se mettent correctement à jour au fil de l'échange, de la même manière qu'une discussion effectuée sur la plateforme en ligne de Convai. Cependant, les déclencheurs n'apparaissent pas dans le composant nouvellement ajouté au personnage. Mais ils ne sont pas perdus pour autant.

En effet, selon la documentation, il est possible de créer un GameObject qui puisse fonctionner en tant que tel. Ainsi, avec un clic droit dans la hiérarchie de la scène de jeu, puis « 3D Object > Cube », je crée un simple cube que je repositionne en face du joueur au lancement du jeu, au niveau de sa tête. Pour le définir en tant que déclencheur, il faut ajouter un nouveau composant à ce cube, un script nommé « NarrativeDesignerTrigger », qui se trouve sous « Assets > Convai > Scripts > Runtime > Features > NarrativeDesign > Runtime ». Ensuite, il faut assigner le personnage Convai qui va réagir, puis sélectionner le déclencheur dans la liste déroulante. La figure 78 illustre ce composant, avec la liste déroulante ayant automatiquement récupéré les deux déclencheurs du graphe narratif créé au point 3.1.10.

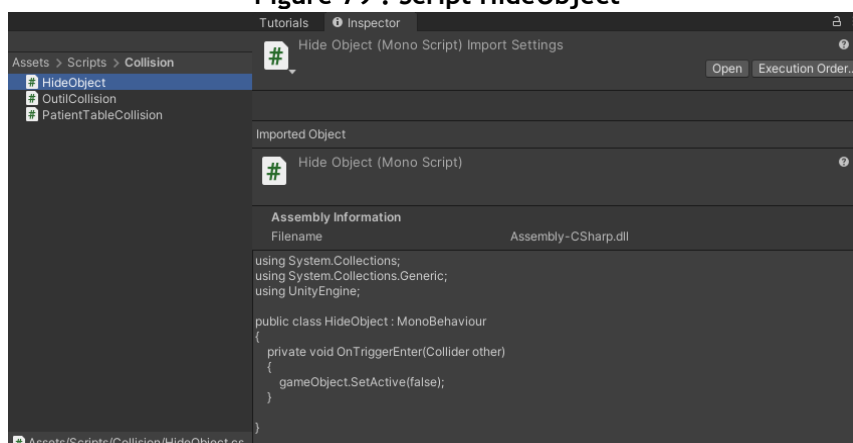
Figure 78 : Les déclencheurs du Design narratif récupérés sur Unity



Source : données de l'auteur

Le GameObject de ce cube est renommé ensuite en « AccueilTrigger », vu que le déclencheur sélectionné est celui de l'entrée du joueur, menant ensuite à la section « Accueil ». De plus, j'ai configuré ce GameObject pour qu'il soit invisible aux yeux du joueur, que sa collision soit une zone qui détecte ce qui rentre à l'intérieur grâce à la coche « Is Trigger » du composant « Box Collider », et qu'il disparaisse au moment où une collision est détectée avec un simple script personnel.

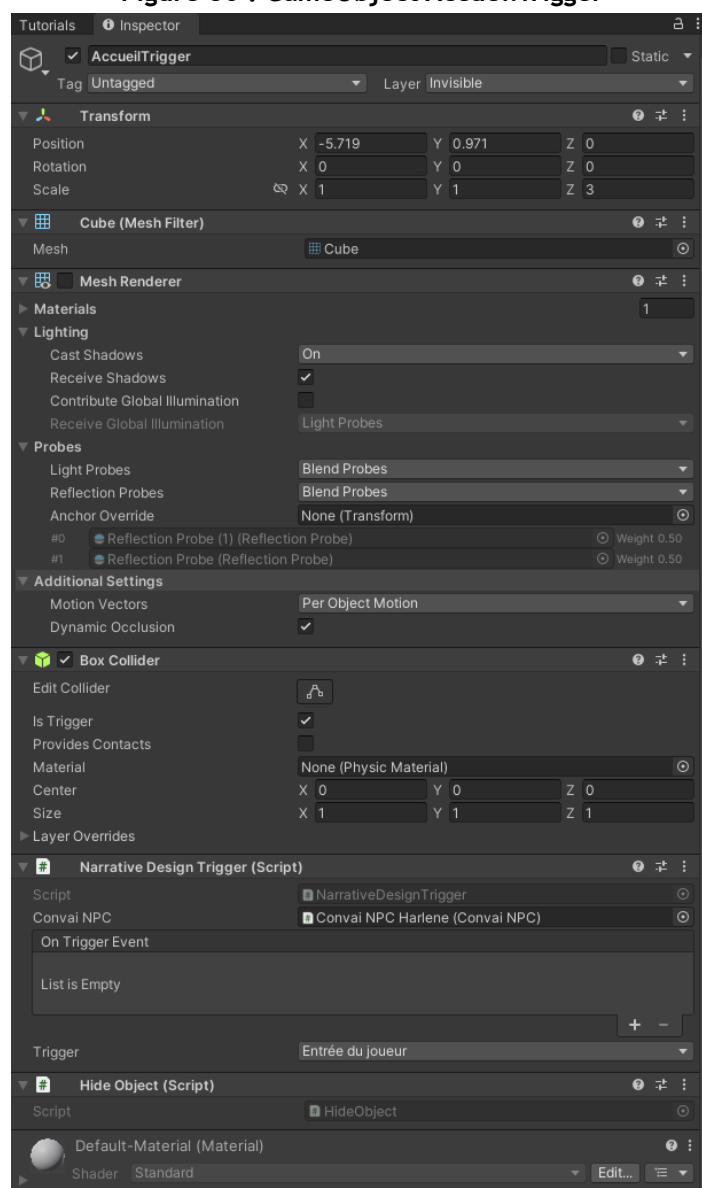
Figure 79 : Script HideObject



Source : données de l'auteur

Ce script, illustré en figure 79, indique que lorsqu'une collision est détectée par l'objet, ce dernier se désactive de la scène du jeu. Cela permet ainsi d'éviter au joueur de rentrer plusieurs fois dans cette zone et d'activer de manière répétée le déclencheur de l'entrée du joueur. Ce script est enregistré dans le dossier concernant les collisions des objets créés durant le projet Sim3D, donc sous « Assets > Scripts > Collision ». Enfin, la construction complète de ce cube se trouve en figure 80.

Figure 80 : GameObject AccueilTrigger

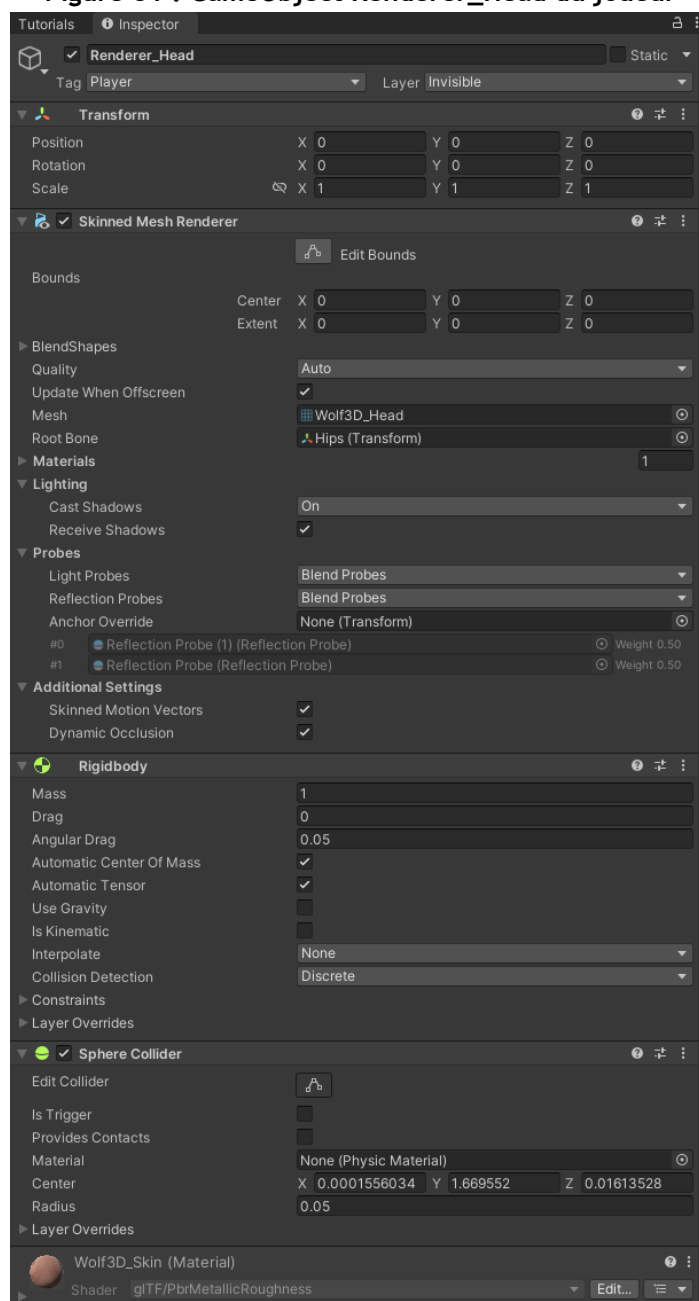


Source : données de l'auteur

Cependant, il faut encore créer un GameObject qui va devoir rentrer dans la zone que constitue ce cube. Cet objet doit alors posséder une collision, et son rôle est de pouvoir être détecté pour que le déclencheur puisse s'activer, et permettre ainsi à l'infirmière d'initier l'échange. Pour ce faire, étant donné que le cube est positionné à la même hauteur que la tête de l'avatar du joueur,

j'ai alors ajouté à ce dernier, plus précisément au GameObject `Renderer_Head` de l'objet `AvatarDoc1`, deux composants pour activer la collision : un « Rigidbody » et un « Sphere Collider ». Pour la sphère de collision, j'ai modifié son rayon pour qu'elle soit inférieure à la tête de l'avatar. La figure 81 montre la composition du GameObject `Renderer_Head` après ces modifications.

Figure 81 : GameObject `Renderer_Head` du joueur



Source : données de l'auteur

En plus de cela, le script du déclencheur `Convai` ne détecte que les objets ayant une balise spécifique, nommée « tag » sur Unity. La figure 81 montre que le GameObject contenant la sphère qui rentrera dans la zone constituée par l'objet `AccueilTrigger` possède le tag « Player ». En effet,

cette balise est définie directement à l'intérieur du code du script NarrativeDesignTrigger, la seule qui permet au déclencheur de fonctionner. La figure 82 illustre la méthode employée par Convai pour détecter un objet, la même que celle utilisée dans le script HideObject pour cacher le GameObject une fois la collision détectée.

Figure 82 : Méthode OnTriggerEnter du script NarrativeDesignTrigger

```
Message Unity | 0 références
private void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("Player")) InvokeSelectedTrigger();
}
```

Source : données de l'auteur

Grâce à toutes ces modifications, lorsque l'on exécute la scène en mode Play, il suffit alors au joueur d'avancer de quelques pas pour que la tête de son avatar rentre dans la zone définie par le GameObject AccueilTrigger, et active le déclencheur de l'entrée du joueur. La figure 83 présente le résultat obtenu. L'infirmière initie l'échange dès lors que le joueur avance dans la chambre.

Figure 83 : L'infirmière initiant la conversation lorsque le joueur s'avance



Source : données de l'auteur

Le déclencheur qui a été créé s'amorce alors selon la position du joueur dans une certaine zone. Cependant, malgré les différents types de déclencheurs décrits par la documentation Convai, il n'est pas possible d'en implémenter un qui soit basé sur la durée ou à la suite d'un événement survenu dans la scène. En effet, les ressources mises à disposition par Convai n'ont pas l'air de supporter ces autres types de déclencheurs. De plus, la documentation ne donne pas d'informations supplémentaires pour les intégrer, ou en tout cas pas pour le moteur de jeu Unity. De ce fait, l'infirmière du prototype ne pourra pas initier l'échange après une absence de réponses du joueur pendant une certaine durée, comme il était normalement prévu par le graphe narratif.

3.3.5. TEST DES RÉPONSES

Il convient aussi d'analyser les réponses du personnage pour observer ses capacités et ses limites. L'annexe VII illustre une transcription complète d'une conversation effectuée avec l'infirmière sur Unity, sans avoir apporté de modifications supplémentaires au PNJ sur la plateforme Convai, traitée au sous-chapitre 3.1 de ce rapport. Trois aspects importants sont à observer dans cet échange : le suivi du scénario, la mémoire du personnage, et la capacité de ce dernier à conserver son rôle.

Selon cette annexe, il est possible de constater dès les premiers échanges que le personnage de l'infirmière, Harlene, a bien compris la description et le Design narratif qui lui ont été fournis. La transcription montre une interaction réaliste et structurée, le PNJ jouant son rôle après avoir assimilé assez rapidement le contexte hospitalier, ma fonction de médecin et le scénario attendu. L'infirmière virtuelle fournit, de manière vocale pour le moment, les outils nécessaires pour l'examen médical et m'assiste donc proactivement. Chaque interaction respecte les étapes du scénario définies au cours du projet Sim3D, du début à son accomplissement, tout en prenant en compte les quatre outils du médecin et la transmission du médicament.

La mémoire du personnage se manifeste à travers la capacité à se rappeler des informations échangées durant la conversation. La liste des outils à utiliser pour l'examen médical est bien assimilée, et lorsque je demande quels sont ceux disponibles après avoir employé le thermomètre, Harlene liste les appareils restants sans confusion : « *J'ai aussi à disposition le pulsomètre, le tensiomètre et le stéthoscope* ». De plus, elle conserve mon prénom dans sa mémoire et l'emploie tout au long de la conversation, en m'appelant « *docteur Bastien* », ce qui ajoute une touche de personnalisation et de professionnalisme.

Aussi, lorsque je lui demande de « *me passer le dernier outil* ». Harlene sait immédiatement qu'il s'agit du pulsomètre, et le précise ainsi dans sa réponse : « *Voici le pulsomètre, docteur Bastien* ». Cela montre que la mémoire contextuelle d'Harlene lui permet de suivre le déroulement de l'interaction et de répondre de manière précise sans nécessiter de clarification supplémentaire de ma part.

La capacité du personnage à conserver son rôle est particulièrement importante pour obtenir une interaction fluide et cohérente. Harlene démontre cette compétence en maintenant une conversation logique et en adaptant ses réponses à mes actions et à mes questions, sans jamais sortir de son rôle d'infirmière. Notamment à la fin de la conversation, lorsque je fais référence à un supposé « *rendez-vous romantique dans un autre pays américains* », Harlene réalise que cet événement concerne un contexte totalement hors-sujet et me corrige poliment : « *Euh... je crois que vous devez faire erreur. Je suis simplement ici pour m'occuper des patients et aider durant les examens médicaux* ». Harlene ne commet pas l'erreur d'inventer des éléments pour me suivre sur

ce sujet. Cela montre que sa mémoire est programmée pour se concentrer sur le contexte professionnel et fournir des interactions pertinentes par rapport au scénario défini.

Un autre exemple illustre parfaitement le respect du scénario, le maintien de la mémoire et du rôle du personnage en fin de conversation. En effet, lorsque je demande à Harlene son prénom après avoir terminé l'examen médical, cette dernière répond ainsi : « *Bien sûr, je m'appelle Harlene. À bientôt, docteur Bastien !* ». Ainsi, le personnage n'a pas oublié son prénom au fil de la discussion, et sait aussi pertinemment que le scénario est fini, vu qu'à ce stade le joueur doit sortir de la chambre.

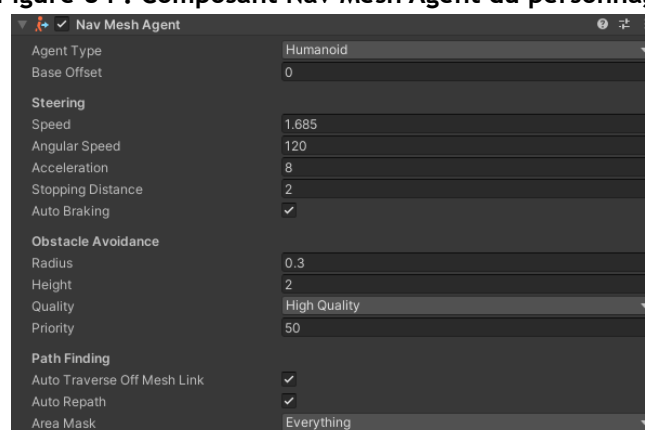
En conclusion, cette transcription en annexe VII montre que le personnage d'Harlene suit correctement le scénario du jeu, utilise une mémoire contextuelle appropriée pour l'interaction en cours et respecte son rôle, ce qui offre une expérience utilisateur fluide et immersive. La capacité du personnage à garder en mémoire des détails comme mon nom, les outils restants à utiliser et l'étape actuelle du scénario renforce alors la crédibilité de l'interaction, faisant d'elle un personnage intelligent et efficace dans son rôle d'infirmière virtuelle.

