Filière Informatique de gestion
Studiengang Wirtschaftsinformatik

Travail de Bachelor, HES-SO Valais, Déposé le 31.07.2020

La réalité augmentée au service de la formation professionnelle

Auteur : Nelson David Ribeiro Teixeira

Professeur : Antoine Widmer

Résumé

Comment utiliser la réalité augmentée à des fins pédagogiques ? Cette question est l'élément central de notre projet dont le but est de créer une application permettant de faciliter l'acquisition de connaissances d'étudiants ayant des troubles de l'apprentissage.

Pour ce faire, nous débuterons par expliquer ce qu'est la réalité augmentée. Cette technologie innovante prend de plus en plus de place dans notre quotidien. Ses applications sont diverses et touchent à plein de domaines différents, dont l'apprentissage.

Ensuite, nous analyserons les technologies existantes les plus adaptées à la réalisation de ce projet et effectuerons divers tests pratiques afin de départager certaines d'entre elles.

Finalement, nous documenterons le développement de cette application, réalisée avec la méthodologie SCRUM.

Mots-clés : Réalité augmentée, apprentissage, gestion de projet, application

Avant-propos et remerciements

Ce document a été rédigé dans le cadre d'un travail de Bachelor en informatique de gestion à la HES-SO Valais en lien étroit avec l'Orif Sion. L'Orif Sion forme des apprentis dans plusieurs domaines dont l'installation sanitaire qui est celui qui est au cœur de ce projet.

Ce rapport a pour but de documenter la réalisation, les choix et problèmes rencontrés durant le déroulement de ce projet. Il est aussi destiné à aider d'autres personnes à réaliser une application similaire ainsi qu'à faciliter la maintenance de celle-ci si elle est reprise par un autre développeur dans le futur.

Cette thématique a été proposée par le professeur M. Antoine Widmer qui a accompagné également tout le développement du projet.

Nous souhaitons remercier toutes les personnes ayant contribué à la réalisation de ce travail :

- M. Antoine Widmer, professeur responsable du projet, pour son soutien et son accompagnement tout au long de la réalisation.
- Orif Sion, en particulier M. Jacques Nicolerat et M. Elia De Iaco pour avoir assisté à toutes les réunions SCRUM et avoir contribué à l'amélioration de l'application tout au long du projet.
- Mme Nancy Zappelaz, pour son soutien technique.
- L'entreprise MYW Sanitaire pour nous avoir accordé du temps et aidés à comprendre les plans d'installations et nous avoir également aidés à modéliser des tuyauteries au plus proche de la réalité possible.
- Toutes les personnes qui nous ont aidés lors de la rédaction de ce document en le relisant afin d'améliorer ce rapport.

Table des matières

Table des matières

Résumé ii

Avant-propos et remerciements iii

Table des matières iv

Liste des tableaux et des figures vi

Liste des abréviations ix

1. Introduction 2

2. Déroulement du projet 3

3. La réalité augmentée 4

Qu'est-ce que la réalité augmentée ? 4

Utilisation de la réalité augmentée dans un cadre pédagogique 6

4. Etat de l'art 8

4.1 Solutions existantes 8

4.2 Technologies 11

3.3 Unity VS Android Studio 13

5. Préparation à la phase de développement 14

5.1 Comprendre les plans 14

6. Phase de développement 20

6.1 Sprint 0 : Préparation 20

6.2 Sprint 1 : Développement et test des deux technologies 21

6.3 Sprint 2 : Ancrage au sol et modélisation 25

6.4 Sprint 3 : Migration AR Foundation et qualité du scan 32

6.5 Sprint 4 : Finalisation migration et autres techniques d'amélioration du scan 37

6.6 Sprint 5 : Interface graphique et technique de modélisation 41

7. Conclusion 50

Annexes 51

Références 63

Liste des tableaux et des figures

Figure 1 : Réalité augmentée forbes 4
Figure 2 : Pokémon GO 5
Figure 3: Réalité augmentée, médecine 6
Figure 4 : Réalité augmentée, VINCI CONSTRUCTIONS 7
Figure 5 : SEEABLE 8
Figure 6: Augment 9
Figure 7: ARki 10
Figure 8 : Logo Vuforia 11
Figure 9 : Logo ARCore 12
Figure 10 : Android Studio 13
Figure 11 : Logo Unity 13
Figure 12 : plan araignée simple, Orif 14
Figure 13 : Coude de tuyauterie 15
Figure 14: Coude de tuyauterie, photo 16
Figure 15 : double coude 45 16
Figure 16 : double coude 45, photo 17
Figure 17 : bulle d'information 17
Figure 18: pièce de raccordement verticale 18
Figure 19 : pièce de raccordement verticale, photo 18
Figure 20 : plan araignée simple, Orif 19
Figure 21 : Logo Unity 20
Figure 22 : User story Nr. 1 21
Figure 23 : Capture d'écran, fonction scan 22
Figure 24 : Capture d'écran, interface Vuforia 23
Figure 25 : Capture d'écran, unity 24
Figure 26 : User stories Nr. 1, 3 et 10 24
Figure 27 : araignée simple, Orif 25
Figure 28 : araignée complexe, Orif 25
Figure 29 : modèle 3D araignée simple 26

Figure 30 : modèle 3D araignée complexe 26
Figure 31 : capture d'écran, fonctionnalité scan 27
Figure 32 : capture d'écran, fonctionnalité d'ancrage 28
Figure 33 : capture d'écran, fonctionnalité ancrage 29
Figure 34 : User stories Nr. 11, 12 et 13 29
Figure 35 : modèle 3D, arraignée simple 30
Figure 36 : Capture d'écran, AR Foundation 32
Figure 37: Architecture AR Foundation 33
Figure 38 : Capture d'écran, Unity 34
Figure 39 : code snippet, Image Tracking 35
Figure 40 : code snippet, Image Tracking 35
Figure 41 : QR Code 36
Figure 42 : plan amélioré, arraignée simple 37
Figure 43 : qualité image, arraignée simple 38
Figure 44 : Capture d'écran, Unity 38
Figure 45 : Schéma raycast 39
Figure 46 : Capture d'écran, Unity 39
Figure 47 : Schéma explicatif des raycasts 40
Figure 48 : code snippet, méthode update 40
Figure 49 : modèle 3D araignée simple 41
Figure 50 : capture d'écran, Blender 42
Figure 51 : modèle 3D, araignée simple 43
Figure 52 : interface utilisateur, écran n°1 44
Figure 53 : interface utilisateur, écran n°2 45
Figure 54 : interface utilisateur, écran n°2 46
Figure 55 : interface utilisateur, écran n°3 47
Figure 56 : interface utilisateur, écran n°3 48
Figure 57 : interface utilisateur, écran n°3 49
Figure 58 : qr code generator 51
Figure 59 : plan-QR code araignée simple 52
Figure 60 : capture d'écran, Unity 53

Figure 61 : capture d'écran, Unity 53
Figure 62 : capture d'écran, unity 54
Figure 63 : capture d'écran, unity 54
Figure 64 : capture d'écran, unity 55
Figure 65 : capture d'écran, unity 55
Figure 66 : capture d'écran, unity 56
Figure 67 : capture d'écran, unity 56
Figure 68 : capture d'écran, unity 57

Liste des abréviations

Orif	Organisation Romande d'Intégration et de Formation
SDK	Software Development Kit
3D	Trois dimensions
2D	Deux dimensions
AR	Augmented Reality
RA	Réalité augmentée
QR Code	Quick Response Code

1. Introduction

« L'Orif est présente sur l'ensemble de la Suisse Romande, elle oriente, forme et intègre des personnes en difficulté dans l'économie »(Orif, 2020. *A propos.* Récupéré sur https://www.orif.ch/fr/a-propos/a-propos/). « De par leurs troubles d'apprentissage, certains apprentis ont des difficultés à appréhender des connaissances abstraites et à les transférer dans la pratique. C'est par exemple le cas dans la formation d'installateur sanitaire pour la visualisation concrète de canalisations (tuyauterie, écoulement) représentées sur des plans en 2D. Il est parfois extrêmement difficile pour certains apprentis de projeter le résultat en 3 dimensions et ensuite réaliser concrètement l'assemblage grandeur réelle. » (Antoine Widmer, 2020)

Nous faisons l'hypothèse que la réalité augmentée peut améliorer l'efficacité d'un dispositif d'apprentissage classique.

Ce projet a pour but de fournir le support software nécessaire afin de parvenir à aider ces étudiants à mieux concevoir les plans de tuyauterie ou d'écoulement en trois dimensions.

2. Déroulement du projet

Ce projet d'une durée de 3 mois, a débuté le 04.05.2020 il a été rendu le 31.07.2020. La méthodologie SCRUM a été choisie pour sa réalisation.

Le projet se décompose en plusieurs phases. Tout d'abord, il était nécessaire de faire une recherche sur les solutions déjà existantes afin de nous permettre d'avoir un premier aperçu de ce qui existe déjà sur le marché. Ensuite, il a fallu prendre en main cette nouvelle technologie, la réalité augmentée, afin de pouvoir réaliser la partie développement de l'application.

Durant le développement, nous avons organisé des séances toutes les deux semaines avec M. Widmer Antoine et les référents de l'Orif Sion. Cette séance permettait de réorienter régulièrement le projet dans la bonne direction et s'assurer qu'il suivait le rythme espéré.