3.3.6. DÉPLACEMENT DU PERSONNAGE EN JEU

Pour intégrer la fonctionnalité de déplacement à ses avatars, Convai emploie dans ses scènes de démonstrations des éléments fournis par Unity. En effet, leurs personnages possèdent dans leurs composants un script nommé « Nav Mesh Agent ». En plus de cela, la surface du terrain de la scène de jeu contient quant à lui un script « NavMeshSurface ». Ainsi, afin de donner à l'infirmière la possibilité de se mouvoir dans la chambre d'hôpital, j'ai décidé d'incorporer de manière similaire ces scripts à la scène du prototype.

Dans un premier temps, j'ajoute au GameObject Convai NPC Harlene un nouveau composant Nav Mesh Agent, que je configure avec les mêmes valeurs utilisées par Convai dans ses scènes, comme illustré en figure 84.

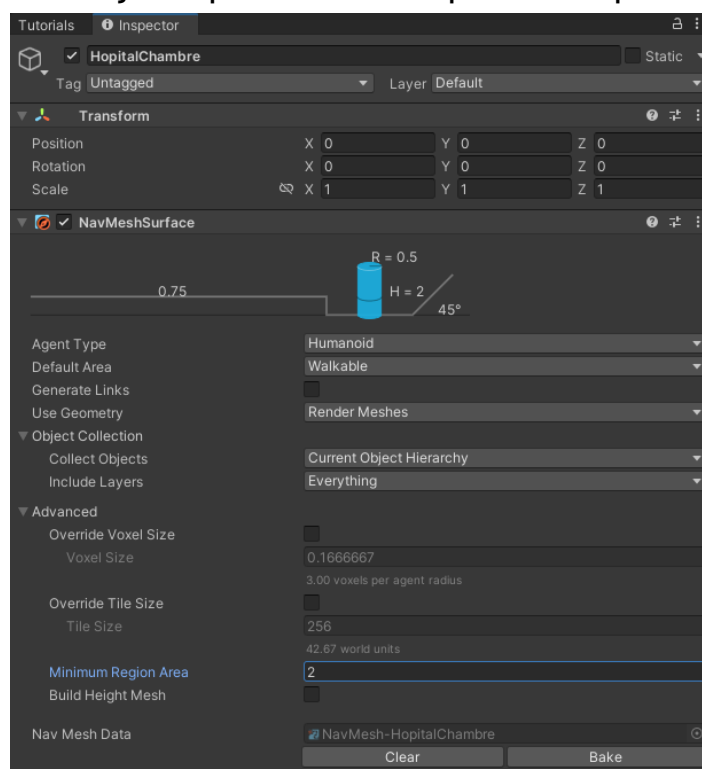
Figure 84 : Composant Nav Mesh Agent du personnage



Source : données de l'auteur

Ensuite, il faut ajouter le NavMeshSurface à l'environnement de la scène. Le GameObject qui englobe tous ces éléments s'appelle « HopitalChambre ». Il y regroupe alors les murs, le lit du patient, le paysage extérieur et d'autres modèles qui contribuent à donner cet aspect de chambre d'hôpital. En figure 85 est affichée la construction de ce GameObject avec ce nouveau script.

Figure 85 : GameObject HopitalChambre incorporant le script NavMeshSurface

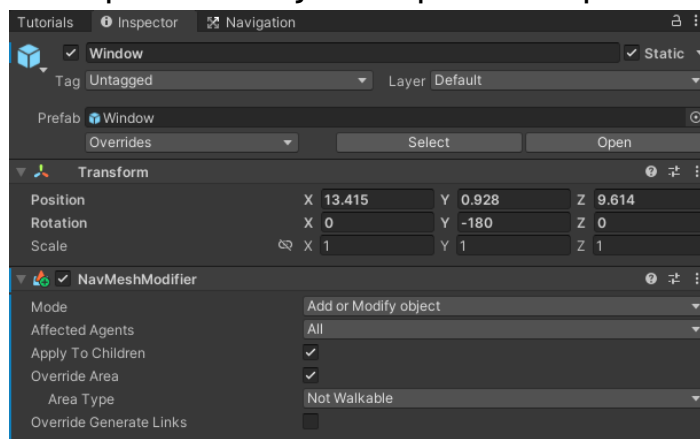


Source : données de l'auteur

À l'aide de la vidéo de LlamAcademy (2021), afin que tous les enfants du GameObject HopitalChambre puissent servir à la navigation, j'ai modifié le paramètre « Collect Objects » sous la partie « Object Collection » du script NavMeshSurface. En sélectionnant « Current Object Hierarchy », comme sur la figure 85, tous les éléments constituant l'environnement de la scène seront ainsi pris en compte dans le calcul des surfaces où l'infirmière pourra s'y déplacer.

Cependant, il serait inapproprié que le personnage puisse, par exemple, considérer le lit du patient comme une zone navigable. Ainsi, toujours selon la vidéo de LlamAcademy, tous les GameObjects enfants de HopitalChambre que l'infirmière ne doit pas considérer dans le calcul de sa trajectoire, tels que le mobilier ou les éléments du paysage entre autres, se voient attribués d'un composant NavMeshModifier, un exemple étant illustré en figure 86. Pour qu'ils soient exclus de la navigation, il faut cocher la catégorie « Override Area », puis sélectionner dans la liste à choix qui apparaît pour « Area Type » l'élément « Not Walkable ».

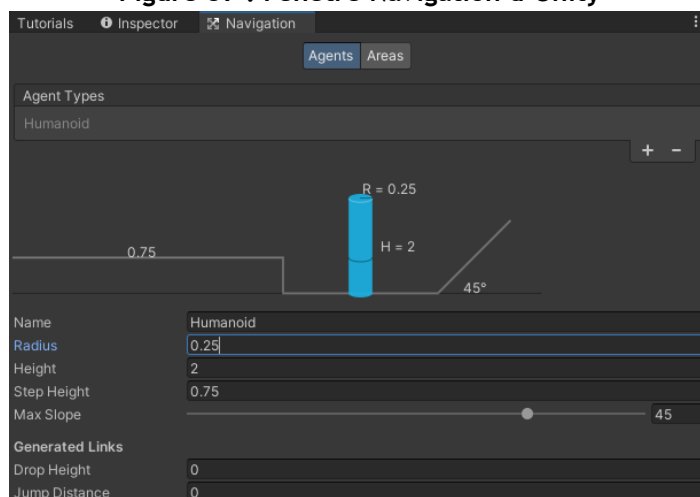
Figure 86 : Exemple de GameObject incorporant le script NavMeshModifier



Source : données de l'auteur

En plus de cela, il est nécessaire de modifier la largeur des agents qui sont définis par le script Nav Mesh Agent, afin que la surface navigable soit suffisamment espacée pour permettre à l'infirmière de se déplacer convenablement. Pour ce faire, il faut aller dans la barre de menu Unity, sous « Window > Ai > Navigation ». Une fenêtre « Navigation » s'affiche, avec l'onglet « Agents » déjà sélectionné. Pour modifier leur largeur, j'ajuste la catégorie « Radius » à 0.25 plutôt que la valeur par défaut qui est 0.5. La figure 87 montre cet onglet avec la modification effectuée.

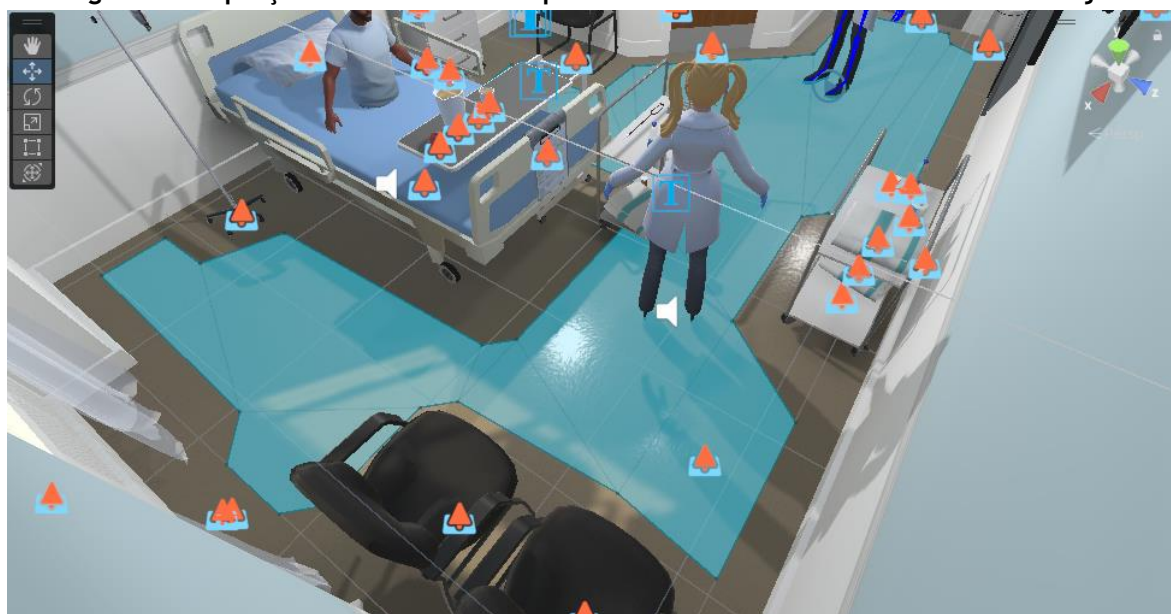
Figure 87 : Fenêtre Navigation d'Unity



Source : données de l'auteur

Grâce à tous ces changements, avec le bouton « Bake » qui se trouve au fond du composant NavMeshSurface, un calcul de la surface navigable des agents est effectué puis affiché de manière graphique. Ainsi, la figure 88 montre un aperçu de la zone où l'infirmière peut se déplacer. Comme il est possible de le constater, la zone de déplacement consiste essentiellement du sol de la chambre d'hôpital, avec la plupart du mobilier qui n'a pas été considéré dans le calcul grâce au script NavMeshModifier, représenté avec une icône de flèche rouge sur une surface bleu ciel dans Unity.

Figure 88 : Aperçu de la surface de déplacement de l'infirmière dans la scène de jeu



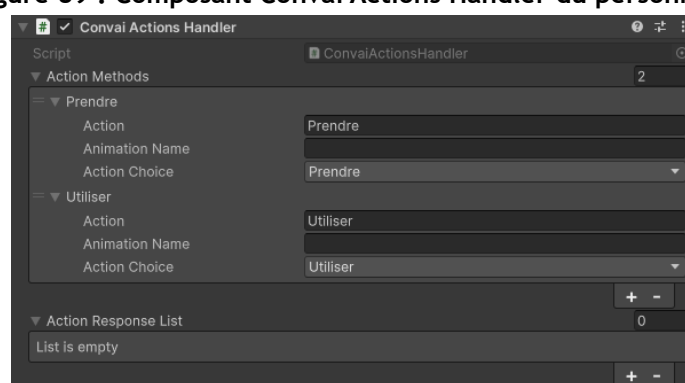
Source : données de l'auteur

3.3.7. ASSIGNATION DES ACTIONS

En plus de la documentation officielle, la vidéo de tutoriel réalisée par Convai en 2024 a fourni une aide précieuse pour cette partie. Cependant, il est quand même précisé que les actions des personnages sont considérées comme des éléments en phase expérimentale, un aspect plus développé au point 3.3.10 concernant le test du jeu.

Malgré tout, le composant ConvaiNPC permet donc d'ajouter la fonctionnalité « NPC Actions », créant ainsi un nouveau script « Convai Actions Handler » qui s'ajoute au GameObject. La liste des actions à construire s'effectue sous « Actions Methods ». Chacune d'entre elles possède un titre à définir sous « Action » et un nom d'animation sous « Animation Name » s'il s'agit uniquement au personnage d'en jouer une, mais cela n'a pas été utilisé dans le prototype. Enfin, l'action à réaliser se sélectionne dans « Action Choice ». Cette dernière n'est pas obligée d'avoir le même nom que le titre de l'action. En figure 89 est illustré la liste des actions définies pour l'infirmière.

Figure 89 : Composant Convai Actions Handler du personnage

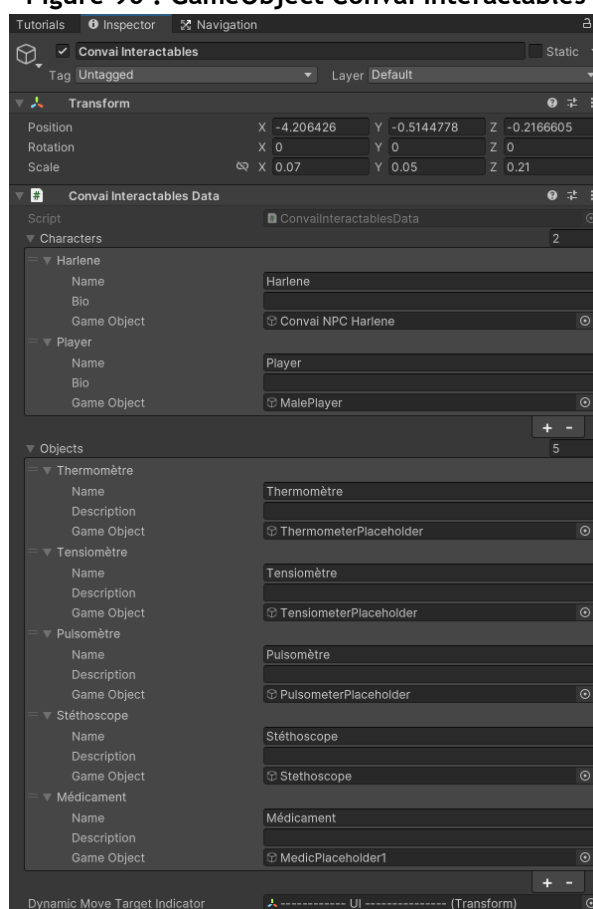


Source : données de l'auteur

Il est important de trouver le verbe correct pour le titre de l'action. En effet, le personnage Convai va détecter à partir des paroles du joueur les gestes qui lui seront demandés, construits dans une liste avec, pour chacun de ses éléments, un verbe comme premier mot. Les éléments de cette liste sont donc comparés à toutes les actions définies sous Actions Methods du personnage. S'il n'y a aucune concordance, le PNJ ne réalisera pas d'actions.

En plus de ce composant, il faut créer un GameObject de configuration afin de définir les objets cibles que le personnage devra saisir ou s'y diriger. Pour cela, il suffit d'en créer un vide dans la hiérarchie de la scène, nommé « Convai Interactables ». Ensuite, il faut attacher à ce GameObject le script « ConvaiInteractablesData », se trouvant sous « Assets > Convai > Scripts > Runtime > Features > Actions ». Dans ce composant se trouve deux sections, une consacrée aux personnages et la seconde pour les objets. Ainsi, je définis l'infirmière et le joueur avec leur GameObject respectif, et j'effectue la même chose pour les outils du médecin et le médicament. Ces objets se trouvent sous le GameObject « PlayerChariot », soit le chariot du joueur créé durant Sim3D. À l'exception du stéthoscope, tous ces objets sont représentés par de petits cubes dans la scène, et possèdent le terme « Placeholder » dans leurs noms. La figure 90 montre le résultat obtenu du GameObject Convai Interactables après sa configuration.

Figure 90 : GameObject Convai Interactables



Source : données de l'auteur

En plus de cela, un champ « Dynamic Move Target Indicator » est présent. Cependant, n'ayant que peu d'informations quant à son utilité, et ne pouvant faire fonctionner les actions du personnage en laissant ce champ vide, je lui attribue alors le GameObject englobant la totalité de l'interface graphique de Convai, ce qui n'a pas semblé poser de problèmes majeurs dans les tests du jeu réalisés à la suite de cela. De plus, tout comme les titres des actions du composant Convai Actions Handler du personnage, il est important d'écrire correctement le nom des objets qui seront à saisir par l'infirmière. Ces derniers seront aussi comparés aux objets interprétés par le personnage dans son échange avec le joueur après l'analyse du verbe de l'action.

Le script où sont effectués ces comparaisons, mais aussi où il est possible de définir de nouvelles actions comme les deux attribuées à l'infirmière en figure 89 « Prendre » et « Utiliser », se nomme « ConvaiActionsHandler », et est enregistré au même endroit où est stocké le script ConvaiInteractablesData. D'abord, j'ai dû adapter la méthode nommée « ParseActions », celle qui réalise la comparaison du verbe et des objets de l'action. En effet, une comparaison stricte est effectuée entre les objets définis sous le GameObject Convai Interactables et celui interprété par le personnage lors de l'échange, ce dernier pouvant varier à chaque session de jeu. Afin de faciliter la détection de l'objet à saisir, j'ai modifié le code pour vérifier que la chaîne de caractères de l'objet interprété contienne le nom de celui voulu à la place de cette comparaison qui attend uniquement le nom exact de l'objet. La figure 91 illustre ce changement.