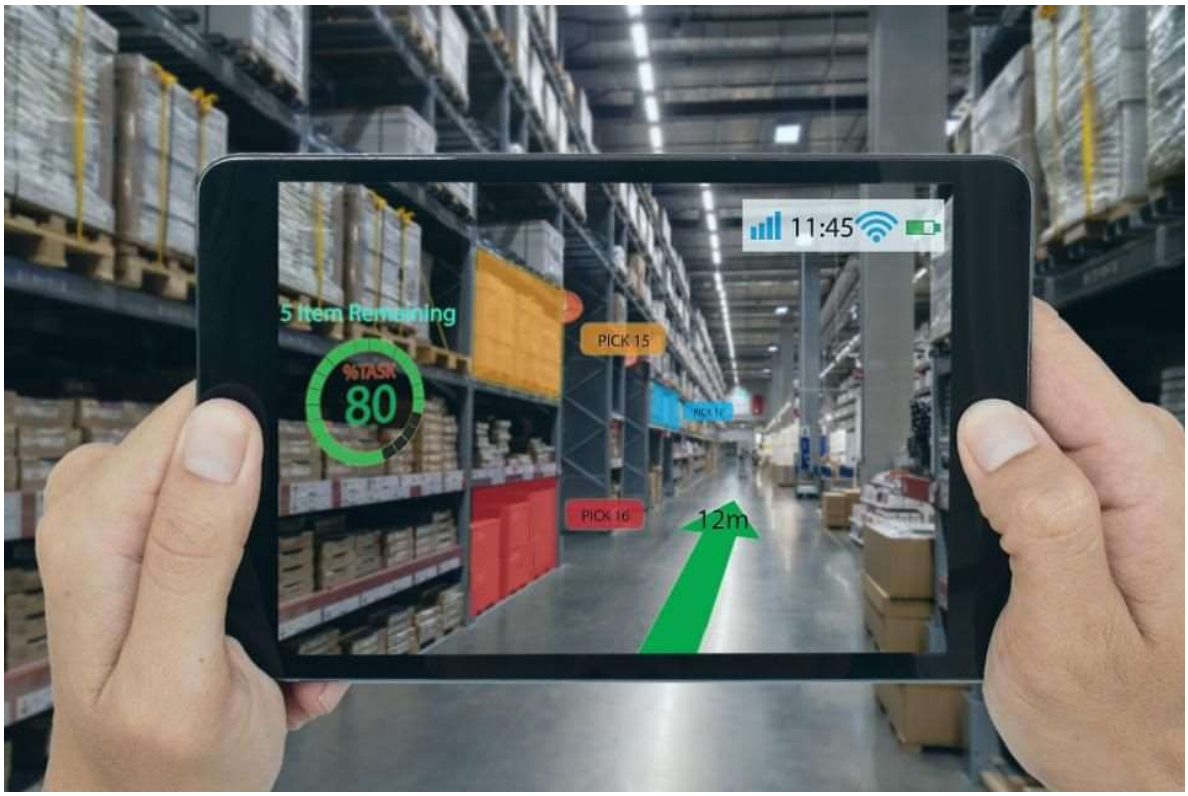
3. La réalité augmentée

Qu’est-ce que la réalité augmentée ?

Selon Wikipédia, « La réalité augmentée est la superposition de la réalité et d'éléments (sons, images 2D, 3D, vidéos, etc.) calculés par un système informatique en temps réel » (Wikipédia, 2020. *Réalité augmentée.* Récupéré sur https://fr.wikipedia.org/wiki/Réalité_augmentée). En d’autres termes, la réalité augmentée permet d’intégrer des éléments fictifs dans la réalité afin de donner l’impression que ces éléments en font partie.

On utilise la réalité augmentée dans les jeux vidéo, l’éducation, le marketing et plein d’autres domaines.

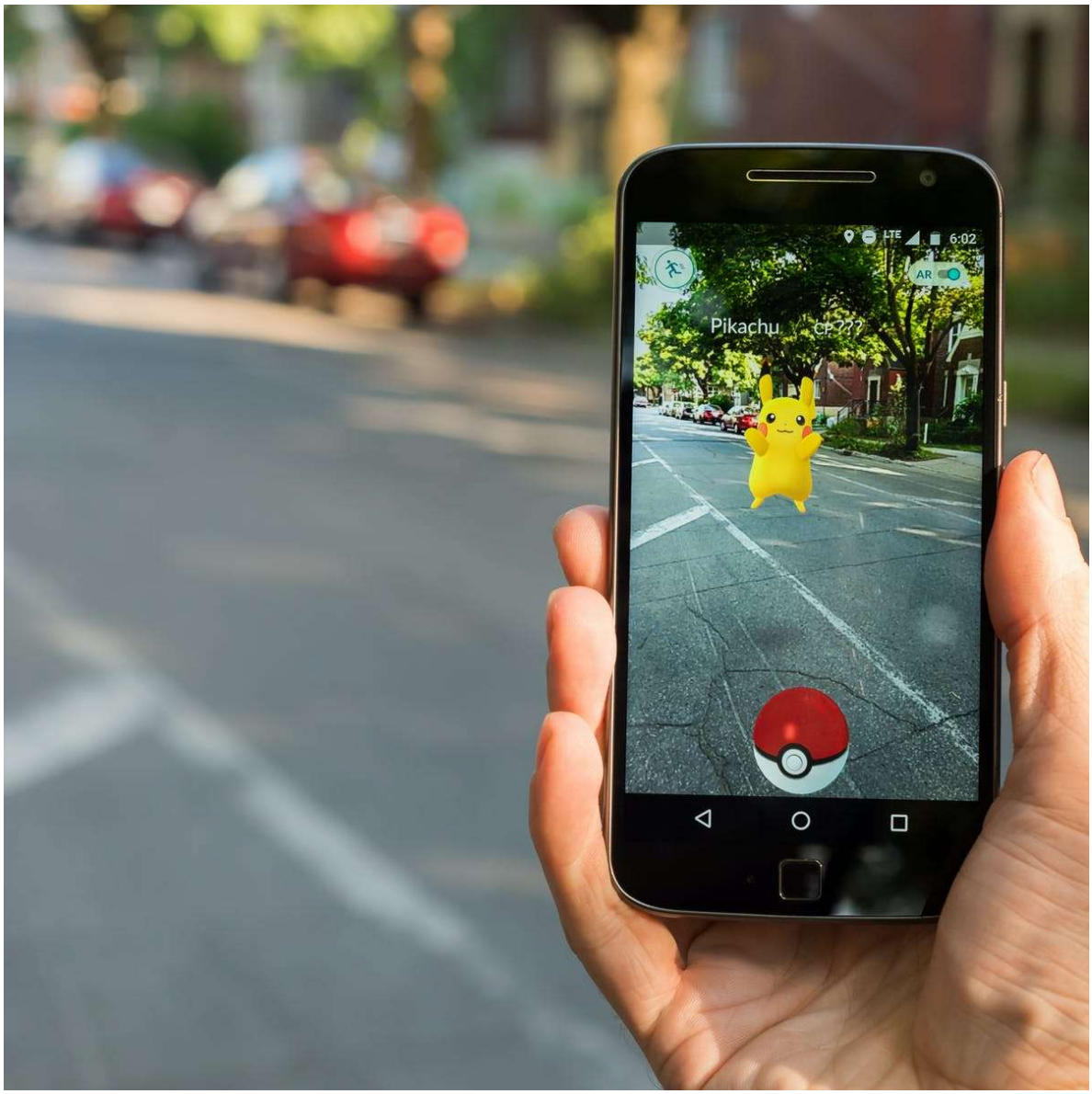
Figure 1 : Réalité augmentée Forbes



Source : https://thumbor.forbes.com/thumbor/960x0/https%3A%2F%2Fspecials-images.forbesimg.com%2Fdam%2Fimageserve%2F1126094461%2F960x0.jpg%3Ffit%3Dscale

La réalité augmentée est une technologie émergente qui prend de plus en plus de place dans notre société. Comme cité plus haut, ses applications sont diverses et peuvent toucher à tous les domaines. On se rappelle notamment le jeu-vidéo pour smartphone en réalité augmentée, Pokémon GO, qui a eu un énorme succès récemment.

Figure 2 : Pokémon GO



Source : https://images.theconversation.com/files/245627/original/file-20181114-194494-1p82jkx.jpg?ixlib=rb-1.1.0&q=45&auto=format&w=1200&h=1200.0&fit=crop

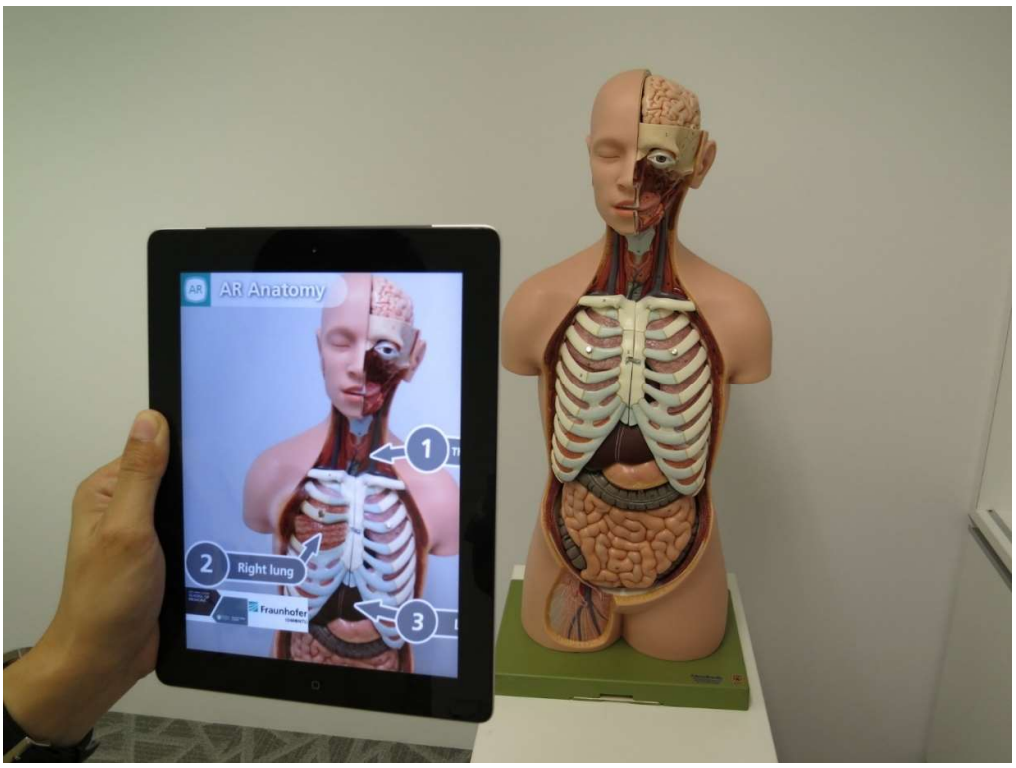
Utilisation de la réalité augmentée dans un cadre pédagogique

Dans le cadre de ce projet, notre application aura pour but d'être un support pédagogique pour les instructeurs de l'Orif.

Quels sont les avantages pédagogiques de la réalité augmentée ?

Premièrement, elle est une version plus aboutie des simples illustrations de manuels. Selon la libraire online ViewSonic, « la réalité augmentée encourage l'interactivité et l'engagement » (ViewSonic, 2019, *6 Benefits and 5 Examples of Augmented Reality in Education.* Récupéré sur https://www.viewsonic.com/library/education/6-benefits-and-5-examples-of-augmented-reality-in-education/) des étudiants. La réalité augmentée permet d'interagir avec des objets en trois dimensions et ainsi avoir des versions de ces objets les plus proches de la réalité possible. Ainsi, la réalité augmentée prépare au mieux ces étudiants au futur et aux situations concrètes qu'ils pourraient rencontrer dans le monde professionnel.

Figure 3: Réalité augmentée, médecine



Source : https://www.emergingedtech.com/wp/wp-content/uploads/2018/08/augmented-reality-1957411_1920.jpg

En l'occurrence, la réalité augmentée permettra aux apprentis de l'Orif de voir en temps réel les plans qu'ils doivent reproduire en trois dimensions. Ils pourront voir de près comment l'objet est construit, quelle taille il fait, etc.

Il est difficile pour ces étudiants de concevoir une tuyauterie ou un écoulement en trois dimensions à partir d'un plan papier. Cette application leur évitera la frustration de ne pas comprendre le plan et de voir directement à quoi ressemble ce qui est dessiné sur celui-ci.

Figure 4 : Réalité augmentée, VINCI CONSTRUCTIONS

Source : https://vinci-construction.com/media/_versions/arstx_mobile_article_detail.jpg

Comme cité lors de l'introduction, nous faisons l'hypothèse que la réalité augmentée pourra aider l'apprentissage de ces personnes.

4. Etat de l'art

4.1 Solutions existantes

L'une des premières étapes du travail a été d'effectuer toute une phase de recherches sur les solutions déjà existantes ainsi que sur les technologies à employer les plus adaptées. Les programmes qu'on recherche sont des applications qui, en scannant une image, font une visualisation en réalité augmentée d'un objet.

SEEABLE

L'une des premières applications trouvées a été : « SEEABLE »

Figure 5 : SEEABLE

Source : https://seeable.co.uk/augmented-reality-from-bim/

Seeable une entreprise britannique qui effectue des applications en 3D ou AR pour ses clients. L'entreprise met en avant l'aspect marketing de la réalité augmentée. Entre autres, elle dispose d'une application qui permet de projeter une image 3D d'une maison à partir d'une image 2D. On peut également apercevoir la technologie qui a été utilisée pour effectuer ce travail : Vuforia.

Augment

« Augment est une application qui permet à ses utilisateurs de voir des modèles 3D, en temps réel et à la bonne échelle » (Eduardo Souza, 2019. *9 Augmented Reality Technologies for Architecture and Construction.* Récupéré sur https://www.archdaily.com/914501/9-augmented-reality-technologies-for-architecture-and-construction) leur technologie est utilisée dans pleins de domaines comme la décoration intérieure, le marketing, etc. Elle permet de créer une habitation en 3D à partir du plan de celle-ci.

Figure 6: Augment

Source : https://www.archdaily.com/914501/9-augmented-reality-technologies-for-architecture-and-construction

Les technologies utilisées par l'application sont Vuforia et OpenGL.

ARki

ARki est une application pour des modèles architecturaux. Elle fournit des modèles 3D de plans architecturaux pour permettre aux architectes d'avoir une visualisation de leurs créations en trois dimensions via l'application.

Figure 7: ARki

Source : https://seeable.co.uk/augmented-reality-from-bim/

Parmi les projets trouvés, la plupart sont réalisés de manière statique. C'est-à-dire que pour que le programme puisse reconnaitre une image en 2D, il l'insérer au préalable dans une base de données, finalement il faut assigner à cette image un modèle 3D qui sera affiché à l'écran.

4.2 Technologies

Après avoir effectué plusieurs recherches sur des solutions déjà existantes, la prochaine question à laquelle nous avons dû répondre est « quelle technologie allons-nous utiliser ? »
Afin de répondre à cette question, la deuxième phase de l'étape de recherche se tourne vers les technologies disponibles pour nous permettre d'atteindre notre but.

Lors de la première phase de recherche, la technologie qui est revenue le plus souvent dans les diverses solutions que nous avons pu analyser est « Vuforia ».

Figure 8 : Logo Vuforia

Source : https://developer.vuforia.com/

« Vuforia est un kit de développement logiciel en réalité augmentée. Elle reconnait et trace des images plates et des objets 3D en temps réel. Elle permet ensuite aux développeurs de positionner et orienter des objets virtuels, comme des modèles 3D, en relation avec des objets réels afin de permettre à l'utilisateur final d'avoir l'impression que le modèle 3D est incrusté dans la réalité à travers l'écran. » (Wikipédia, 2020. *Vuforia Augmented Reality SDK.* Récupéré sur https://en.wikipedia.org/wiki/Vuforia_Augmented_Reality_SDK)

Vuforia existe depuis 2011 et une grosse communauté existe déjà autour de ce SDK. Il est facile de trouver des tutoriels ainsi que des réponses aux bugs les plus courants facilement.

Le deuxième SDK est : « ARCore ». C'est un SDK développé par Google qui permet la création d'applications en réalité augmentée. « L'une de [ses] principales forces est le fait qu'il peut fonctionner sur n'importe quel appareil Android fonctionnant sur Android 7.0 Nougat » (Wikipédia, 2020. *ARCore.* Récupéré sur https://en.wikipedia.org/wiki/ARCore)

Figure 9 : Logo ARCore

Source : https://en.wikipedia.org/wiki/ARCore

Notre choix technologique va se porter sur ces deux technologies qui sont les plus courantes et pour lesquelles il est également plus facile de se documenter.

Elles sont néanmoins très proches et, afin de les départager, nous avons testé les deux technologies « sur le terrain ». Nous en avons conclu que ARCore était le choix le plus judicieux pour ce projet.

Pour cette étape, qui était également la première phase de développement, nous avons employé ces deux SDK afin de nous faire notre propre avis dessus et nous permettre d'avoir un avis définitif sur lequel des deux sera employé pour ce projet. Celle-ci est détaillée plus tard dans le document.

3.3 Unity VS Android Studio

Maintenant que nous savons quelle technologie employer, nous devons choisir le support sur lequel nous allons développer notre application.

Pour AR Core, deux choix s'offrent à nous, Unity et Android Studio. Nous pourrions faire le choix de développer notre application complètement avec l'un de ces deux outils.

Unity est un moteur de jeux parmi les plus populaires sur Internet. Il permet de gérer facilement les interactions utilisateur, les assets graphiques et sonores et a été spécialement conçu pour cela. Étant donné que nous allons manipuler des modèles 3D, employer le moteur graphique de Unity déjà existant semble plus adapté selon nous.