Figure 91 : Modification de la méthode ParseAction du script ConvaiInteractablesData

```
// Iterate through each object in global action settings to find a match
foreach (ConvaiInteractablesData.Object @object in _interactablesData.Objects)
{
    // Modification de la comparaison pour passer d'un .Equals à un .Contains pour faciliter la recherche de l'objet
    /*if (string.Equals(@object.Name, string.Join(" ", tempString2),
        StringComparison.CurrentCultureIgnoreCase))*/
    if (string.Join(" ", tempString2).Contains(@object.Name, StringComparison.CurrentCultureIgnoreCase))
    {
        Logger.DebugLog($"Active Target: {string.Join(" ", tempString2).ToLower()}",
            Logger.LogCategory.Actions);
        tempGameObject = @object.gameObject;
    }
}
```

Source : données de l'auteur

Toujours dans ce script, une énumération nommée « ActionChoice » permet de modifier les actions disponibles dans la liste déroulante du composant Convai Actions Handler du personnage. Pour en ajouter de personnalisées telles que celles attribuées à l'infirmière, et comme le précise la documentation Convai, je mets les noms de mes nouvelles actions à la suite de cette liste, comme l'illustre la figure 92.

Ensuite, une seconde méthode, « DoAction », doit être modifiée afin de pouvoir jouer mes nouvelles actions. Plus précisément, j'inclus deux nouveaux « cases » à la suite de l'instruction « switch » présente, pour permettre l'appel des méthodes correspondantes à ces actions. Ainsi, la figure 93 montre ces ajouts pour les cas Prendre et Utiliser.

Figure 92 : Ajout d'actions supplémentaires à l'énumération ActionChoice

```
// STEP 1: Add the enum for your custom action here.
17 références
public enum ActionChoice
{
    None,
    Jump,
    Crouch,
    MoveTo,
    PickUp,
    Drop,
    OpenWeb,
    Prendre, // Ajout d'une nouvelle action dans la liste déroulante
    Utiliser
}
```

Source : données de l'auteur

Figure 93 : Ajout à la méthode DoAction de cases correspondant aux actions supplémentaires

```
case ActionChoice.None:
    // Call the AnimationActions function and yield until it's completed
    yield return AnimationActions(action.Animation);
    break;

case ActionChoice.OpenWeb:
    OpenWebsite();
    break;

// Ajouter la nouvelle action dans le switch de la méthode DoAction
case ActionChoice.Prendre:
    yield return Prendre(action.Target);
    break;

// Ajouter la nouvelle action dans le switch de la méthode DoAction
case ActionChoice.Utiliser:
    yield return Utiliser(action.Target);
    break;
}
```

Source : données de l'auteur

Enfin, il faut écrire les méthodes qui seront appelées par DoAction. Comme expliqué précédemment, le personnage Convai construit une liste d'actions interprétées selon les paroles du joueur. Cependant, tous mes essais avec le prototype ont montré que malgré une suite d'actions trouvées, le personnage n'en réalisait que la première et pas les suivantes. Afin de forcer l'infirmière à effectuer tous les gestes attendus, la méthode que j'ai écrite pour l'action Prendre est adaptée du code des actions déjà présentes dans ce script. Ainsi, l'infirmière va d'abord se diriger vers l'objet cible grâce au code écrit pour l'action « MoveTo », ramasse ensuite la cible avec la méthode « PickUp », se déplace à nouveau avec MoveTo vers la destination, et finalement pose l'objet avec une méthode personnelle adaptée à partir de l'action « Drop » de Convai.

Une construction similaire est réalisée pour l'action Utiliser, où le personnage va à nouveau se rendre vers l'outil, le ramasser et se déplacer vers le patient. Ensuite, je mime son utilisation de l'appareil par une attente de cinq secondes. Enfin, l'infirmière se dirige vers le chariot contenant les outils du médecin, pour y déposer l'outil emprunté. J'ai estimé que ces deux actions, Prendre et Utiliser, faisaient sens dans le cadre de la scène de jeu du prototype.

L'action Drop de Convai permet au personnage de déposer l'objet tenu dans la scène, mais sans que ne se joue une animation, contrairement aux actions MoveTo et Pickup par exemple. Afin de donner l'impression que le personnage dépose effectivement l'objet, j'ai alors écrit une nouvelle méthode d'action, uniquement appelée par mes actions personnalisées, que j'ai nommé « Poser ». Elle combine le code des méthodes Pickup et Drop de Convai pour que l'infirmière puisse jouer l'animation de la saisie de l'objet, mais au lieu de le ramasser, il est déposé dans la scène de jeu, en tant que GameObject enfant de sa destination. La figure 94 présente la méthode Poser, qui est un exemple de construction d'une action personnalisée.

Figure 94 : Code de l'action personnalisée Poser

```
private IEnumerator Poser(GameObject target, GameObject place)
{
    ActionStarted?.Invoke("Poser", target);

    Logger.DebugLog($"Dropping Target: {target.name}", Logger.LogCategory.Actions);

    // Retrieve the Animator component from the current NPC.
    Animator animator = _currentNPC.GetComponent<Animator>();

    // Start the "Picking Up" animation with a cross-fade transition.
    animator.CrossFade(Animator.StringToHash("Picking Up"), 0.1f);

    // Wait for one second to ensure that the Animator has had time to transition to the "Picking Up" animation state.
    yield return new WaitForSeconds(1);

    // Define the time it takes for the hand to reach the object in the "Picking Up" animation.
    // This is a specific point in time during the animation that we are interested in.
    float timeToReachObject = 1f;

    // Wait for the time it takes for the hand to reach the object.
    yield return new WaitForSeconds(timeToReachObject);

    // Once the hand has reached the object, set the target's parent to the place's transform, effectively "dropping" the object
    target.transform.parent = place.transform;
    target.SetActive(true);

    // Transition back to the "Idle" animation.
    animator.CrossFade(Animator.StringToHash("Idle"), 0.4f);

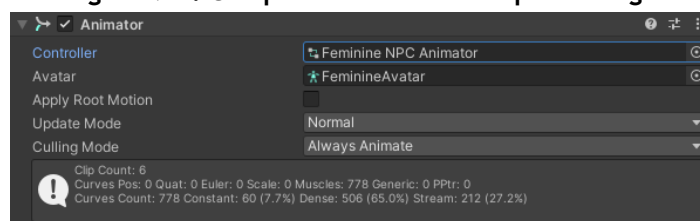
    ActionEnded?.Invoke("Poser", target);
}
```

Source : données de l'auteur

3.3.8. ANIMATION DU PERSONNAGE

À l'importation d'un personnage Convai, un composant Animator est directement créé et configuré pour que les animations se lancent automatiquement dès le lancement du mode Play de la scène Unity. Dans le cas de l'infirmière Harlene, cet élément est illustré en figure 95.

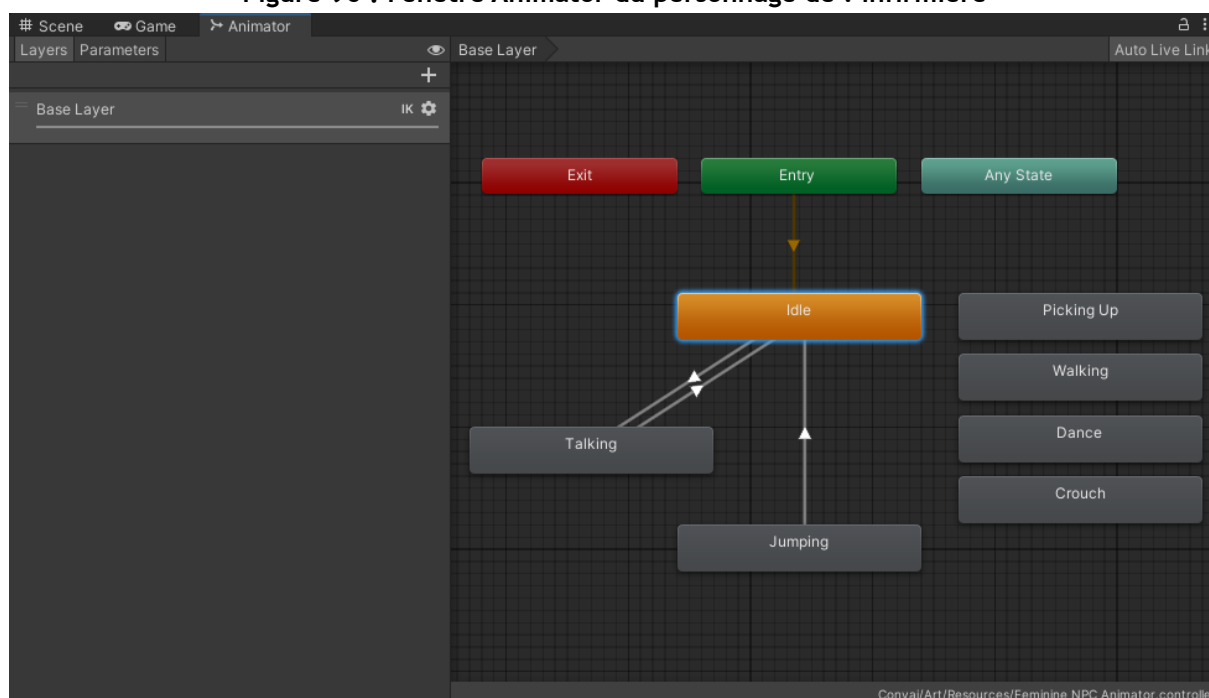
Figure 95 : Composant Animator du personnage



Source : données de l'auteur

Pour ajouter de nouvelles animations, c'est l'élément Animator, défini dans le champ « Controller », qu'il faut modifier. Pour le personnage du prototype, j'ai conservé celui fourni par Convai, « Feminine NPC Animator », qui est disponible sous « Assets > Convai > Art > Resources ». En ouvrant cet élément, une fenêtre Animator s'affiche, et montre la construction de différentes animations données par Convai, visible en figure 96.

Figure 96 : Fenêtre Animator du personnage de l'infirmière



Source : données de l'auteur

Comme il est possible de constater dans cette fenêtre, le graphique présenté est construit à la manière d'un FSM. Chaque animation représente un état, à l'exception de « Entry », « Exit » et « Any State » qui sont des états particuliers. Dans la figure 96, le rectangle « Idle » est coloré en orange, et représente l'animation par défaut que le personnage joue au lancement du jeu. Il possède une transition avec l'état « Talking », qui s'exécute dès lors que le PNJ se met à parler, avant de revenir sur Idle lorsqu'il a terminé son dialogue.

Les autres animations définies par Convai, comme « Picking Up » et « Walking », ne possèdent pas de transitions, à l'exception de « Jumping » qui revient à l'état par défaut une fois que son exécution est terminée. En effet, ces animations sont directement appelées dans le code des méthodes du script ConvaiActionsHandler. En prenant l'exemple de la méthode Poser illustrée en figure 94, l'animation « Picking Up » est exécutée après avoir récupéré au préalable l'Animator du personnage de la manière suivante :

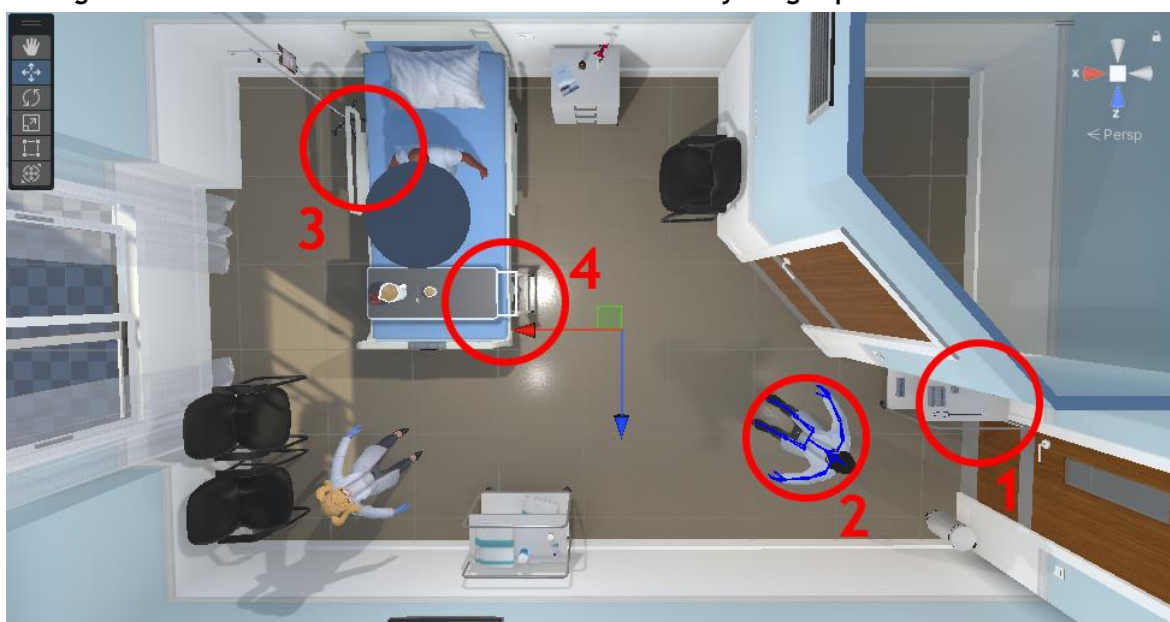
```
Animator animator = _currentNPC.GetComponent<Animator>();
animator.CrossFade(Animator.StringToHash("Picking Up"), 0.1f);
```

Pour revenir à l'état par défaut, un appel similaire mais avec le nom de l'état Idle est réalisé. C'est pourquoi ces animations supplémentaires ne possèdent pas de transitions dans le graphe de l'Animator. Ainsi, lorsque l'on ajoute une nouvelle animation, afin de la faire exécuter par le personnage, il faut l'appeler soit dans une méthode du script ConvaiActionsHandler, soit directement à partir du composant Convai Actions Handler du personnage, en mettant le nom correct de l'état dans le champ Animation Name et sélectionner « None » pour le champ Action Choice.

3.3.9. TEST DE LA SCÈNE

Dans Sim3D, le chariot du joueur contenant tous les outils du médecin se trouvait devant le lit du patient. Pour forcer l'infirmière à se déplacer dans la chambre lorsqu'il faut chercher les objets, j'ai déplacé ce chariot à l'entrée de la pièce. En plus de cela, j'ai créé à des endroits différents de la scène de jeu trois GameObjects vides ayant le mot « Depot » dans leur nom. Ils vont servir de destination à l'infirmière selon l'action qu'elle doit effectuer, et prouver que la zone de déplacement du personnage est correctement définie. En figure 97 sont entourés en rouge l'emplacement de ces GameObjects de dépôt.

Figure 97 : Position des zones où l'infirmière devra s'y diriger pour réaliser ses actions



Source : données de l'auteur

Au lancement du jeu, l'infirmière démarre à proximité des chaises côté fenêtre, comme il est possible de le voir dans la figure 97. Le cercle numéro 1 représente le chariot où se trouvent les quatre outils du médecin ainsi que le médicament du patient. En plus d'avoir déjà assigné les différents objets au personnage Convai pour qu'il sache où les ramasser, le GameObject vide créé à cet emplacement se nomme « DepotChariot », afin d'indiquer à l'infirmière où reposer l'outil après son utilisation.

Le second cercle s'agit de l'avatar du joueur, qui est le seul à pouvoir changer de place au sein de la chambre. Le troisième concerne le GameObject indiquant l'emplacement du patient où utiliser l'outil, nommé « DepotPatient ». Enfin, le dernier cercle est la table du patient, représenté par le dernier GameObject vide nommé « DepotTable », afin d'y déposer le médicament du patient.

Selon le Design narratif du personnage, c'est lors de la section de l'état du patient et la suivante, celle de l'administration du médicament, qu'il fait sens au joueur d'instruire le personnage à effectuer les deux actions définies au point 3.3.7. Ainsi, lorsque je demande à Harlene de prendre le thermomètre par exemple, elle se dirige vers l'emplacement de cet outil, initialement sur le chariot à l'entrée de la chambre, le ramasse, et se déplace vers ma position pour y poser l'appareil afin de simuler la transmission de cet objet au joueur. Après avoir posé le thermomètre, Harlene conserve sa nouvelle position dans la chambre, dans ce cas là où elle a déposé l'objet.