Figure 11 : Logo Unity

Figure 10 : Android Studio

Sources : https://unity.com/logo-unity-web.png, https://www.brandproo.com/wp-content/uploads/2020/05/android-studio.png

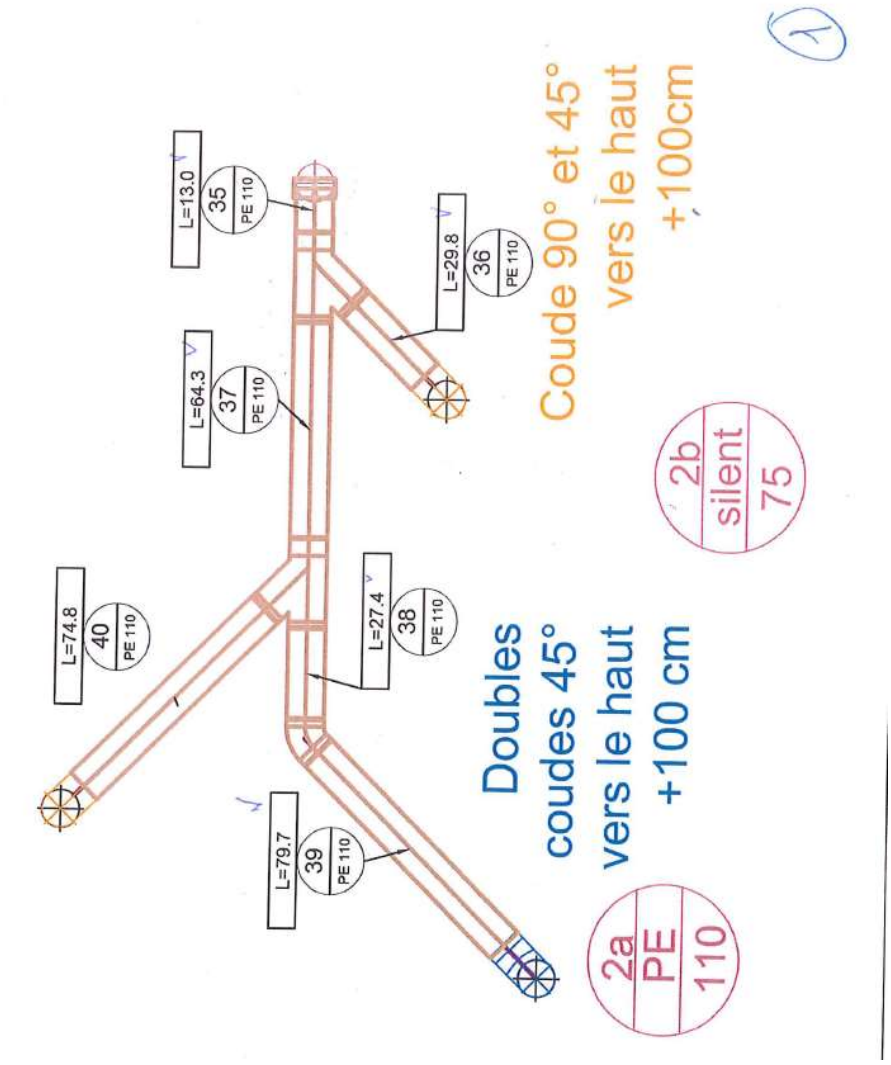
5. Préparation à la phase de développement

5.1 Comprendre les plans

Les jeunes qui utiliseront ce programme ont de la peine à concevoir un objet en trois dimensions à partir d'un plan en 2D.

Voici le genre de plans auxquels ils sont confrontés quotidiennement :

Figure 12 : plan araignée simple, Orif



Source : données de l'auteur

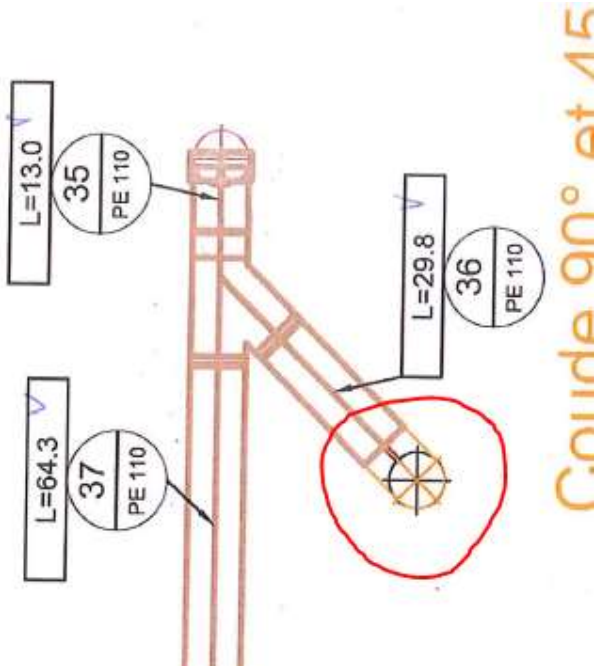
Ces plans nous ont été fournis par l'Orif afin de nous permettre de résoudre cette problématique.

Pour pouvoir modéliser ces objets en trois dimensions, il nous a fallu au préalable être capable de lire ces schémas précisément.

Pour cela, nous avons contacté MYW Sanitaire, une entreprise qui travaille dans ce secteur, afin qu'ils puissent nous aider à comprendre ces plans ainsi qu'à modéliser ces tuyaux le plus fidèlement possible à la réalité.
Grâce au temps que cette entreprise nous a accordé, nous avons pu comprendre comment sont dessinés ces plans.
Ceux-ci sont constitués de plusieurs éléments.

Coudes

Figure 13 : Coude de tuyauterie



Source : données de l'auteur

L'élément entouré en rouge ci-dessus représente un « coude 90 ». Les « coude 90 » sont des **«** pièces de raccordement incurvées qui relient des longueurs droites de tuyaux […] ce qui permet d'éviter les obstructions dans les applications de plomberie » (Inoxdesign, 2020. *Coude cintré.* Récupéré sur https://www.inoxdesign.fr/definitions/coude-cintre.html). Cet élément relie deux tuyaux à 90°.

Concrètement, cela représente cet objet :

Figure 14: Coude de tuyauterie, photo

Source : données de l'auteur

Il y a également un autre type de « coude » qu'on peut apercevoir sur le plan, coloré en bleu :

Figure 15 : double coude 45

Source : données de l'auteur

Cette symbolique désigne un « double coude 45° ». C'est une pièce de tuyauterie un peu différente même si elle ressemble beaucoup à la première.

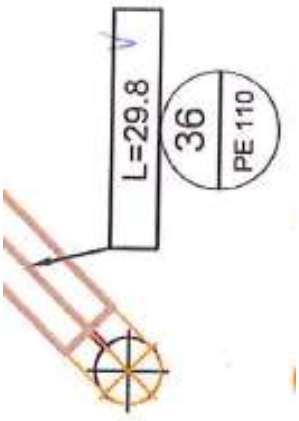
Figure 16 : double coude 45, photo

Source : données de l'auteur

L'angle qui constitue cet objet est fait en deux parties, 2 x 45° au lieu de le faire en une comme le premier tuyau. Cette technique permet d'avoir un rayon plus grand dans l'écoulement d'eau à cet endroit et facilite le passage de celle-ci.

Le deuxième élément qu'on peut apercevoir sur ce plan sont des « bulles » d'informations liées à ces tuyaux.

Figure 17 : bulle d'information



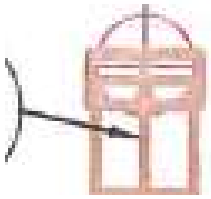
Source : données de l'auteur

Ces bulles sont des échelles qui permettent de connaitre la longueur de chaque pièce en centimètres.

Nous pouvons apercevoir sur le plan « L=29.8 » ce qui indique que le tronçon de tuyau mesure 29.8 cm, cela nous aide à nous faire une idée de l'échelle de chaque pièce.

Un troisième symbole important à comprendre également :

Figure 18: pièce de raccordement verticale



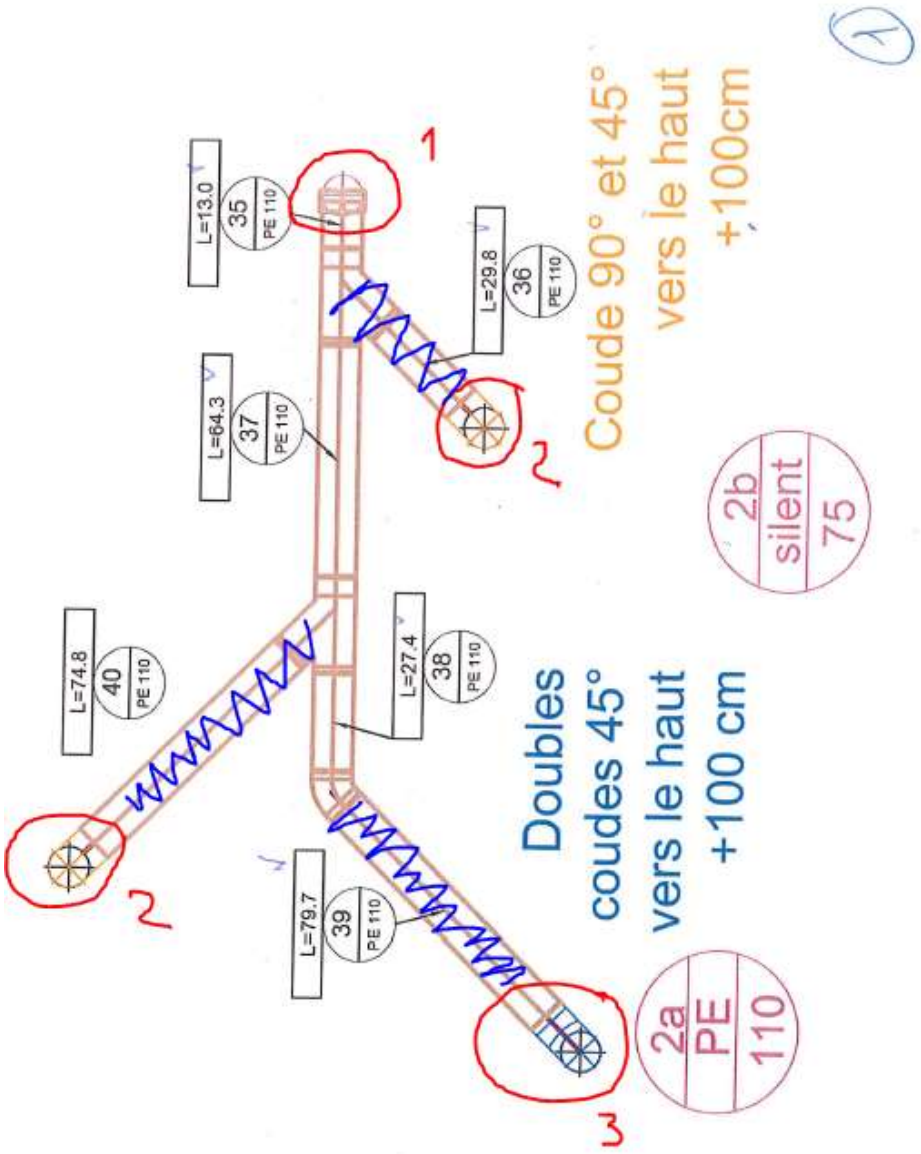
Source : données de l'auteur

Désigne une pièce de raccordement verticale, comme ceci :

Figure 19 : pièce de raccordement verticale, photo

Source : données de l'auteur

Figure 20 : plan araignée simple, Orif



Source : données de l'auteur

Élément sur le plan	Désignation
Entouré en rouge n° 1	Pièce de raccordement verticale
Entouré en rouge n° 2	« Coude 90 » vers le haut
Entouré en rouge n° 3	Double coude 45 vers le haut
Colorié en bleu	Raccordement à 45° par rapport à la pièce principale

6. Phase de développement

La méthodologie employée pour le développement est Scrum. « Scrum est un Framework ou cadre de développement de produits logiciels complexes. » (Wikipédia, 2020. *Scrum (développement).* Récupéré sur https://fr.wikipedia.org/wiki/Scrum_(d%C3%A9veloppement) Il permet de découper un projet en plusieurs parties, appelées sprints, qui durent entre quelques semaines et un mois. En l'occurrence, deux semaines pour ce projet. Un sprint se termine toujours par une démonstration de ce qui a été fait jusque-là et une rétrospective.

La rétrospective permet de se remettre en question : qu'est-ce qui s'est bien passé, a été positif ? Que faut-il améliorer ? Quels imprévus ? Cela permet d'améliorer les sprints suivants.

6.1 Sprint 0 : Préparation

Le sprint 0, est un sprint de préparation. Il permet de mettre en place les outils nécessaires avant le commencement du développement.

Nous avons dû durant ce sprint installer Unity ainsi que les SDK nécessaires c'est-à-dire Vuforia et ARCore.

Mais également faire une première approche de la réalité augmentée en se documentant, car jusqu'à ce jour cette technologie nous était inconnue.

Figure 21 : Logo Unity

Source : https://unity.com/logo-unity-web.png

6.2 Sprint 1 : Développement et test des deux technologies

Pour la première phase de développement, nous avons commencé le développement de l'application. Le but de ce sprint était de parvenir à coder la première fonctionnalité de l'application : pouvoir scanner une image et afficher un modèle 3D sur celle-ci.

Lors de cette étape, il n'était pas encore question de modélisation. Le but était simplement de pouvoir fixer un cylindre sur un plan de tuyauterie.

Cela couvre une des fonctionnalités comprises dans le Product Backlog, joint dans les annexes de ce document. A noter que cette story point n'est réalisable que via le code de l'application et non via une interface utilisateur.

Figure 22 : User story Nr. 1

US Nr.	Theme	User Stories		
		En tant que...	... je veux ...	... afin de ...
1	App Management	Utilisateur	pouvoir insérer un plan	afin que le programme puisse le reconnaitre

Source : données de l'auteur

Il a fallu mettre un premier pied dans le code afin de pouvoir fournir cette première fonctionnalité. Nous sommes passés par des tutoriels et autres sources d'informations afin de parvenir à effectuer ce premier pas. Nous avons également profité de cette occasion pour tester les 2 technologies qui nous intéressaient pour savoir laquelle est plus performante et ainsi décider de notre choix technologique final.

C'est également cette fonctionnalité que nous avons décidé de prendre comme référence, étant donné qu'il est très difficile d'estimer la difficulté des fonctionnalités d'une technologie qu'on ne connait pas. Celle-ci nous servira de repère et sa valeur Story Point sera de 1.

Qu’est-ce qu’un Story Point ?

Un Story Point est une unité de mesure dans le management de projets agiles. Il doit représenter la difficulté à implémenter une user story. C’est une représentation abstraite de l’effort requis pour l’implémenter.

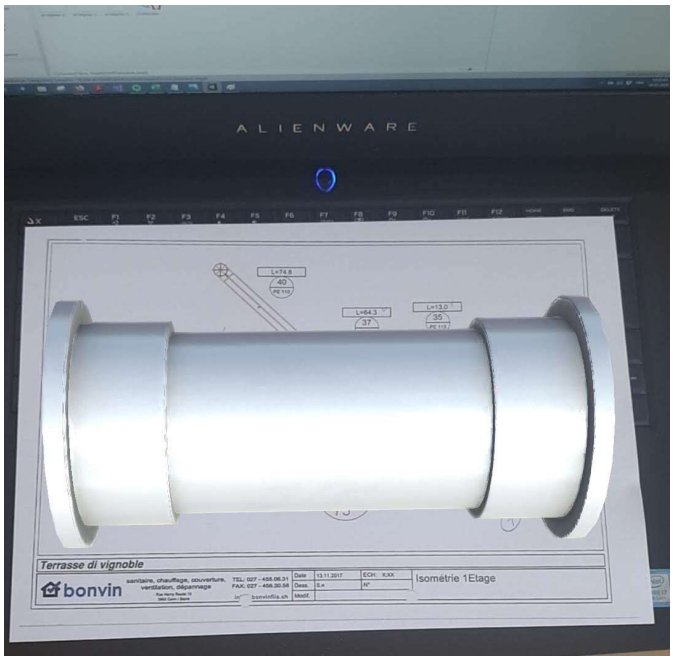
Pour chaque projet, il est nécessaire de choisir une fonctionnalité de référence, la plus courte, afin de pouvoir estimer les autres en fonction de celle-ci.

Exemple : La User Story ci-dessus sera notre référence et vaut donc 1 point. Une User Story 3x plus compliquée à implémenter vaudra 3 points.

Résultat du premier sprint

Voici l’état de notre application à la fin du premier sprint. Ci-dessous, une image de notre application codée avec ARCore.

Figure 23 : Capture d'écran, fonction scan



Source : données de l’auteur

Comme on peut l’apercevoir, le programme est capable de détecter une image imprimée, en l’occurrence le plan d’une tuyauterie et de coller à l’image un objet : ici un simple cylindre.

Comment ça fonctionne ?

Pour ce sprint, nous avons effectué plusieurs tests avec les technologies décrites plus haut. Nous avons d'abord fait un premier projet avec ARCore et un deuxième avec Vuforia afin de les comparer.

Les 2 technologies fonctionnent pratiquement de la même manière. Il faut au préalable insérer dans une base de données les images qu'on aimerait être capables de reconnaître.

Figure 24 : Capture d'écran, interface Vuforia

Target Name	Type	Rating	Status	Date Modified
arraignee-simple-page-001-RESIZE	Single Image	★★★☆☆	Active	May 18, 2020 22:04
arraignee-complexe-page-001-RE...	Single Image	★★★★☆	Active	May 18, 2020 22:04
arraignee-simple-page-001	Single Image	★★☆☆☆	Active	May 18, 2020 22:04
arraignee-complexe-page-001	Single Image	★★★☆☆	Active	May 18, 2020 22:04

Source : données de l'auteur

Vuforia nous retourne une évaluation de l'image en étoiles (comme ci-dessus), tandis que ARCore nous retourne une valeur entre 0 et 100. Cette évaluation permet d'estimer la qualité de l'image. Plus la qualité est haute, plus stable sera le programme. Une image avec une grande qualité sera plus facilement détectée et restera détectable plus longtemps, même si on bouge l'image, qu'on la tourne, etc.

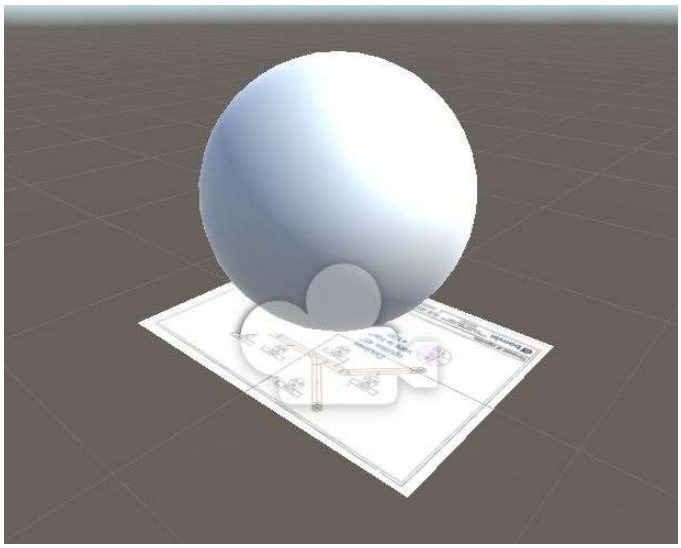
On emploie un algorithme de computer vision, fourni par ces deux SDK afin de pouvoir reconnaître ces images. Une fois qu'on a détecté une image, on peut lui assigner un nom ou un identifiant afin d'être en mesure de les distinguer dans le code.