Ensuite, quand je demande à l'infirmière d'utiliser le thermomètre, elle se dirige vers l'outil, qui peut soit être sur le chariot, soit à la position où elle l'a posé après l'instruction Prendre. Puis, elle se déplace en direction du GameObject DepotPatient, simule l'utilisation de l'appareil avec cette attente de quelques secondes avant de se diriger vers la position du chariot où sont stockés les outils pour reposer l'objet. Après avoir réalisé l'action Utiliser, Harlene reste à proximité du chariot.

Quand j'indique à l'infirmière que tous les outils ont été utilisés, la section de l'administration du médicament démarre, ce qui démontre que le personnage Convai a bien compris que le joueur a besoin de ces quatre appareils avant de passer à la section suivante. Pour l'instruction Prendre, j'ai paramétré une condition, illustrée en figure 98, qui détecte que si le nom de l'objet cible de cette action est celui du GameObject représentant le médicament, la destination ne serait pas l'avatar du joueur, mais le DepotTable. Ainsi, Harlene se rend auprès du chariot, ramasse le médicament, et se dirige ensuite auprès de cette table pour y poser l'outil.

Figure 98 : Condition pour déterminer la destination du personnage selon l'objet à prendre

```
// 3. Déterminer la cible vers laquelle le personnage va s'y rendre
GameObject tableTarget;
if (target.name.Equals("MedicPlaceholder1"))
{
    tableTarget = GameObject.Find("DepotTable");
}
else
{
    tableTarget = GameObject.Find("AvatarDoc1");
}
```

Source : données de l'auteur

Les méthodes d'actions du script ConvaiActionsHandler qui nécessitent au personnage de se déplacer, soit MoveTo, Prendre et Utiliser, récupèrent chacun le composant Nav Mesh Agent du PNJ. Afin que l'infirmière soit au plus proche des objets à ramasser ou des zones où les déposer, je

modifie un paramètre nommé « `stoppingDistance` » de ce composant, qui est utilisé par la suite pour calculer la distance restante entre le personnage et la destination. Les morceaux de code suivants montrent comment le Nav Mesh Agent est récupéré par ces méthodes, ainsi que la valeur donnée à l'attribut `stoppingDistance` :

```
NavMeshAgent navMeshAgent = _currentNPC.GetComponent<NavMeshAgent>();  
navMeshAgent.stoppingDistance = 0.5f;
```

Avec la configuration du déplacement du personnage ainsi que l'assignation de ses actions, lorsque l'ordre lui est donné et que la tâche à réaliser a été reconnue, l'infirmière se déplace correctement au sein de la chambre. Par exemple, elle fait effectivement le tour du lit après l'utilisation de l'outil sur le patient pour se rendre auprès du chariot et y déposer l'appareil, avec les animations correctes qui sont jouées. Cependant, il est difficile de paramétrer la position de ces objets lorsque l'infirmière les saisit. Même si l'emplacement général est instruit au personnage avec ces différents GameObjects de dépôt, et que sa distance avec la destination soit quasiment inexistante, en général les outils se retrouvent malgré tout par terre au lieu d'être sur la table ou sur le chariot.

De plus, le personnage Convai a parfois du mal à reconnaître les actions à réaliser. Par exemple, elle peut prendre l'initiative d'utiliser directement l'outil elle-même plutôt que de le prendre et l'emmener au joueur et d'attendre que ce dernier lui donne l'ordre de l'utiliser. Parfois, les actions interprétées par le personnage sont traduites en anglais, faisant que l'infirmière ne les réalise pas vu qu'elles sont écrites en français. Ou bien elles sont reconnues mais sans leurs objets cibles, ce qui à nouveau empêche le personnage de réaliser ses actions.

En général, il faut bien prononcer le verbe de l'action attendue pour garantir que son interprétation par l'infirmière soit correcte en plus du nom de l'outil, mais il arrive que par la suite, dans la même partie, elle interprète correctement l'action à réaliser si un autre verbe est utilisé, ou même l'objet à prendre si on lui demande le dernier outil, comme démontré dans la transcription de l'annexe VII.

3.4. BILAN

La limite journalière de Convai reste malgré tout restrictive, notamment lorsque l'on veut tester et ajuster certains éléments. En effet, lors du lancement du jeu, l'initialisation du personnage est comptée dans le quota, ainsi que chaque prise de parole en réponse du joueur ou à la suite du déclencheur. Les actions et les déplacements affichés par le personnage en jeu ne sont pas décomptés, étant donné que ce sont des éléments construits manuellement. Finalement, l'intelligence artificielle du PNJ se présente dans ses réponses, qui démontrent un respect certain de la description et de réactions en accord avec le contexte donné, un suivi intelligent du graphe

narratif qui démontre une certaine attention au scénario, et une interprétation d'actions à entreprendre plus ou moins correcte.

La configuration du personnage sur la plateforme Convai est encore assez simple et peut être réalisée assez rapidement grâce à la documentation officielle. En effet, il m'a fallu un total d'environ 2 heures pour enregistrer toutes les informations du personnage qui sont présentées au chapitre 3.1 de ce rapport. Cependant, la documentation n'est pas aussi précise qu'elle en a l'air lorsque l'on passe dans la partie consacrée au moteur de jeu Unity. Certains éléments ont dû faire l'objet de recherches supplémentaires, notamment les actions et les déplacements du personnage. De plus, les actions ont été particulièrement difficiles à manipuler. Elles sont avant tout destinées à des utilisateurs anglophones, comme le montre la méthode `ParseActions` du script `ConvaiActionsHandler`, celle réalisant la comparaison du verbe et des objets de l'action, qui possède du code dont le but est d'enlever la dernière lettre « s » du verbe s'il en possède.

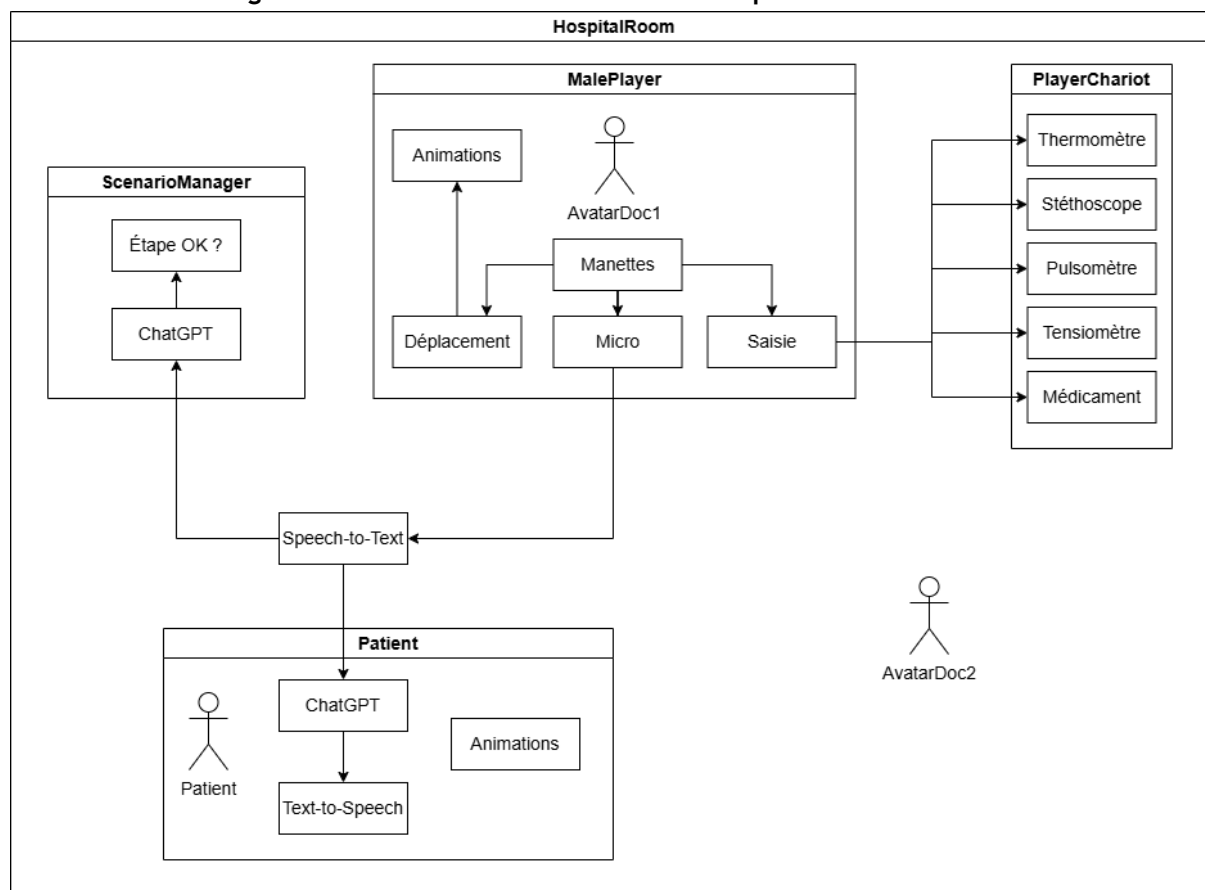
Il est possible que l'infirmière se trompe parfois. Les erreurs d'interprétation des actions à réaliser ont déjà été mentionnées, mais il m'est arrivé que l'infirmière pense que le médicament à transmettre est au médecin, et donc au joueur, plutôt qu'au patient. Ou que le médicament est à utiliser sur le patient, alors que le scénario de Sim3D demande simplement de le déplacer du chariot sur la table du patient. Il est difficile de savoir si cela est dû à la reconnaissance vocale, qui n'écrit pas toujours correctement les mots que je prononce et qui pourrait impacter les réponses du personnage, ou si c'est la faute de ma propre prononciation, ou encore s'il s'agit d'un défaut de compréhension de la description et du Design narratif du personnage durant la session de jeu.

Il est aussi intéressant de noter que l'infirmière n'a finalement pas de détection spatiale, si ce n'est la zone de déplacement construite avec le `NavMeshSurface` d'Unity et les déclencheurs du Design narratif. En effet, même si les outils ne sont pas posés exactement à l'emplacement voulu, il m'est arrivé que, lorsque j'ai terminé de prendre les différentes mesures du patient, l'infirmière me dise que tous les outils ont été reposés sur le chariot. Je ne lui avais alors uniquement demandé de me les apporter sans lui dire de les utiliser, ce qui aurait effectivement remis ces appareils à peu près à leur place sur le chariot. Convai ne fournit donc pas de moyens pour vérifier si certains `GameObjects` sont réunis dans une zone précise avant de poursuivre la progression du graphe narratif.

Un aspect qui n'a pas été mentionné durant le développement du prototype est la possibilité de faire communiquer entre eux deux personnages intelligents. Sur les scènes d'exemple fournies par Convai, disponibles sous « Assets > Convai > Demo > Scenes », il est possible d'observer une telle démonstration, apparemment sans possibilité pour le joueur d'intervenir dans leur discussion. Cependant, comme le développement s'est concentré sur un seul PNJ, cette fonctionnalité n'a pas pu être abordée plus en détail.

Pour conclure, afin de montrer l'évolution du projet Sim3D vers le prototype, j'ai créé deux schémas pour décrire de manière visuelle les fonctionnalités existantes, et celles rajoutées grâce à Convai. La figure 99 montre les fonctionnalités disponibles lors du rendu du projet Sim3D.

Figure 99 : Schéma des fonctionnalités disponibles sous Sim3D



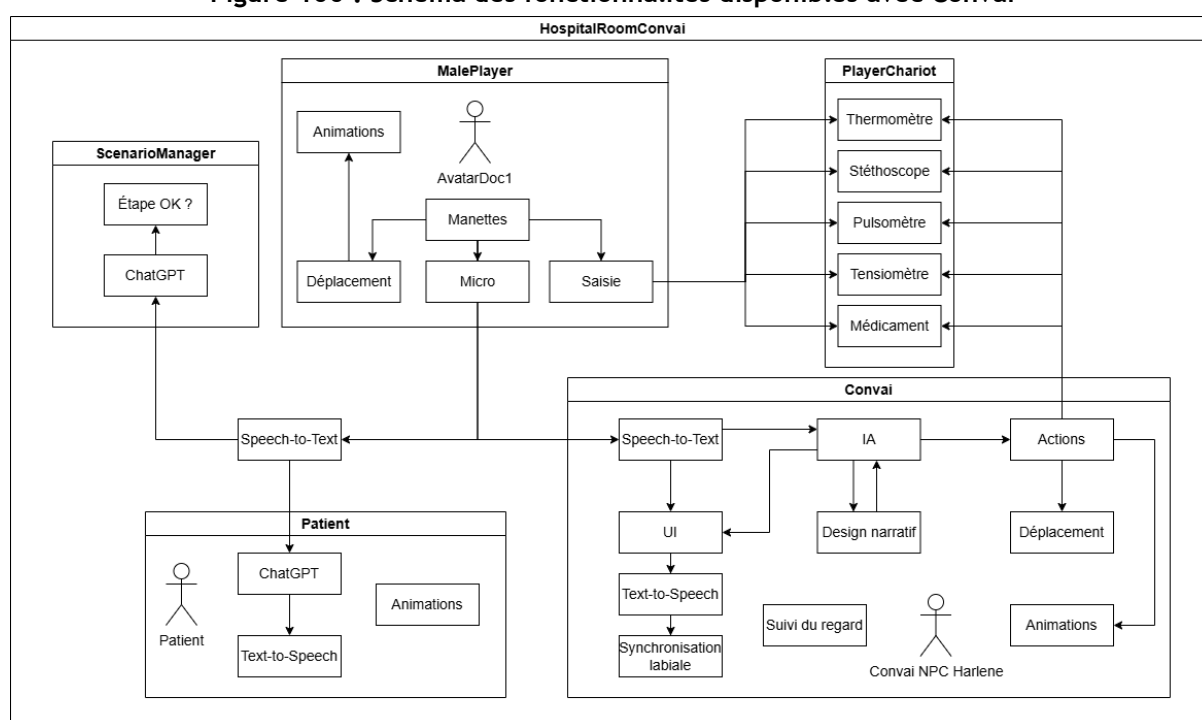
Source : données de l'auteur

Sous la scène HospitalRoom, le joueur est représenté par le GameObject MalePlayer, avec l'avatar de l'infirmier sous AvatarDoc1. Il peut utiliser les manettes du casque de réalité virtuelle pour se déplacer, saisir les objets stockés dans le GameObject PlayerChariot, et activer le micro afin de communiquer. Le déplacement invoque une animation de mouvement du personnage pour ajouter un aspect immersif.

Lorsque le joueur parle, ses paroles sont converties en texte à l'aide d'un script de reconnaissance vocale, représenté sur la figure en tant que « Speech-to-Text ». Ce texte est ensuite transmis à deux instances de ChatGPT. Le patient, représenté par le GameObject « Patient », possède son propre avatar ainsi qu'une des instances de ChatGPT. Cette dernière formule les réponses du personnage en accord avec son rôle, défini à l'aide d'une première instruction. Le texte généré est ensuite passé dans un script de synthèse vocale, représenté par le « Text-to-Speech », permettant ainsi de communiquer oralement les réponses de ChatGPT. Les animations du patient jouent en boucle pour montrer qu'il est couché sur le lit.

La seconde instance de ChatGPT se trouve dans le GameObject « ScenarioManager ». Elle supervise la progression du jeu, s'assurant que les étapes du scénario se déroulent correctement. ChatGPT vérifie si les paroles du joueur répondent à l'objectif actuel, et le ScenarioManager débloque certains éléments de la scène selon l'étape du jeu, notamment la possibilité de saisir les outils du chariot lorsque le joueur doit effectuer la prise des mesures sur le patient. Enfin, l'AvatarDoc2 représente l'infirmière qui, à ce stade, n'est là qu'à titre décoratif. Son évolution est illustrée dans la figure 100, qui reprend ce même schéma en ajoutant les fonctionnalités implémentées à l'aide de Convai.

Figure 100 : Schéma des fonctionnalités disponibles avec Convai



Source : données de l'auteur

Avec la scène HospitalRoomConvai, le GameObject Convai remplace AvatarDoc2 par Convai NPC Harlene, et introduit plusieurs composants qui apportent une nouvelle dimension à Sim3D grâce à ce personnage intelligent. De plus, ces ajouts se complémentent aux fonctionnalités déjà implémentées sur la scène HospitalRoom, enrichissant alors le jeu de simulation.