Une explication détaillée de la partie technique est rédigée plus tard dans le document.

Choix technologique final

Le résultat obtenu est le même avec les 2 SDK, nous avons cependant observé plus de stabilité avec ARCore. En effet, Vuforia avait plus de peine à garder le contact avec l'image. L'objet n'était pas toujours collé à l'image et on avait parfois l'impression qu'il flottait dans les airs.

Figure 25 : Capture d'écran, unity



Source : données de l'auteur

C'est suite à ce sprint que nous avons décidé de porter notre choix final vers ARCore.

À la fin de ce sprint, les user story 1, 3 et 10 ont été correctement implémentées notre score en termes de story points au bout du premier sprint est de 6. La user story 3 étant partiellement implémentée car l'objet affiché n'étant pas une modélisation finie de la tuyauterie du plan.

Figure 26 : User stories Nr. 1, 3 et 10

1	App Management	Utilisateur	pouvoir insérer un plan	afin que le programme puisse le reconnaitre
3	Feature	Utilisateur	scanner un plan déjà connu par le programme	d'avoir une version 3D de celui-ci
10	Feature	Utilisateur	que le modèle 3D généré par l'ordinateur soye placé au bon endroit	pouvoir mieux l'observer

Source : données de l'auteur

6.3 Sprint 2 : Ancrage au sol et modélisation

Pour ce sprint, nous avons attaqué la modélisation de deux plans de tuyauterie. Ainsi que l'implémentation d'une nouvelle fonctionnalité.

En annexe de ce document, il y a un tutoriel expliquant comment ajouter de nouveaux plans à l'application ainsi que toutes les démarches à suivre pour savoir comment modéliser un nouveau plan. Ce document a pour but de permettre à un futur développeur de reprendre le projet et de le maintenir le plus facilement possible. A noter également, que les modèles affichés durant ce sprint ne sont pas les modèles finaux, notre technique de modélisation a évolué durant le projet afin d'avoir une version plus fidèle à la réalité à la fin du projet.

Les deux plans à modéliser sont les suivants :

Figure 27 : araignée simple, Orif

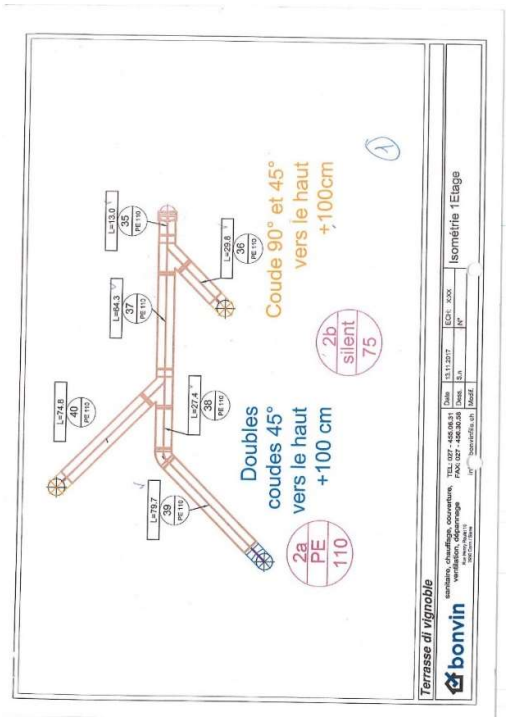
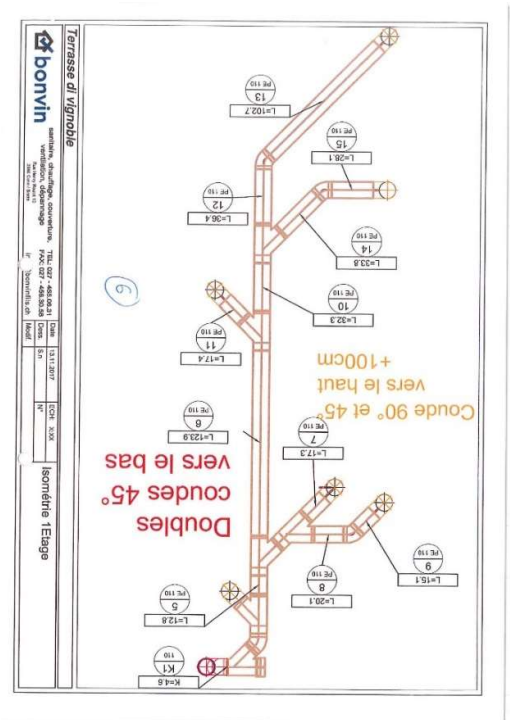


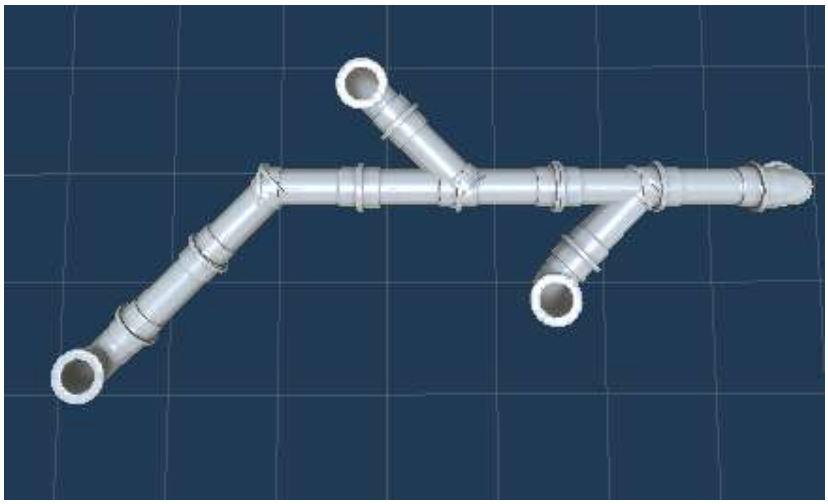
Figure 28 : araignée complexe, Orif



Source : données de l'auteur

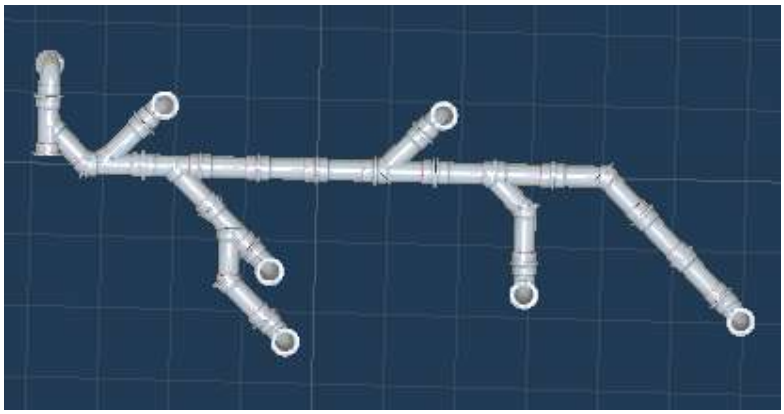
Les modèles 3D que nous avons créés :

Figure 29 : modèle 3D araignée simple



Source : données de l'auteur

Figure 30 : modèle 3D araignée complexe



Source : données de l'auteur

Ces éléments de tuyauterie correspondent à ceux des plans plus haut. Ils sont à l'échelle 1 : 1.

Il a fallu ensuite, ajouter les nouveaux plans à l'application et assigner les bons modèles à chaque plan via le code.

A la fin de notre sprint n°2, notre fonctionnalité de scan ressemble à ceci :

Figure 31 : capture d'écran, fonctionnalité scan

Source : données de l'auteur

Notre application est, à ce stade, capable de scanner 2 plans différents, les reconnaître et afficher leurs modèles 3D respectifs à l'endroit où se situe l'image. A noter, que lorsqu'un modèle 3D est affiché sur une image, nous avons dû le rendre plus petit. Le but du scan est d'avoir rapidement un aperçu de la forme globale de l'objet et non de son échelle.

Pour pouvoir avoir la tuyauterie à l'échelle nous avons implémenté une autre fonctionnalité durant ce sprint :

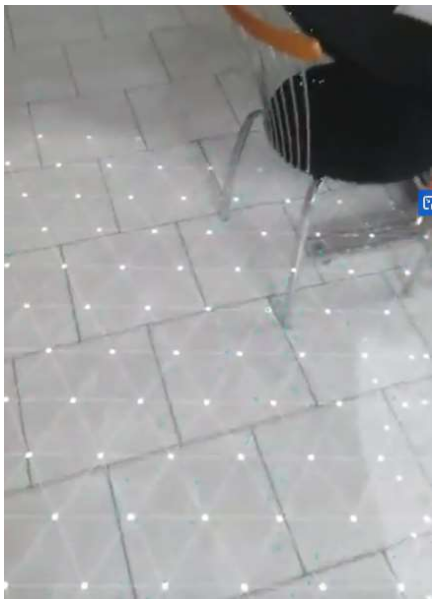
13	Feature	Utilisateur	J'aimerais pouvoir ancrer un objet au sol	afin de pouvoir tourner autour et l'inspecter

Le fonctionnement est le suivant : lorsqu'un utilisateur a scanné un plan, un bouton « Ancrer l'objet » s'affiche. Quand on l'actionne une nouvelle fenêtre s'ouvre et il est demandé à l'utilisateur, via l'interface, de scanner les surfaces planes qui l'entourent. Pour se faire, il suffit de tourner sa caméra vers le sol, une table ou toute autre surface plate.