Convai possède son propre système de reconnaissance vocale pour convertir les paroles du joueur en texte, lequel s'affiche sur une interface graphique dédiée, en plus d'être traité par une IA sur leur serveur, de manière similaire à ChatGPT pour le ScenarioManager et le patient. Cette intelligence se base sur les informations enregistrées du personnage sur la plateforme en ligne de Convai, ainsi que sur le graphe du Design narratif, ce dernier étant mis à jour en fonction du déroulement de la discussion. Tout ceci permet de préciser les réponses de l'infirmière et d'assurer leur cohérence. Le texte généré par l'IA est aussi affiché sur l'interface graphique, en plus d'être

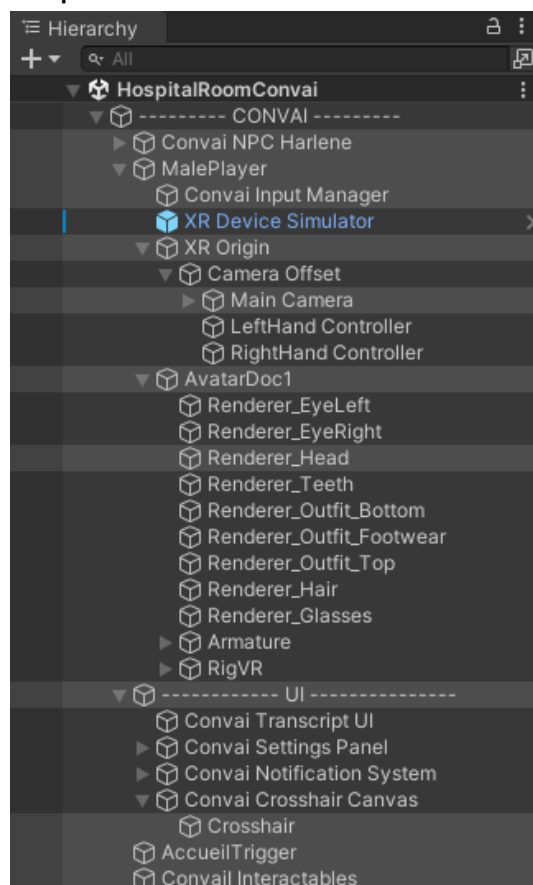
converti en parole grâce à une synthèse vocale, avec une synchronisation labiale pour l'avatar de l'infirmière. De plus, Convai permet de configurer le suivi du regard du personnage.

L'IA interprète également les actions à réaliser selon les paroles du joueur. Si elles correspondent à celles définies sur Unity, elles sont exécutées. Ces actions permettent le déplacement du personnage Convai ainsi que son animation. En indiquant les objets cibles à l'infirmière, cela lui permet dès lors d'interagir avec son environnement, créant ainsi une nouvelle expérience de jeu plus immersive. Ainsi, Harlene possède les bases d'un personnage proactif, démontrant des capacités qui enrichissent le jeu de simulation et contribue à le rendre plus interactif, le tout implémenté à l'aide d'une seule solution.

3.5. CONSTRUCTION DE LA SCÈNE

Ce dernier sous-chapitre détaille les GameObjects qui ont servi à la création du prototype. Pour rappel, la scène implémentée se nomme HospitalRoomConvai, et cette dernière est enregistrée sous « Assets > Scenes ». En figure 101 est présentée une première partie de la hiérarchie de la scène, consacrée aux éléments de Convai, les plus importants étant surlignés en gris clair. Ils sont tous regroupés sous un GameObject général nommé « ----- CONVAI ----- ».

Figure 101 : Première partie de la hiérarchie de la scène HospitalRoomConvai



Source : données de l'auteur

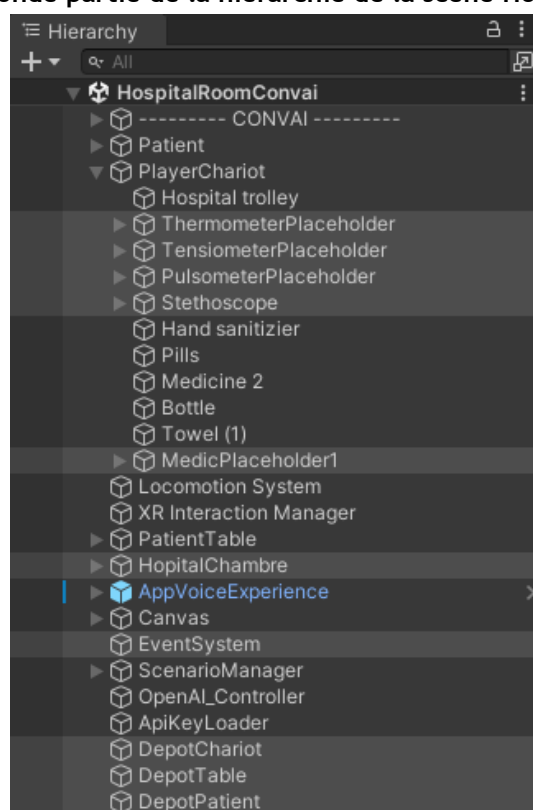
- Convai NPC Harlene : il s'agit du GameObject de l'infirmière, contenant l'avatar du personnage, ainsi que tous les scripts de fonctionnalités Convai implémentées au cours de ce travail de Bachelor ;
- MalePlayer : c'est le GameObject général du joueur ;
 - Convai Input Manager : nécessaire pour configurer les touches du joueur afin qu'il puisse activer le micro pour dialoguer avec le personnage Convai ;
 - XR Device Simulator : il permet de simuler l'utilisation du casque de réalité virtuelle dans la scène de jeu ;
 - XR Origin : un GameObject qui permet d'implémenter le fonctionnement du casque de réalité virtuelle en jeu ;
 - Main Camera : représente l'écran du joueur, il possède aussi les scripts qui permettent de détecter quel PNJ le joueur observe et d'effectuer l'échange avec le serveur Convai pour générer les réponses du personnage ;
 - AvatarDoc1 : représente l'avatar du joueur ;
 - Renderer_Head : contient les composants de collision pour activer le déclencheur du Design narratif du personnage ;
- « ----- UI ----- » : GameObject général qui implémente l'interface graphique de Convai ;
 - « Crosshair » : contient l'image du réticule du joueur, qui a été mis en transparent pour le cacher sans pour autant le désactiver ;
- AccueilTrigger : le déclencheur du Design narratif de l'infirmière Harlene ;
- Convai Interactables : GameObject qui permet de configurer les objets cibles pour les actions du personnage ;

La figure 102 représente la suite des GameObjects de la scène, sans qu'ils soient forcément liés à Convai. À nouveau, les plus importants sont surlignés en gris clair.

- PlayerChariot : GameObject général qui comprend tous les éléments du chariot du joueur ;
 - « ThermometerPlaceholder » : représente le thermomètre ;
 - « TensiometerPlaceholder » : représente le tensiomètre ;
 - « PulsometerPlaceholder » : représente le pulsomètre ;
 - « Stethoscope » : représente le stéthoscope ;
 - « MedicPlaceholder1 » : représente le médicament ;
- HopitalChambre : GameObject général qui comprend tous les éléments qui constituent le décor de la scène de jeu. Il possède le script NavMeshSurface pour définir la surface de

- navigation du personnage, tandis que ses GameObjects enfants qui en sont exclus contiennent le script NavMeshModifier ;
- EventSystem : GameObject déjà existant au développement de Sim3D, il incorpore le script ConvaiDynamicInputSystem de celui préfabriqué dans Convai Essentials ;
 - DepotChariot : GameObject de position pour indiquer où l'infirmière doit reposer les outils après leur utilisation ;
 - DepotTable : GameObject de position pour indiquer où l'infirmière doit poser le médicament une fois ramassé ;
 - DepotPatient : GameObject de position pour indiquer où l'infirmière doit utiliser les outils une fois saisis ;

Figure 102 : Seconde partie de la hiérarchie de la scène HospitalRoomConvai



Source : données de l'auteur

4. GESTION DU PROJET

Ce chapitre détaille les différentes phases de la gestion du projet, notamment le développement du prototype, qui a été conduit en suivant un développement agile avec des sprints. L'outil utilisé pour obtenir les différents graphes présentés dans ce chapitre est le même que celui employé lors du projet Sim3D : Microsoft Excel. En effet, un fichier Excel complet a été mis à disposition pour chaque groupe dans le cadre du module de cours 646-2 Projet Agile, afin de pouvoir documenter de manière claire le déroulement du travail. J'ai donc repris ce fichier et l'ai simplifié afin de l'adapter aux besoins d'un développement effectué seul.

4.1. PRÉPARATION

Cette partie a débuté le 20 mai 2024, date officielle du début du travail de Bachelor, pour se terminer le 16 juin 2024, veille de l'installation des logiciels pour le développement du prototype. Il s'agit de la phase la plus longue, et je ne la considère pas comme un sprint en tant que tel. Elle comprend plusieurs aspects préparatoires essentiels, dont l'analyse des exigences, la détermination des objectifs et des problématiques, ainsi que les recherches en ligne pour approfondir le sujet et identifier les outils nécessaires au développement. Cette période inclut également les entretiens avec mon professeur responsable au Technopôle, la construction de la matrice de décision, et le début de l'écriture du travail de Bachelor, comprenant alors les premières ébauches, les relectures et la préparation des chapitres. Le tout représente environ 62 heures de travail. Pour un détail du calcul de ces heures, mon journal de bord complet est disponible en annexe III.

4.2. SPRINT 0

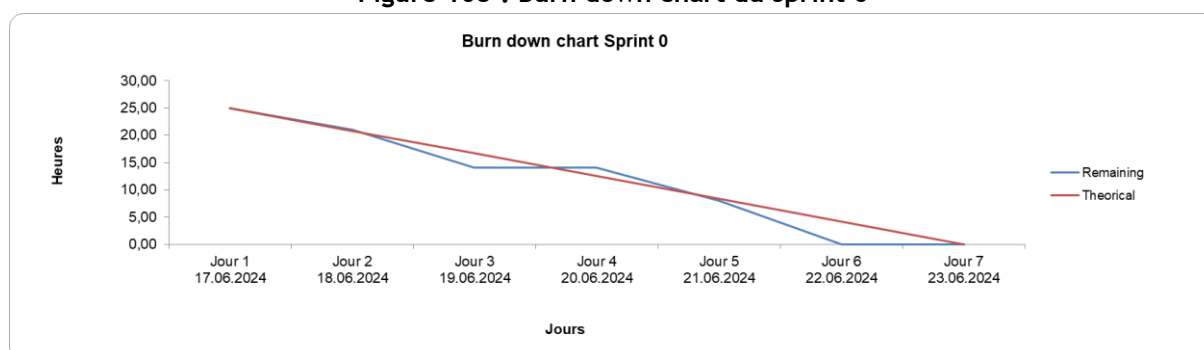
Ce premier sprint a démarré le 17 juin 2024 et s'est terminé le 23 juin 2024. L'objectif de cette semaine était de préparer l'environnement de travail pour pouvoir me mettre au développement du prototype. Cela comprend notamment l'installation de l'outil sélectionné par la matrice de décisions, qui s'est finalement avéré être Convai, et donc la mise à jour de la version d'Unity du projet Sim3D pour pouvoir l'utiliser.

Durant cette semaine, j'ai pu tester les plateformes en ligne d'Inworld et de Convai afin de pouvoir créer le personnage intelligent, tout en examinant comment installer les ML-Agents d'Unity sans devoir mettre à jour le projet Sim3D. J'étais encore en pleine recherche du meilleur outil à sélectionner, en observant cette fois leur utilisation pratique, élément qui n'est pas toujours évident à ressortir sur leurs pages officielles. Finalement, l'entretien avec le professeur Widmer le 19 juin 2024 m'a permis de sélectionner de manière définitive la solution Convai, surtout après avoir rencontré des difficultés avec la plateforme d'Inworld et avec l'installation des ML-Agents d'Unity. Cette entrevue m'a aussi aidé à rédiger le « Product Backlog » de manière plus précise.

Après l'installation de Convai sur le projet Sim3D avec sa nouvelle version d'Unity, un processus qui m'a pris un total de 2 heures environ, j'ai pu effectuer un test en exécutant le jeu en mode Play pour constater que l'environnement était prêt pour le développement. Une fois cela fait, je me suis penché sur la définition des « User Stories », soit la liste de fonctionnalités attendues pour le prototype. Ainsi, ce sprint 0 ne répond à aucune user story spécifique, mais ne m'a pas empêché pour autant de définir les tâches de préparations nécessaires pour accomplir mon objectif, afin d'assurer la suite du développement. Le total des heures estimées pour cette semaine de travail est d'environ 22 heures et 30 minutes.

La progression du travail réalisé est illustrée en figure 103. Ce graphique se nomme « Burn down chart » dans le développement agile. L'axe horizontal représente les jours du sprint, tandis que l'axe vertical concerne la quantité de travail estimée en heures. La diagonale rouge, labellisée « Theorical », montre une progression idéale, tandis que la ligne bleue, nommée « Remaining », correspond à la progression réelle. Le travail réalisé respecte alors cette courbe idéale, et se termine même un jour avant, ce qui m'a permis alors de me concentrer sur la rédaction du travail de Bachelor. L'annexe IV présente le détail complet de ce sprint, avec les estimations en heures de chacune des tâches réalisées durant cette semaine.

Figure 103 : Burn down chart du sprint 0



Source : données de l'auteur

4.2.1. PRODUCT BACKLOG

Le Product Backlog est présenté en annexe II de ce document. Il contient 10 User Stories, toutes écrites du point de vue du joueur, reprenant certaines problématiques mentionnées au chapitre 1.4 de ce rapport. Chacune décrit une fonctionnalité ou un comportement spécifique que le personnage intelligent, défini en tant qu'infirmière, devra démontrer en jeu. Les critères d'acceptance permettent de vérifier si les éléments développés dans le prototype répondent à la User Story.

Chaque User Story est attribuée à un sprint et est évaluée en termes de difficulté, nommée « Story points », en utilisant la suite de Fibonacci. La difficulté est estimée de manière croissante, avec les valeurs 1, 2, 3, 5, et 8. Comme il est possible de le constater dans l'annexe II, la User Story théoriquement la plus simple, évaluée à 1, demande à ce que l'infirmière donne une réponse audio

après que le joueur ait parlé au micro. En revanche, celles ayant le plus haut degré de difficulté, soit 8, attendent du personnage une certaine connaissance du jeu et du scénario, ainsi que la capacité d'utiliser les quatre outils du joueur sur le patient.

4.2.2. DEFINITION OF DONE

Pour qu'une User Story soit jugée comme validée, il est nécessaire de fournir les points à accomplir afin de pouvoir la considérer en tant que telle. Ces points sont regroupés sous la « Definition of Done ». Dans le cadre de ce travail de Bachelor, voici les éléments que j'ai définis pour valider une User Story :

- Le code est mis à jour sur la plateforme de développement en ligne « GitLab » ;
- La fonctionnalité est testée en mode Play sur Unity ;
- Les critères d'acceptance sont respectés ;
- La fonctionnalité est documentée dans le travail de Bachelor ;

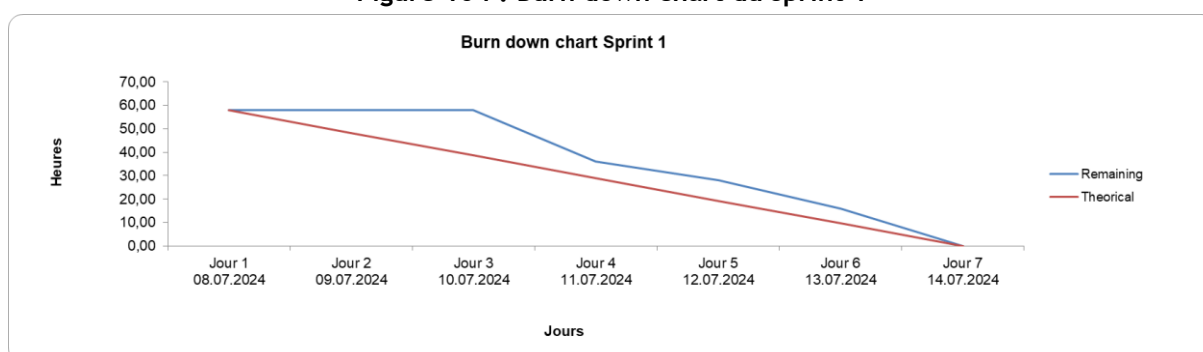
4.3. PAUSE INTER-SPRINT

Cette phase intervient entre le sprint 0 et le sprint 1. Elle comprend la période du 24 juin 2024 jusqu'au 7 juillet 2024, et a nécessité 47 heures et 30 minutes de travail. Comme le sprint précédent s'est terminé un jour plus tôt, j'ai en profité pour commencer à relire et corriger mon rapport, notamment sur les éléments déjà rédigés jusque-là, en particulier le chapitre sur les objectifs et problématiques ainsi que la description des solutions envisagées pour le prototype.

J'ai ensuite décidé de me concentrer sur l'écriture. J'ai alors ajouté le sous-chapitre sur Charisma.ai et commencé la rédaction sur la matrice de décisions et ses catégories. J'ai également documenté le processus de mise à jour d'Unity et l'utilisation de la plateforme en ligne de Convai, deux éléments que j'ai pu respectivement réaliser et tester durant le sprint 0. Cependant, la finalisation et la relecture de ces nouvelles sections ont empiété sur le début du sprint 1.