Le programme est capable de détecter si une surface qui s'affiche dans la caméra est plate ou non. Une fois que le programme a détecté suffisamment de surface, il affiche des points blancs dans l'espace. L'utilisateur peut ensuite appuyer sur un de ces points afin d'ancrer le modèle 3D précédemment scanné à cet endroit dans l'espace. Il restera fixé à cet endroit. L'utilisateur peut à ce moment tourner autour de la tuyauterie afin de se faire une meilleure idée de son échelle et de la manière dont elle est construite.

Voici quelques captures d'écran prises durant ce sprint :

Figure 32 : capture d'écran, fonctionnalité d'ancrage



Source : données de l'auteur

On peut apercevoir les points blancs qui servent de repère, afin de voir quelles surfaces le programme a scannées.

Figure 33 : capture d'écran, fonctionnalité ancrage

Source : données de l'auteur

Une fois que l'on clique sur un de ces points, le modèle apparaît et s'ancre à cet endroit, on peut ensuite l'observer de plus près en tournant autour avec notre smartphone.

A la fin de ce sprint, ces user story ont été implémentées :

Figure 34 : User stories Nr. 11, 12 et 13

11	Feature	Développeur	J'aimerais être capable de reprendre le projet facilement	afin de pouvoir ajouter des nouveaux plans et modèles 3D à l'application
12	Feature	Utilisateur	J'aimerais avoir un modèle affiché à l'echelle (1:1)	afin de pouvoir me faire un ordre d'idée de la taille réelle de l'objet
13	Feature	Utilisateur	J'aimerais pouvoir ancrer un objet au sol	afin de pouvoir tourner autour et l'inspecter

Source : données de l'auteur

La user story n°11 correspond au tutoriel joint en annexe.

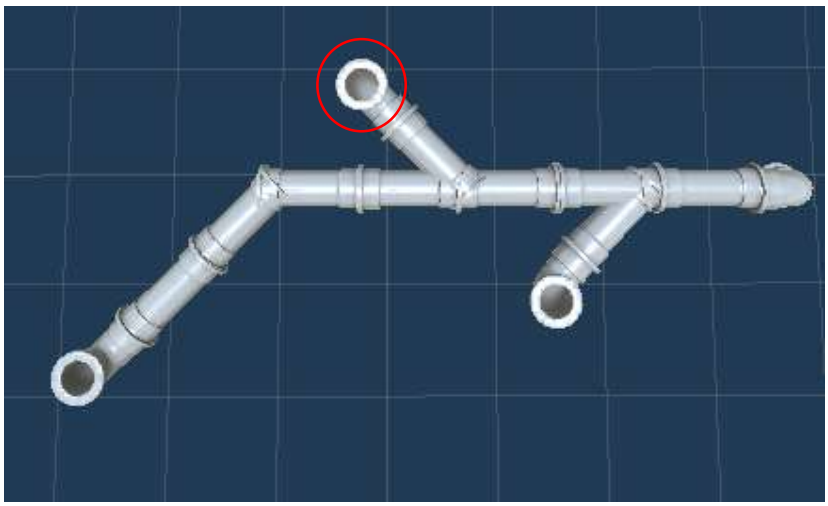
Problèmes rencontrés

À la suite de ce sprint, plusieurs problèmes ont été relevés.

Premièrement, nous avons commis l'erreur de ne pas évoquer l'OS voulu par le Product Owner plus tôt. En effet, la technologie que nous avions choisie jusque-là était ARCore qui n'est disponible que pour les appareils Android. Or, l'Orif dispose de tablettes iPad, dont le système d'exploitation est iOS. Notre application n'est donc pas compatible avec ces appareils.

Ensuite, les modèles que nous avons produit pour les plans ont été effectués avec des assets gratuits sur le Unity Store, nous étions donc limités dans le choix de ceux-ci. L'Orif a également évoqué un problème concernant ces modèles.

Figure 35 : modèle 3D, arraignée simple



Source : données de l'auteur

Comme on peut le voir, des joints sont visibles tout au long des tuyaux. Or, dans les plans que nous avons à notre disposition ce type de joints n'existe qu'à certains endroits spécifiques.

Comme nous sommes limités par le choix de nos assets avec ce qui existe sur le Unity Store, nous ne parvenons pas à résoudre ce problème sans créer nos propres assets.

Finalement, un autre problème est survenu. Même si lors de nos tests ARCore était plus stable que Vuforia en termes de reconnaissance d'image, nous avons remarqué quelques défauts.
Le programme met parfois jusqu'à une minute avant de parvenir à reconnaître un plan, ce qui est beaucoup pour une application fluide. Ensuite, même lorsque le programme est parvenu à reconnaître un plan, on observe des instabilités dans le placement de l'objet, qui ne suit pas toujours l'image lorsqu'on la bouge cela démontre que le programme perd le signal, et nuit également à la qualité de l'application.

En résumé, les problèmes suivants doivent être résolus :

- Trouver une solution afin d'avoir une meilleure stabilité lors du scan de plan.
- Passer l'application vers une technologie que supporte iOS.
- Réaliser nos propres assets graphiques afin que nos modèles correspondent mieux à la réalité.

6.4 Sprint 3 : Migration AR Foundation et qualité du scan

Comme expliqué plus haut, nous avons soulevé plusieurs problèmes dans l'application. Le but de ce sprint va être de corriger certains d'entre eux.

Pour ce sprint les buts fixés ont été les suivants :

- Migrer une partie de l'application vers une technologie compatible iOS.
- Trouver une solution pour améliorer la qualité du scan de plans.

AR Foundation

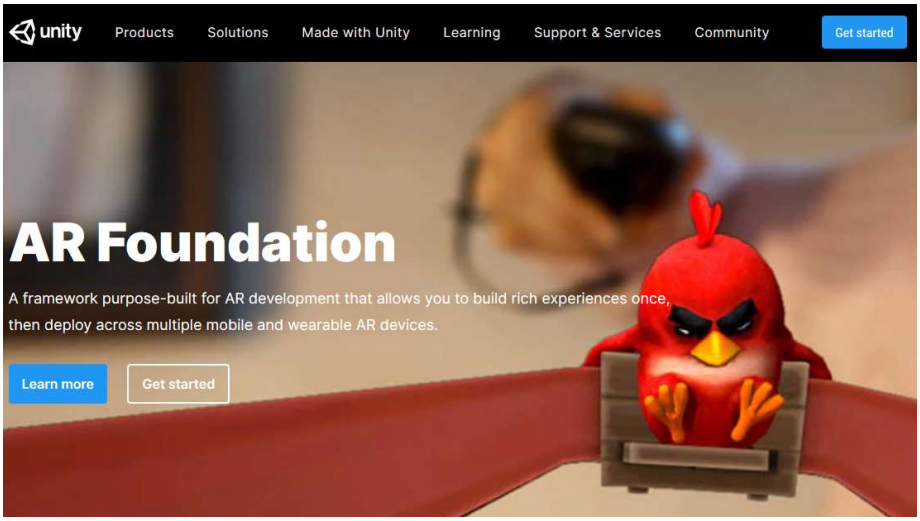
Afin de parvenir à rendre notre application compatible pour iOS et Android nous avons décidé de la migrer vers AR Foundation.

Qu'est AR Foundation et pourquoi avoir choisi cette technologie ?

AR Foundation est une couche software supplémentaire qui permet de rendre notre application compatible pour Android et iOS. Google (Android) dispose de son propre SDK pour ces appareils : ARCore, tandis que Apple (iOS) utilise ARkit.

Il faudrait donc deux applications distinctes si on désirait avoir une application qui marche sur ces 2 systèmes d'exploitation. AR Foundation permet de contourner ce problème.

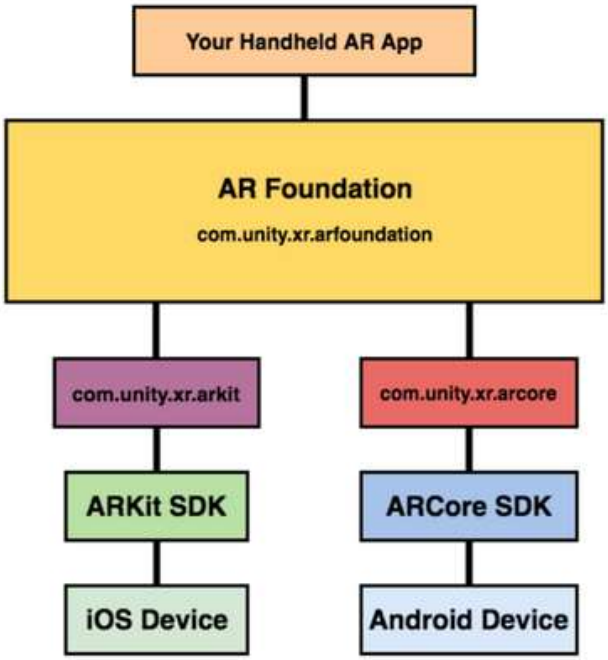
Figure 36 : Capture d'écran, AR Foundation



Source : https://unity.com/unity/features/arfoundation

AR Foundation fonctionne de la manière suivante :

Figure 37: Architecture AR Foundation



Source : https://www.credera.com/wp-content/uploads/2019/12/AR-Foundation.png

Ce Framework est capable de détecter quel système d'exploitation est installé sur un appareil. S'il détecte que c'est un appareil iOS, il utilisera ARkit pour l'application, tandis que si c'est un appareil Android ce sera ARCore. AR Foundation est un « pont » entre ces deux technologies rendant ainsi notre application compatible avec les deux systèmes d'exploitation.