4.4. SPRINT 1

Figure 104 : Burn down chart du sprint 1



Source : données de l'auteur

Le sprint suivant a démarré le 8 juillet 2024 et s'est terminé le 14 juillet 2024. La figure 104 illustre la progression du travail réalisé. À cause de la relecture et de la finalisation des éléments rédigés lors de la pause inter-sprint, il n'y a pas eu de véritable avancée les trois premiers jours, ce qui a provoqué un retard. Cependant, il a pu être rattrapé et le travail a été achevé dans les délais, principalement grâce à une estimation assez large des tâches à réaliser. La durée effective de travail se monte approximativement à 22 heures et 30 minutes selon mon journal de bord

Le but du sprint était d'intégrer une infirmière capable d'initier une conversation et d'échanger. Les User Stories sélectionnées sont illustrées en figure 105, et en annexe V se trouve le détail complet du sprint. Les tâches incluaient l'importation du personnage créé sur Convai, l'intégration du suivi du regard et du mouvement des lèvres, et la récupération du Design narratif créé sur la plateforme Convai pour permettre à l'infirmière de suivre le scénario, en plus de paramétrer les déclencheurs dans la scène. J'ai aussi effectué plusieurs échanges avec le personnage afin de pouvoir corriger la synchronisation labiale fournie par Convai et de vérifier si la description et le graphe narratif sont respectés par l'infirmière. Ses réponses ont aussi été testées afin de pouvoir enregistrer sous la transcription figurant en annexe VII de ce rapport un exemple de dialogue complet en ayant suivi la totalité du scénario du jeu.

Figure 105 : User stories sélectionnées pour le sprint 1

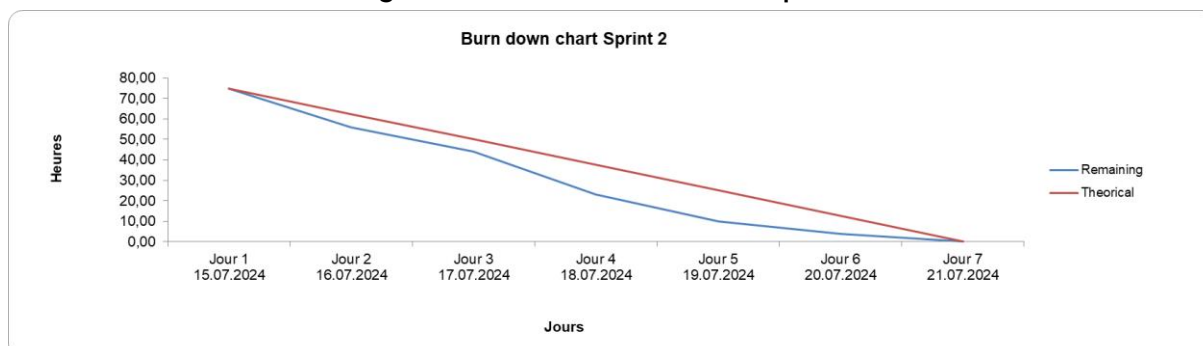
US Nr.	En tant que	je veux	afin de	Critères d'acceptance	Story points
1	Joueur	Une infirmière qui donne une réponse quand on lui parle	Pouvoir échanger avec elle pour une expérience de jeu interactive	Après que le joueur s'est exprimé au micro, le personnage fournit une réponse en accord avec ce qui a été dit	1
2	Joueur	Une infirmière qui observe le joueur et anime ses lèvres durant la conversation	Rendre l'échange plus réaliste et naturel	Le personnage regarde le joueur tout au long de l'échange et ses lèvres s'animent à chaque réponse donnée	2
3	Joueur	Une infirmière qui possède une mémoire de mes actions réalisées durant la partie	Créer une immersion et rendre le personnage humain	Si le joueur donne son nom en début de jeu, à la fin de la partie le personnage doit toujours savoir comment le joueur s'appelle	3
4	Joueur	Une infirmière qui connaisse le scénario du jeu et ne sort pas de son contexte durant l'échange	Maintenir l'immersion du jeu	Le personnage doit garder son rôle d'infirmière tout le long de la partie, et les réponses données sont en lien avec l'objectif actuel du jeu	8
5	Joueur	Une infirmière qui initie l'échange dans certaines situations	Rendre l'expérience de jeu plus dynamique et naturelle	Le personnage démarre la conversation sans intervention du joueur	3

Source : données de l'auteur

L'étape la plus chronophage a été l'implémentation de l'interface Convai sur la scène de Sim3D, car c'est cet élément qui a débloqué aux personnages intelligents la capacité d'échanger avec le joueur. Je pense notamment à la partie 3.2.4 concernant les références de TextMesh Pro, une erreur qui n'avait pas fait surface au sprint 0 car je n'y ai que vérifié si le mode Play d'Unity se lançait, sans chercher à tester les fonctionnalités de Convai. Ces corrections sont survenues juste après l'importation du personnage dans la scène. Elles sont donc comprises avec dans le journal de bord et s'élèvent alors à un total de 4 heures et 30 minutes. Les autres étapes du sprint se sont déroulées sans trop de difficultés, si ce n'est la documentation de ces éléments qui me prend toujours du temps.

4.5. SPRINT 2

Figure 106 : Burn down chart du sprint 2



Source : données de l'auteur

Le sprint 2 a commencé le 15 juillet 2024 pour se terminer le 21 juillet 2024. À nouveau, le graphique de la progression du travail est présenté en figure 106. Contrairement au premier sprint, j'ai pu commencer à travailler sur le prototype dès les premiers jours. Grâce à une meilleure compréhension des outils de Convaï lors la semaine précédente, j'ai sélectionné alors toutes les User Stories restantes du Product Backlog, et j'ai également pu progresser assez rapidement dans les tâches à réaliser. Cela m'a permis d'enregistrer un premier aperçu du prototype que je pouvais montrer à mon professeur la semaine suivante, en plus de pouvoir tester toutes les fonctionnalités implémentées.

Le but du sprint était d'obtenir une infirmière prête à entreprendre des actions permettant de réussir les objectifs du joueur dans Sim3D. Comme l'indique la figure 107, les cinq User Stories restantes du Product Backlog qui ont été attribuées à ce sprint possèdent un total de 26 Story Points, contrairement à la somme de 17 pour le sprint 1. De plus, l'annexe VI fournit le détail complet de la réalisation de cette semaine de travail, avec les tâches et les estimations en heures.

Figure 107 : User stories sélectionnées pour le sprint 2

US Nr.	En tant que	je veux	afin de	Critères d'acceptance	Story points
6	Joueur	Une infirmière qui se déplace dans la chambre	Permettre au personnage de devenir un acteur du jeu	Le personnage évite les éléments du décor le plus possible dans ses déplacements	5
7	Joueur	Une infirmière qui joue des animations appropriées à chaque action effectuée	Maintenir une cohérence visuelle	Une animation est jouée pour chaque action réalisée (marcher, ramasser/poser, utiliser)	3
8	Joueur	Une infirmière qui saisit un objet quand l'ordre lui est donné	Permettre au personnage de réaliser des étapes du jeu	Le personnage prend l'objet qui a été demandé dans la main	5
9	Joueur	Une infirmière qui pose le médicament sur la table du patient	Réaliser l'objectif de l'administration du médicament de manière indirecte	Le personnage dépose l'objet à l'endroit approprié	5
10	Joueur	Une infirmière qui utilise un outil sur le patient à l'endroit approprié	Réaliser l'objectif de la prise des mesures de manière indirecte	Chaque outil est utilisé dans la zone appropriée, proche du patient	8

Source : données de l'auteur

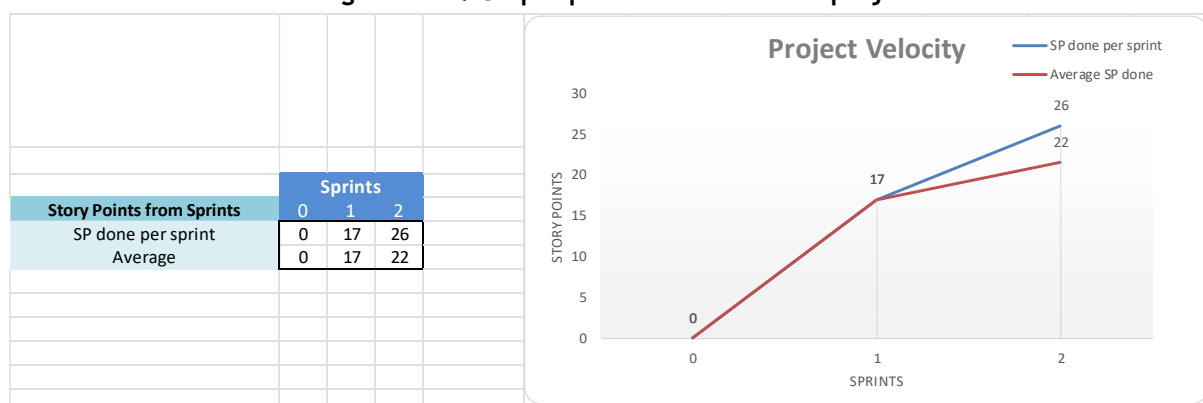
Les principales tâches comprenaient l'intégration du déplacement du personnage dans la scène, le paramétrage des animations, la saisie des objets par l'infirmière, et la création de différentes actions permettant au PNJ de simuler l'utilisation des divers outils, sans oublier la documentation nécessaire pour ces différentes tâches. L'élément le plus chronophage a été l'implémentation des

actions du personnage. Non seulement l'infirmière n'interprète pas toujours correctement les ordres qui lui sont donnés, mais ses actions ne sont pas générées par l'IA. Elles sont construites manuellement, et j'ai dû en créer de nouvelles, basées sur une compréhension du code fourni par Convai dans le script ConvaiActionsHandler. Cela m'a pris un total d'environ 4 heures et 30 minutes.

De plus, j'ai également consacré du temps à tester la position des zones de dépôt ou d'utilisation des objets, afin que l'infirmière se déplace précisément aux endroits souhaités et pour contribuer ainsi à un rendu plus réaliste pour le joueur. Ces paramétrages de positions, combinés avec l'implémentation de la navigation dans la scène de jeu, m'ont pris un total de 3 heures de travail. Au total, j'ai consacré approximativement 20 heures pour ce sprint. À nouveau, pour un détail du calcul du temps effectif réalisé à chaque phase, mon journal de bord complet est disponible en annexe III.

4.6. VÉLOCITÉ

Figure 108 : Graphique de la vélocité du projet



Source : données de l'auteur

Selon la figure 108, le graphique de la vélocité montre une augmentation continue des Story Points complétées par sprint. Bien que le sprint 0 ne compte aucun Story Point, le sprint 1 en totalise 17 et le sprint 2 s'élève à 26 Story Points. Cette progression indique une amélioration de l'efficacité et de la productivité au fil des sprints. La moyenne, calculée à partir du sprint 1, est de 22 Story Points à la fin du sprint 2, ce qui donne une indication du volume de travail que je peux accomplir. De plus, cette tendance positive reflète une montée en compétence et une meilleure maîtrise de l'outil Convai.

4.7. FINALISATION

Cette dernière partie de la gestion de projet s'étend du 22 juillet 2024 jusqu'au 16 août 2024, date de la remise du travail de Bachelor. Elle inclut principalement la finalisation de la rédaction de ce document et ne fait pas partie d'un sprint spécifique.

Lors de mon dernier entretien avec le professeur Widmer le 23 juillet 2024, je lui ai présenté les enregistrements démontrant un exemple de déroulement de mon prototype, obtenu à la suite des différents sprints réalisés jusque-là. En voyant que le résultat montré l'a satisfait, il a été convenu que le développement du prototype pouvait être considéré comme achevé et que je devais désormais me concentrer sur la documentation de mon travail.

Ainsi, cette période a principalement été consacrée à la rédaction des derniers chapitres de bilan, de conclusion, et de la fin de la gestion de projet ainsi qu'à la relecture de l'entièreté du rapport. Au total, j'ai investi environ 56 heures et 30 minutes pour cette phase. Les détails spécifiques des tâches réalisées sont disponibles dans mon journal de bord en annexe III. Cette finalisation a permis d'assurer que mon travail réponde aux exigences académiques de la HES-SO avant sa remise le 16 août 2024.

CONCLUSION

Ce travail de Bachelor a démontré l'intégration d'une solution propriétaire, Convai, sur un moteur de jeu particulier, Unity, dans le cadre d'un scénario simple mais spécifique, le personnage de l'infirmière dans la simulation d'hôpital de Sim3D. Bien que Convai ait été la solution retenue, il est important de souligner que les divers outils présentés dans la matrice de décisions auraient pu mener à un développement très différent du personnage. En effet, chacune de ces solutions offre des approches uniques pour la création de PNJs intelligents, et Convai n'est qu'une des nombreuses manières employées pour aborder ce thème. De plus, le prototype a pu préciser que l'intelligence artificielle que propose Convai englobe tout ce qui concerne l'échange, et ne génère donc pas d'actions, d'animations ou de déplacements pour le PNJ, ces éléments étant construits à la main.

Il est également essentiel de mentionner que l'utilisation de Convai nécessite une connexion internet, sans laquelle l'échange avec le personnage ne fonctionne pas. Cela peut être une limitation dans certains contextes, mais elle est aussi représentative des nombreuses solutions basées sur l'intelligence artificielle. Ce travail montre que le recours à un outil propriétaire peut s'avérer crucial, mais il en résulte alors une dépendance vis-à-vis de l'entreprise éditrice et de ses décisions, ce qui peut potentiellement impacter le développement futur. De plus, comme toute IA, Convai n'est pas exempte d'erreurs. Cela rappelle la nature perfectible de ces technologies, y compris des modèles comme ChatGPT.

La gestion de ce projet a révélé une distribution du temps largement orientée vers l'écriture plutôt que vers le développement du prototype. Ce déséquilibre met en évidence ma difficulté avec la rédaction, mais rappelle également que l'évaluation de ce travail de Bachelor se concentre avant tout sur le rapport final, comme mon professeur me l'a rappelé à plusieurs reprises. Bien que j'aie pu implémenter la plupart des fonctionnalités offertes par Convai dans le projet Sim3D, je regrette toutefois d'avoir passé plus de temps à rédiger plutôt qu'à programmer sur Unity. Ainsi, les autres éléments proposés par Convai, tels que les interactions entre plusieurs personnages intelligents, ou l'expérimentation de différentes personnalités pour l'infirmière, n'ont pas pu être abordés.

En conclusion, l'implémentation de Convai sur Sim3D a démontré la viabilité ainsi que les défis de l'intégration d'un tel outil dans un projet déjà riche en fonctionnalités. Le personnage obtenu montre des capacités immersives qui, même si elles restent perfectibles, peuvent mener à des expériences de jeu uniques. Cela m'a offert une nouvelle expérience sur Unity, un logiciel que je n'aurais probablement pas vu durant mon cursus à la HES-SO sans l'opportunité offerte par Sim3D et par ce travail de Bachelor. Pour finir, afin de poursuivre le développement, la prochaine étape serait d'explorer les fonctionnalités de Convai qui n'ont pas pu être abordées dans ce travail, avec des tests dans d'autres types de scénarios de jeu pour observer davantage ses capacités et ses limites, et ainsi déterminer si son utilisation est pertinente pour les équipes de recherche.

DÉCLARATION DE L'AUTEUR

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

- Monsieur Antoine Widmer
- Monsieur Jeff Zufferey

Sierre, le 16 août 2024

Bastien Salamin

Signature :



BIBLIOGRAPHIE

- BehaviorTree.CPP. (s. d.). *Introduction to BTs*. Consulté le 5 juin 2024, sur https://www.behaviortree.dev/docs/learn-the-basics/BT_basics
- Charisma.ai. (2024, 29 mai). *Unity Plug 'n' Play SDK Docs - Charisma Docs*. Récupéré sur <https://charisma.ai/docs/plugin-play/unity>
- Charisma.ai. (2024, 7 juin). *Licensing Information - Charisma Docs*. Récupéré sur <https://charisma.ai/docs/licencing>
- Convai. (s. d.). *Pricing*. Consulté le 8 juin 2024, sur <https://convai.com/pricing>
- Convai. (2024, 10 juin). *Make 3D AI Characters Act on Your Command | Scene Aware Characters | Convai Unity Tutorial* [Vidéo]. Récupéré sur <https://www.youtube.com/watch?v=rjZ7VdZhoTU>
- Dawe, M., Gargolinski, S., Dicken, L., Humphreys, T., & Mark, D. (2013, septembre). Behavior Selection Algorithms: An Overview. *Game AI Pro: Collected Wisdom of Game AI Professionals*. 47-60.
- Gillis, A. S., Brush, K., Scardina, J. (2024, mai). What is a chatbot?. *TechTarget Customer Experience*. Récupéré sur <https://www.techtarget.com/searchcustomerexperience/definition/chatbot>
- Grail. (s. d.). *Grail Overview :: Grail Documentation*. Consulté le 10 juin 2024, sur <https://grail.com.pl/documentation/documentation/1.3.0/index.html>
- Grail. (s. d.). *Licensing Grail*. Consulté le 10 juin 2024, sur <https://grail.com.pl/licensing.html>
- Inworld AI. (s. d.). *Frequently Asked Questions*. Consulté le 9 juin 2024, sur <https://docs.inworld.ai/docs/tutorial-basics/faq/>
- Inworld AI. (s. d.). *Licensing options*. Consulté le 9 juin 2024, sur <https://inworld.ai/license>
- Juliani, A., Berges, V. P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H. Mattar, M. & Lange, D. (2020, 6 mai). Unity: A general platform for intelligent agents. *arXiv e-Print archive*. DOI:10.48550/arXiv.1809.02627.
- LlamAcademy. (2021, 9 février). *NavMesh Basics - Introduction to the NavMeshSurface | AI Series Part 1 | Unity Tutorial* [Vidéo]. Récupéré sur <https://www.youtube.com/watch?v=aHFSDcEQzQ>
- lucbloom. (2020, 27 mars). Actual solution: Delete your project's /Library/ folder! I wasted too much time before trying this. (Warning: re-importing your project might [Commentaire sur le post du forum en ligne TMP Input Field "No Monobehaviour Scripts"]. Unity Forum. Récupéré sur <https://forum.unity.com/threads/tmp-input-field-no-monobehaviour-scripts.1541540/>
- Mukherjee, P. (2023, 20 août). Introducing Convai [Publication sur blog]. Récupéré sur <https://convai.com/blog/introducing-convai>
- Nigretti, A. (2017, 11 décembre). Using Machine Learning Agents Toolkit in a real game: A beginner's guide [Publication sur blog]. Récupéré sur <https://blog.unity.com/games/using-machine-learning-agents-in-a-real-game-a-beginners-guide>

- Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023, octobre). Generative agents: Interactive simulacra of human behavior. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*.1-22. DOI:10.1145/3586183.3606763.
- Rabin, S. (2017, juin). The Illusion of Intelligence. *Game AI Pro 3: Collected Wisdom of Game AI Professionals*. 3-9.
- Ready Player Me. (2023). [Rendu de l'avatar de l'infirmière après sa création] [Image digitale]. Récupéré sur <https://readyplayer.me/fr/hub/avatars>
- Total AI. (s. d.). *Total AI Guide: Getting Started - Introduction*. Consulté le 12 juin 2024, sur <http://totalai.org/doc-introduction.html>
- Total AI. (2021, 27 janvier). *Total AI Install & Setup* [Vidéo]. Récupéré sur <https://www.youtube.com/watch?v=XsqUmfCPf7M>
- Sarge. (2023, 26 avril). Face Animation with Generated Audio using Oculus LipSync! [Vidéo]. Récupéré sur <https://www.youtube.com/watch?v=Q4sPGTVylnY>
- Sissler, J. (2024, mai). Enhancing Non-player Characters in Unity 3D using GPT-3.5. *ACM Games Just Accepted*. DOI:10.1145/3662003.
- Unity Asset Store. (2018, 25 avril). *IBRAIN - NPC Character A.I.* Récupéré sur <https://assetstore.unity.com/packages/tools/behavior-ai/ibrain-npc-character-a-i-116128>
- Unity Asset Store. (2022, 21 septembre). *AI NPC Engine v3 - Dialogue & Behavior for Unity - Inworld*. Récupéré sur <https://assetstore.unity.com/packages/tools/behavior-ai/ai-npc-engine-v3-dialogue-behavior-for-unity-inworld-229406>
- Unity Asset Store. (2023, 13 avril). *NPC AI Engine - Dialog, actions, voice and lipsync - Convai*. Récupéré sur <https://assetstore.unity.com/packages/tools/behavior-ai/npc-ai-engine-dialog-actions-voice-and-lipsync-convai-235621>
- Unity Asset Store. (2023, 26 juin). *Grail AI Framework Tiered Pricing*. Récupéré sur <https://assetstore.unity.com/packages/tools/behavior-ai/grail-ai-framework-tiered-pricing-258038>
- Unity Asset Store. (2023, 15 décembre). *Ready Player Me Avatar and Character Creator*. Récupéré sur <https://assetstore.unity.com/packages/tools/game-toolkits/ready-player-me-avatar-and-character-creator-259814>
- Unity Technologies (2022, 2 décembre). *Example Learning Environments - Unity ML-Agents Toolkit*. Récupéré sur <https://unity-technologies.github.io/ml-agents/Learning-Environment-Examples/>
- Valecillos, D. (2020, 1 décembre). *How To Add The XR Device Simulator To Speed Up Development | Unity XR Toolkit* [Vidéo]. Récupéré sur <https://www.youtube.com/watch?v=d4bTpkvBwrs>

Annexe I : Donnée du Travail de Bachelor

HES-SO Valais

FEE	FIG	FTO
	X	

Information regarding Bachelor's thesis

FO.2.2.02.28.EC
mob/30/10/2017

Degree programme:

BUSINESS INFORMATION TECHNOLOGY

Confidential

☐

Student SURNAME: Name: SALAMIN Bastien Mobile: +41 79 333 76 58	Year 2024
Submitted by: DiverSSity SA Professor: Antoine Widmer	Language <input checked="" type="checkbox"/> French <input type="checkbox"/> German <input checked="" type="checkbox"/> English

Title: Smart Virtual Characters using AI

Description:

Context (company research project, etc.)

DiverSSity SA is a Sion-based start-up that develops **therapeutic Mixed Reality (MR) exercises** for people with neurodevelopmental disorders. These exercises use **Artificial Intelligence** to create immersive and adaptive scenarios. DiverSSity is collaborating with the **institute of informatics** on an **Innosuisse project** to enhance the MR exercises. A key part of the project is to train social anxiety. For this training, we need to develop **Virtual Characters** able to move freely and interact in a natural fashion with real users. These virtual characters need to follow rules assigned to them and act accordingly with real users. Therefore, generative AI algorithms are needed to drive these virtual characters.



What concrete issues does the project deal with?

In this project we are interested in investigating how to develop AI algorithms able to drive interaction between real users and virtual characters in Virtual and Mixed Reality simulations. To achieve this, the current work will explore the state-of-art in the development of smart virtual characters and develop a working prototype testable in a relevant scenario.

HES-SO Valais

FEE	FIG	FTO
	X	

Information regarding Bachelor's thesis

FO.2.2.02.28.EC
mob/30/10/2017

Planned project stages:

Analysis or state of the art

- State-of-Art on AI for Non-Playable Characters.
- Analysis of existing requirements for the creation of non-playable intelligent characters. (animation, pathfinding, behaviour, attachment to context, etc.)
- Analysis of existing tools to create smart Non-Playable Characters.

Decision and justification by the student

- Elaboration of a product backlog
- Choice and justification of the technology used to solve the problem
- Elaboration of a prototype using AI for Non-Playable Characters.

Implementation & Testing

- Implementation of the prototype.
- Perform test on 1 person

Available resources

Data

(Specify the data type, the data format and the metrics used)

Data about how the NCP should behave will be given by DiverSSity SA at the start of the project.

Use case

(Describe the use case)

This project will lay down the foundation to build a complete set of therapeutic MR exercises for people with neurodevelopmental disorders.

Hardware & resources

(Specify the resources made available and their environment)

Hardware available for this work:

Meta quest 3; Gaming PC; product owner from the company

Comments by the professor:

Signature

Head degree programme Business Information Tech.

Professor:

Student:

Key dates

Start :

20.05.2024

Due date of report:

16.08.2024 12.00

Public exhibition of Bachelor's theses:

To be defined

Annexe II : Product Backlog

US Nr.	En tant que je veux	afin de	Critères d'acceptance	Story points	Sprint
1	Joueur Une infirmière qui donne une réponse quand on lui parle	Pouvoir échanger avec elle pour une expérience de jeu interactive	Après que le joueur s'est exprimé au micro, le personnage fournit une réponse en accord avec ce qui a été dit	1	1
2	Joueur Une infirmière qui observe le joueur et anime ses lèvres durant la conversation	Rendre l'échange plus réaliste et naturel	Le personnage regarde le joueur tout au long de l'échange et ses lèvres s'animent à chaque réponse donnée	2	1
3	Joueur Une infirmière qui possède une mémoire de mes actions réalisées durant la partie	Créer une immersion et rendre le personnage humain	Si le joueur donne son nom en début de jeu, à la fin de la partie le personnage doit toujours savoir comment le joueur s'appelle	3	1
4	Joueur Une infirmière qui connaisse le scénario du jeu et ne sort pas de son contexte durant l'échange	Maintenir l'immersion du jeu	Le personnage doit garder son rôle d'infirmière tout le long de la partie, et les réponses données sont en lien avec l'objectif actuel du jeu	8	1
5	Joueur Une infirmière qui initie l'échange dans certaines situations	Rendre l'expérience de jeu plus dynamique et naturelle	Le personnage démarre la conversation sans intervention du joueur	3	1
6	Joueur Une infirmière qui se déplace dans la chambre	Permettre au personnage de devenir un acteur du jeu	Le personnage évite les éléments du décor le plus possible dans ses déplacements	5	2
7	Joueur Une infirmière qui joue des animations appropriées à chaque action effectuée	Maintenir une cohérence visuelle	Une animation est jouée pour chaque action réalisée (marcher, ramasser/poser, utiliser)	3	2
8	Joueur Une infirmière qui saisit un objet quand l'ordre lui est donné	Permettre au personnage de réaliser des étapes du jeu	Le personnage prend l'objet qui a été demandé dans la main	5	2
9	Joueur Une infirmière qui pose le médicament sur la table du patient	Réaliser l'objectif de l'administration du médicament de manière indirecte	Le personnage dépose l'objet à l'endroit approprié	5	2
10	Joueur Une infirmière qui utilise un outil sur le patient à l'endroit approprié	Réaliser l'objectif de la prise des mesures de manière indirecte	Chaque outil est utilisé dans la zone appropriée, proche du patient	8	2

Annexe III : Journal de bord

Date	Quoi	Temps		
23.05.2024	Premier RDV avec le professeur Widmer pour débiter le bachelor	01:00:00		
24.05.2024	Cours de méthodologie N°1	03:00:00		
24.05.2024	Premières recherches sur l'IA des PNJs sur Unity et création du Word du TB	02:30:00		
27.05.2024	Avancées des recherches et rédaction d'un fichier Word sur les objectifs et problématiques	03:00:00		
28.05.2024	Corrections du fichier Word sur les objectifs et problématiques pour le RDV du jour suivant	01:00:00		
29.05.2024	RDV au Technopôle pour un retour sur les objectifs/problématiques et définition de la suite	01:00:00		
29.05.2024	Envoi du document des données du TB signé et mises au propre des notes prises au Technopôle	02:00:00		
30.05.2024	Élaboration du sommaire, construction de la matrice de décisions et recherche de documents scientifiques	02:30:00		
31.05.2024	Tentative d'écriture du chapitre 1 "Introduction" du TB	02:00:00		
01.06.2024	Avancement dans l'écriture du chapitre 1 du TB	03:00:00		
02.06.2024	Recherches sur la distinction entre IA d'un jeu et IA d'un LLM, avec début de bibliographie	01:00:00		
02.06.2024	Finalisation du chapitre 1	01:00:00		
03.06.2024	Lecture de littératures pour commencer et avancer l'écriture du chapitre 2	04:30:00		
04.06.2024	Avancement dans l'écriture du chapitre 2 du TB	02:30:00		
05.06.2024	Avancement dans l'écriture du chapitre 2 du TB	03:00:00		
06.06.2024	Avancement dans l'écriture du chapitre 2 du TB	02:30:00		
06.06.2024	Installation d'Unity et du projet Sim3D	01:00:00		
07.06.2024	Finalisation du chapitre 2	01:30:00		
07.06.2024	Lecture de littératures scientifiques et corrections bibliographie	01:30:00		
08.06.2024	Début d'écriture du chapitre 3 et lecture des spécificités de Convai	03:00:00		
08.06.2024	Modification de la matrice de décisions et première attributions de points pour Convai	00:30:00		
09.06.2024	Avancement du chapitre 3 avec lecture des spécificités d'Inworld	03:30:00		
10.06.2024	Avancement du chapitre 3 avec lecture des spécificités de Grail	02:00:00		
11.06.2024	Avancement du chapitre 3 avec lecture des spécificités des agents Unity	03:00:00		
12.06.2024	Avancement du chapitre 3 avec lecture des spécificités de Total AI	02:00:00		
12.06.2024	Avancement du chapitre 3 avec une première lecture des spécificités de IBRAIN	00:30:00		
13.06.2024	Avancement du chapitre 3 avec finalisation du sous-chapitre IBRAIN	01:30:00		
14.06.2024	RDV au Technopôle pour vérification de la matrice de décisions et mise en forme du document du TB	02:00:00		
15.06.2024	Mise au propre des notes prises la veille, corrections des pages du TB et de la matrice de décisions	02:30:00		
16.06.2024	Finalisation de la matrice de décisions, corrections de la bibliographie et début du Product Backlog	02:00:00		
			Préparation	62:00:00
16.06.2024	Création d'un fork du projet Sim3D	01:00:00		
16.06.2024	Recherche sur l'ancienne sortie des ML-Agents d'Unity à utiliser avec la version de Sim3D	00:30:00		
17.06.2024	Ecriture des premières User-stories	01:00:00		
17.06.2024	Inscription sur Inworld et recherches des fonctionnalités possibles avec cette solution	00:30:00		
17.06.2024	Constatation qu'Inworld ne semble pas si approprié que ça et ressemble plus à un Chatbot anglais	01:00:00		
17.06.2024	Inscription sur Convai afin de faire un premier test et vérifier comment changer la matrice de décisions	01:00:00		
17.06.2024	Remaniement de la matrice de décisions, vérification de la solution Charisma.ai et modification des US	01:30:00		
18.06.2024	Finalisation de la matrice de décisions et du Product Backlog	01:00:00		
18.06.2024	Construction d'un premier graphique de design narratif sur Convai	01:00:00		
18.06.2024	Tentatives d'installation de Convai (raté, erreur de version ?) et Unity ML-Agents (réussi après un moment)	02:30:00		
19.06.2024	Corrections de la matrice de décision et du Product Backlog	00:30:00		
19.06.2024	Poursuite de l'installation d'Unity ML-Agents	02:00:00		
19.06.2024	RDV au Technopôle pour choix de la solution (Convai) et corrections à effectuer	01:00:00		
19.06.2024	Installation de la version d'Unity 2022.3.33f1 et de Convai ainsi que nettoyage des dossiers locaux	02:00:00		
20.06.2024	Finalisation de la matrice de décisions, avancée du Product Backlog et définition of done	01:30:00		
20.06.2024	Initialisation du backup de Sim3D pour prendre les captures d'écran et listes des points à écrire sur le TB	01:00:00		
21.06.2024	Repréparation du git pour qu'il soit propre et vérifier si les étapes écrites dans le TB sont correctes	02:00:00		
21.06.2024	Test du jeu Sim3D avec mise à jour de la version Unity et XR Device Simulator pour vérifier si tout marche	01:30:00		
			Sprint 0	22:30:00
21.06.2024	Préparation des chapitres du TB après les précédents RDVs ainsi que des étapes pour le développement	00:30:00		
22.06.2024	Modification du sommaire du TB et écriture de l'avant-propos	02:00:00		
23.06.2024	Finalisation avant-propos et corrections de l'introduction	02:30:00		
24.06.2024	Ajout d'un sous-chapitre sur les chatbots au chapitre 2	02:00:00		
25.06.2024	Ajout d'un sous-chapitre sur Unity et Ready Player Me au chapitre 2, en plus de corriger la partie Scénario	03:30:00		
26.06.2024	Finalisation du chapitre 2	01:30:00		
27.06.2024	Relecture du chapitre 2 et début de la correction du chapitre 3 avec Convai	02:00:00		
28.06.2024	Corrections du paragraphe sur Inworld, relecture du chapitre 3 et recherches pour Charisma.ai	02:30:00		
29.06.2024	Relecture du chapitre 3 et écriture du sous-chapitre Charisma.ai	02:00:00		
30.06.2024	Finalisation du paragraphe sur Charisma.ai et début d'écriture sur la matrice de décisions	04:00:00		
01.07.2024	Ecriture des catégories concernant la matrice de décisions	02:30:00		
02.07.2024	Finalisation chapitre 3	02:30:00		
03.07.2024	Relecture chapitre 3	01:00:00		
03.07.2024	RDV à Bellevue pour un point sur l'état actuel et la suite, avec mise par écrit des points abordés après le RDV	01:00:00		
04.07.2024	Début d'écriture du chapitre 4 sur la plateforme Convai	03:00:00		
05.07.2024	Avancée de la rédaction sur la plateforme Convai	01:30:00		
06.07.2024	Avancée de la rédaction sur la plateforme Convai	03:00:00		
07.07.2024	Finalisation du sous-chapitre sur la plateforme Convai	02:30:00		
08.07.2024	Rédaction sur le processus de mise à jour Unity du projet Sim3D	03:30:00		
09.07.2024	Rédaction sur le processus de mise à jour Unity du projet Sim3D	02:30:00		
10.07.2024	Finalisation de la rédaction sur le processus de mise à jour Unity du projet Sim3D	00:30:00		
10.07.2024	Préparation des pages de Sprint du fichier Excel	01:30:00		
			Pause intersprint	
11.07.2024	Corrections des éléments de Convai, et importation de l'infirmière avec préparation de la scène	04:30:00		47:30:00

11.07.2024	Corrections des US du Sprint 1	01:00:00		
12.07.2024	Documentation de deux US du Sprint 1	04:30:00		
13.07.2024	Finalisation de la documentation de deux US du Sprint 1	01:00:00		
13.07.2024	Implémentation du Design narratif sur Unity	01:30:00		
13.07.2024	Test d'un échange avec l'infirmière et transcription du dialogue réalisé en annexe	01:00:00		
13.07.2024	Mise à jour de l'Excel de développement agile et des chapitre du TB	01:30:00		
14.07.2024	Documentation des dernières US du Sprint 1	04:00:00		
15.07.2024	Ecriture de la partie Démarrage de la gestion de projet	02:30:00		
16.07.2024	Ecriture de la partie Sprint 1 de la gestion de projet	01:00:00		
			Sprint 1	22:30:00
16.07.2024	Début d'implémentation des actions pour le personnage Convai	03:00:00		
16.07.2024	Recherche et implémentation de la navigation pour le personnage et la scène	01:00:00		
17.07.2024	Création de méthodes personnalisées pour forcer l'infirmière à faire ce que l'on attend d'elle	01:30:00		
17.07.2024	Vérifier si Mixamo possède des animations utiles	01:00:00		
17.07.2024	Ajustement de la position des objets pour le déplacement de l'infirmière	02:00:00		
18.07.2024	Documentation de deux US du Sprint 2	04:00:00		
19.07.2024	Test d'une première capture vidéo par le biais de Teams	02:00:00		
19.07.2024	Avancée dans la documentation des US	01:00:00		
20.07.2024	Avancée dans la documentation des US	02:00:00		
22.07.2024	Finalisation de la documentation des US du Sprint 2	01:30:00		
22.07.2024	Ecriture de la partie Sprint 2 de la gestion de projet	01:00:00		
			Sprint 2	20:00:00
23.07.2024	Préparation du RDV de l'après-midi	01:00:00		
23.07.2024	RDV à Bellevue pour un point sur l'état actuel et la suite, avec mise par écrit des points abordés après le RDV	01:00:00		
24.07.2024	Mise au propre du Product Backlog	04:30:00		
25.07.2024	Rédaction de la partie Gestion du projet du travail de Bachelor	03:30:00		
26.07.2024	Finalisation de la partie rédigée la veille	02:30:00		
26.07.2024	Ecriture des dernières parties sur le prototype	01:30:00		
27.07.2024	Ecriture des dernières parties sur le prototype	03:00:00		
27.07.2024	Construction des schémas pour illustrer la différence entre Sim3D et le prototype	01:00:00		
28.07.2024	Ecriture de la différence entre Sim3D et le prototype, et description de la hiérarchie de la scène	03:00:00		
29.07.2024	Finalisation du chapitre sur le développement du prototype et de la gestion de projet	02:30:00		
30.07.2024	Rédaction de la conclusion du travail de Bachelor	02:30:00		
30.07.2024	Tentative d'écriture du résumé du travail de Bachelor	01:00:00		
31.07.2024	Rédaction du résumé du travail de Bachelor	02:00:00		
01.08.2024	Finalisation de la conclusion et du résumé	00:30:00		
01.08.2024	Création de l'image en page de garde	00:30:00		
01.08.2024	Début de la création du poster	01:00:00		
02.08.2024	Finalisation du poster	02:00:00		
02.08.2024	Relecture des directives du travail de Bachelor ainsi que le guide de présentation des travaux écrits	00:30:00		
03.08.2024	Relecture du travail de Bachelor	03:30:00		
04.08.2024	Relecture du travail de Bachelor	04:00:00		
05.08.2024	Relecture du travail de Bachelor	02:00:00		
13.08.2024	Préparation de la clé USB pour l'après-midi et relecture du poster	00:30:00		
13.08.2024	Déploiement d'une première version du travail de Bachelor sur un casque de réalité virtuelle	04:30:00		
14.08.2024	Corrections du poster	02:00:00		
14.08.2024	Corrections du rapport selon la relecture des membres de ma famille	02:30:00		
15.08.2024	Finalisation du travail de Bachelor	02:30:00		
15.08.2024	Préparation des fichiers de rendu	01:30:00		
			Finalisation	56:30:00

Annexe IV : Détail du sprint 0

The chart displays the progress of Sprint 0. The 'Theoretical' line (red) shows a steady decrease in work from 25 hours to 0 hours over 7 days. The 'Remaining' line (blue) shows the actual work remaining, which starts at 25 hours, drops to 15 hours by Day 3, and then fluctuates slightly before reaching 0 hours by Day 7.

Jour	Date	Theoretical (H)	Remaining (H)
Jour 1	17.06.2024	25,00	25,00
Jour 2	18.06.2024	20,00	20,00
Jour 3	19.06.2024	15,00	15,00
Jour 4	20.06.2024	10,00	12,00
Jour 5	21.06.2024	5,00	10,00
Jour 6	22.06.2024	0,00	5,00
Jour 7	23.06.2024	0,00	0,00

Annexe V : Détail du sprint 1

Sprint		1							
Date de début		08.07.2024							
Date de fin (inclus)		14.07.2024							
Estimation du sprint Initial (SP)		17							
Total SP effectué		17							
But du sprint		Intégrer une infirmière immersive capable d'initier une conversation et d'échanger							
Nbre de jours		7							
ID US.task	Nom de la tâche	Estimation initiale	Jour 1 08.07.2024	Jour 2 09.07.2024	Jour 3 10.07.2024	Jour 4 11.07.2024	Jour 5 12.07.2024	Jour 6 13.07.2024	Jour 7 14.07.2024
1.1	Importer le personnage créé sur Conval	2.00	2.00	2.00	2.00	0.00	0.00	0.00	0.00
1.2	Paramétrer le personnage pour pouvoir échanger	8.00	8.00	8.00	8.00	0.00	0.00	0.00	0.00
1.3	Effectuer un échange avec le personnage en Play mode	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
1.4	Documentation	5.00	5.00	5.00	5.00	4.00	1.00	0.00	0.00
2.1	Intégrer le suivi du joueur par le regard	3.00	3.00	3.00	3.00	0.00	0.00	0.00	0.00
2.2	Intégrer l'animation des lèvres	5.00	5.00	5.00	5.00	0.00	0.00	0.00	0.00
2.3	Tester le suivi du regard et la synchronisation labiale en Play mode	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
2.4	Documentation	5.00	5.00	5.00	5.00	4.00	1.00	0.00	0.00
3.1	Questionner le personnage sur les éléments précédemment discutés	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00
3.2	Documentation	5.00	5.00	5.00	5.00	5.00	5.00	4.00	0.00
4.1	Tenter de sortir le personnage de son contexte durant l'échange	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00
4.2	Documentation	5.00	5.00	5.00	5.00	5.00	5.00	4.00	0.00
5.1	Importer le Design narratif de Conval sur Unity	4.00	4.00	4.00	4.00	4.00	4.00	0.00	0.00
5.2	Paramétrer les déclencheurs du Design narratif	6.00	6.00	6.00	6.00	6.00	6.00	4.00	0.00
5.3	Tester l'initiation de l'échange par l'infirmière en jeu	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00
5.4	Documentation	5.00	5.00	5.00	5.00	5.00	5.00	4.00	0.00
Total		58.00	58.00	58.00	58.00	36.00	28.00	16.00	0.00
Remaining		58.00	58.00	46.53	36.67	29.00	19.33	9.67	0.00
Theoretical		58.00	58.00	46.53	36.67	29.00	19.33	9.67	0.00

Burn down chart Sprint 1

The chart displays the progress of Sprint 1. The Y-axis represents hours (0.00 to 70.00), and the X-axis represents days (08.07.2024 to 14.07.2024). The blue line (Remaining) starts at 58.00 and decreases to 0.00 by Day 7. The red line (Theoretical) starts at 58.00 and decreases to 0.00 by Day 7.

Jours	Remaining	Theoretical
Jour 1	58.00	58.00
Jour 2	58.00	58.00
Jour 3	58.00	58.00
Jour 4	58.00	58.00
Jour 5	58.00	58.00
Jour 6	58.00	58.00
Jour 7	58.00	58.00

Annexe VI : Détail du sprint 2

Sprint		2							
Date de début		15.07.2024							
Date de fin (indus)		21.07.2024							
Estimation du sprint initial (SP)		26							
Total SP effectué		26							
But du sprint		Obtenir une infirmière prête pour réaliser les objectifs de Sim3D							
Nbre de jours		7							
ID UStask	Nom de la tâche	Estimation initiale	Jour 1 15.07.2024	Jour 2 16.07.2024	Jour 3 17.07.2024	Jour 4 18.07.2024	Jour 5 19.07.2024	Jour 6 20.07.2024	Jour 7 21.07.2024
6.1	Intégrer les composants de déplacement au personnage	3.00	3.00	0.00	0.00	0.00	0.00	0.00	0.00
6.2	Spécifier les obstacles de la scène pour avoir un déplacement correct	6.00	6.00	3.00	0.00	0.00	0.00	0.00	0.00
6.3	Tester le déplacement du personnage	1.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
6.4	Documentation	5.00	5.00	3.00	3.00	0.00	0.00	0.00	0.00
7.1	Vérifier les animations Conval et les reparamétrer selon les actions	4.00	4.00	4.00	2.00	1.00	1.00	0.00	0.00
7.2	Tester les animations du personnage	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
7.3	Documentation	5.00	5.00	5.00	4.00	4.00	4.00	2.00	0.00
8.1	Intégrer les composants de saisie des objets au personnage	3.00	3.00	0.00	0.00	0.00	0.00	0.00	0.00
8.2	Spécifier les objets à saisir	4.00	4.00	1.00	1.00	0.00	0.00	0.00	0.00
8.3	Demandar au personnage de ramasser les objets	1.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
8.4	Documentation	5.00	5.00	4.00	4.00	3.00	1.00	0.00	0.00
9.1	Créer une action pour permettre au personnage de déposer l'objet	5.00	5.00	5.00	3.00	1.00	0.00	0.00	0.00
9.2	Paramétrer la position du dépôt du médicament dans la scène	4.00	4.00	3.00	3.00	1.00	0.00	0.00	0.00
9.3	Demandar au personnage de poser le médicament	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00
9.4	Documentation	5.00	5.00	5.00	5.00	4.00	2.00	1.00	0.00
10.1	Créer une action pour permettre au personnage d'utiliser l'objet	9.00	9.00	9.00	7.00	2.00	0.00	0.00	0.00
10.2	Paramétrer la position où utiliser l'outil du patient	7.00	7.00	5.00	4.00	1.00	0.00	0.00	0.00
10.3	Demandar au personnage d'utiliser les quatre outils	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00
10.4	Documentation	5.00	5.00	5.00	5.00	4.00	2.00	1.00	0.00
Total		75.00	75.00	56.00	44.00	23.00	10.00	4.00	0.00
Remaining		75.00	75.00	62.50	50.00	37.50	25.00	12.50	0.00
Theoretical		75.00	75.00	62.50	50.00	37.50	25.00	12.50	0.00

Burn down chart Sprint 2

Jours	Remaining	Theoretical
Jour 1	75.00	75.00
Jour 2	62.50	62.50
Jour 3	50.00	50.00
Jour 4	37.50	37.50
Jour 5	25.00	25.00
Jour 6	12.50	12.50
Jour 7	0.00	0.00

Annexe VII : Transcription d'un dialogue avec l'infirmière Convai

La transcription ci-dessous provient d'une conversation effectuée durant le mode Play de l'éditeur Unity avec l'infirmière Harlene. La transcription a été téléchargée telle quelle depuis le site de Convai, sous <https://convai.com/pipeline/dashboard/character?id=df8c1896-2cae-11ef-aaf8-42010a7be00e>, dans la catégorie de la mémoire du personnage. La conversation date du 13 juillet 2024, et est enregistrée sous l'identifiant de session « 3286e3125a47f091865e95c361dfdbd1 ».

User: [An Event has occurred]: L'utilisateur rentre dans la chambre d'hôpital

Harlene: Bonjour ! Je suis Harlene, l'infirmière ici. Êtes-vous le docteur venu pour faire l'examen de routine auprès du patient ?

User: Oui bonjour, je m'appelle Bastien. Je suis un nouveau médecin ici, je vais commencer mais première journée aujourd'hui.

Harlene: Enchantée de faire votre connaissance, Bastien ! Le patient est dans un état stable, et les outils sont sur le chariot, prêts à être utilisés. N'hésitez pas à me demander ce dont vous avez besoin pour l'examen.

User: quels outils avez-vous préparé

Harlene: J'ai préparé les outils suivants : le thermomètre, le pulsomètre, le tensiomètre et le stéthoscope. Vous pouvez me demander celui dont vous avez besoin, et je vous le passerai.

User: pouvez-vous me passer le thermomètre s'il vous plaît

Harlene: Bien sûr, voici le thermomètre. Tenez, docteur Bastien.

User: La température du patient à l'air stable je vous rends le thermomètre, pouvez-vous me passer d'un autre outil s'il vous plaît.

Harlene: D'accord, je reprends le thermomètre. Quel outil avez-vous besoin maintenant, docteur Bastien ?

User: Et qu'elle avez-vous à disposition.

Harlene: J'ai aussi à disposition le pulsomètre, le tensiomètre et le stéthoscope. Lequel souhaitez-vous utiliser ?

User: Je vais passer avec le tensiomètre. Alors pouvez-vous me le passer.

Harlene: Bien sûr, voici le tensiomètre. Tenez, docteur Bastien.

User: J'ai fini avec le tensiomètre, pouvez-vous me passer cette fois le stéthoscope s'il vous plaît.

Harlene: D'accord, je reprends le tensiomètre. Voici le stéthoscope, docteur Bastien.

User: Parfait, j'ai fini avec le stéthoscope, pouvez-vous me passer le dernier outil s'il vous plaît.

Harlene: D'accord, je reprends le stéthoscope. Voici le pulsomètre, docteur Bastien.

User: Ok, j'ai fini avec le pulsomètre. J'ai regardé les valeurs que j'ai pu relever le patient a l'air dans un état normal quand que devons-nous que devons-nous faire pour la suite.

Harlene: Je vais reprendre le pulsomètre. Ensuite, vous pouvez déposer le médicament sur la table du patient. Une fois cela fait, nous pourrons quitter la chambre ensemble et souhaiter une bonne journée au patient.

User: que dites-vous de posez-vous le médicament sur la table du patient

Harlene: Bien sûr, je vais déposer le médicament sur la table du patient.

User: Bien alors, je crois que nous avons terminé ici vous avez été d'une grande aide.

Harlene: Merci beaucoup, docteur Bastien. C'était un plaisir de travailler avec vous. Je vous souhaite une excellente journée, et vous aussi, cher patient !

User: Juste avant de partir, pouvez-vous me rappeler quel est votre prénom s'il vous plaît.

Harlene: Bien sûr, je m'appelle Harlene. À bientôt, docteur Bastien !

User: et quel est votre rôle

Harlene: Je suis infirmière ici. Je vous assiste durant les examens et m'assure que tout le matériel nécessaire soit prêt et à disposition. Je m'occupe aussi du bien-être des patients tout au long de leur séjour.

User: Vous vous souvenez qu'on avait dans le qu'on avait pris un rendez-vous romantique dans un autre pays américains.

Harlene: Euh... je crois que vous devez faire erreur. Je suis simplement ici pour m'occuper des patients et aider durant les examens médicaux. Peut-être confondez-vous avec quelqu'un d'autre ?

User: Pardon toutes mes excuses, je me suis effectivement trompé. Allez, j'arrête de vous embêter. Je vous souhaite une bonne journée.

Harlene: Pas de souci, cela peut arriver à tout le monde. Je vous souhaite également une excellente journée, docteur Bastien. À bientôt !