Afin de corriger les problèmes évoqués plus haut, deux user story ont été ajoutées au Product Backlog afin qu'elles figurent dans ce sprint :

7	App Management	Utilisateur	pouvoir utiliser l'application dans n'importe quel smartphone	afin de ne pas être limité dans le choix de mes appareils
4	App Management	Utilisateur	que lorsque j'ai scanné un plan, le modèle qui apparait reste fixé au plan de manière stable	afin d'avoir une expérience fluide de l'application

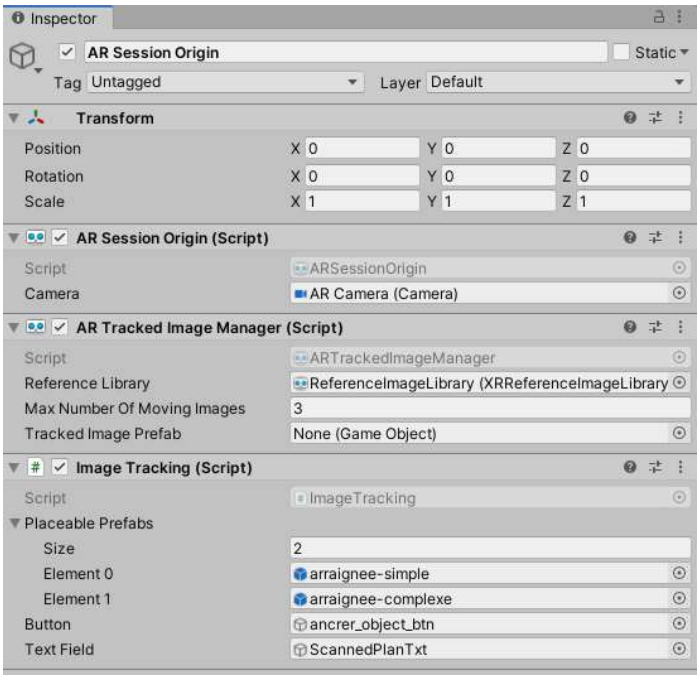
Durant ce sprint nous avons donc migré une première fonctionnalité vers AR Foundation. Le scan de plan. Tandis que l'autre fonctionnalité, l'ancrage d'objet se fera dans le sprint suivant. En termes de fonctionnement rien n'a changé, simplement que notre application est maintenant à 50% migrée vers AR Foundation. Le code utilisé jusque-là ne servait plus à rien il a fallu recommencer l'application de 0.

Point technique

D'un point de vue technique, voici comment AR Foundation fonctionne pour la fonctionnalité de scan.

Dans notre scène « ImageTracking » nous avons un objet appelé « AR Session Origin ». Cet objet s'occupe de tout ce qui est lié à la session de l'utilisateur, notamment l'accès à la caméra.

Figure 38 : Capture d'écran, Unity



Source : données de l'auteur

On peut voir ci-dessus qu'un script « Image Tracking » est attaché à cet objet. Comme son nom l'indique, c'est ce script qui traque les images qui passent devant la caméra de l'appareil.

AR Foundation rend très facile la manière de savoir si une image a été détectée ou non puisqu'il suffit d'accéder au paramètre « trackedImagesChanged » de l'objet « AR Tracked Image Manager ». Si une image est détectée et traquée elle est ajoutée à une liste, autrement elle est retirée de cette liste.

Figure 39 : code snippet, Image Tracking

```
  private void OnEnable()
    {
        aRTrackedImageManager.trackedImagesChanged += ImageChanged;
    }
```

Source : données de l'auteur

Une fois qu'il a détecté une image qu'il connaît, le programme fait appel à la méthode « UpdateImage » qui prend en paramètre l'image détectée (trackedImage).

Figure 40 : code snippet, Image Tracking

```
private void UpdateImage(ARTrackedImage trackedImage)
    {
        // Getting target image information
        string name = trackedImage.referenceImage.name;
        Vector3 position = trackedImage.transform.position;



        // Instantiating object
        GameObject prefab = spawnedPrefabs[name];
        prefab.transform.position = position;
        prefab.transform.rotation = trackedImage.transform.rotation;

        // change its local scale in x y z format
        prefab.transform.localScale = new Vector3(0.1f, 0.1f, 0.1f);
        prefab.SetActive(true);

        // Deactivating other objects
        foreach(GameObject go in spawnedPrefabs.Values)
        {
            if(go.name!= name)
            {
                go.SetActive(false);
            }
        }

        // Updating UI
        updateUI(name);


    }
```

Source : données de l'auteur

Avec l'objet « trackedImage », ci-dessus, on peut facilement avoir les informations dont on a besoin concernant l'image, c'est-à-dire sa position et son nom.

Cet objet nous permet de situer cette image dans l'espace en nous fournissant un Vector3, un objet de programmation avec lequel on peut accéder aux coordonnées de l'image dans l'espace avec ses trois coordonnées (X, Y, Z).
Une fois que nous avons stocké ces informations, nous allons ensuite trouver le modèle 3D correspondant à l'image, c'est-à-dire celui qui a le même nom et l'instancier à l'endroit où se situe l'image. Nous allons également le rétrécir afin qu'il ne paraisse pas trop grand à l'écran pour cette fonctionnalité.

Amélioration du scan

Pour améliorer la stabilité du scan de plan nous avons également essayé d'utiliser des QR codes. En effet, ceux-ci retournent des scores de qualité très élevés. Les images sont très contrastées ce qui permet au programme de les reconnaître beaucoup plus facilement, ce qui entraine une plus grande stabilité du programme.

Figure 41 : QR Code

Source :
https://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/QR_code_for_mobile_English_Wikipedia.svg/1200px-QR_code_for_mobile_English_Wikipedia.svg.png

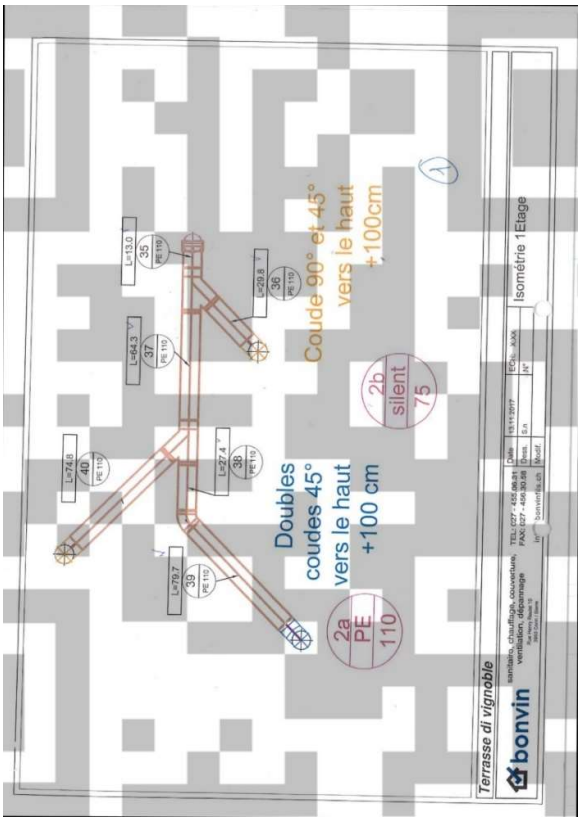
Néanmoins, lors du Sprint Review, les personnes qui nous accompagnent de l'Orif ont estimé que d'un point de vue pédagogique il était intéressant de conserver les plans de tuyauterie plutôt que d'utiliser de simples QR code. Il va donc falloir trouver une autre solution.

6.5 Sprint 4 : Finalisation migration et autres techniques d'amélioration du scan

Pour ce quatrième sprint, nous avons décidé de terminer la migration vers AR Foundation en passant la dernière fonctionnalité vers cette technologie également : l'ancrage d'objet au sol.
De plus, nous devons trouver une autre solution pour l'amélioration de qualité des images car les QR codes ne convenaient pas.

Pour se faire nous avons trouvé une idée intermédiaire, intégrer des QR code aux plans de tuyauterie :

Figure 42 : plan amélioré, arraignée simple



Source : données de l'auteur

Comme on peut l'apercevoir ci-dessus, de cette façon les tuyauteries restent visibles et nous parvenons à atteindre une très bonne qualité d'image de cette façon.

Figure 43 : qualité image, arraignée simple

Source : données de l'auteur

On peut voir que pour l'image ci-dessus la qualité atteinte est de 100%, ce qui est parfait.
Nous parvenons à atteindre une haute qualité d'image ainsi qu'à préserver l'aspect pédagogique de la lecture de plans par les étudiants, le QR Code étant partiellement transparent on parvient tout de même à lire les plans.

Point technique

Sur AR Foundation, l'ancrage au sol est découpé en plusieurs parties :

- Le scan de surfaces planes
- L'ancrage d'un objet au sol

Pour la première partie, nous disposons dans notre scène d'un objet appelé « AR Plane Manager »

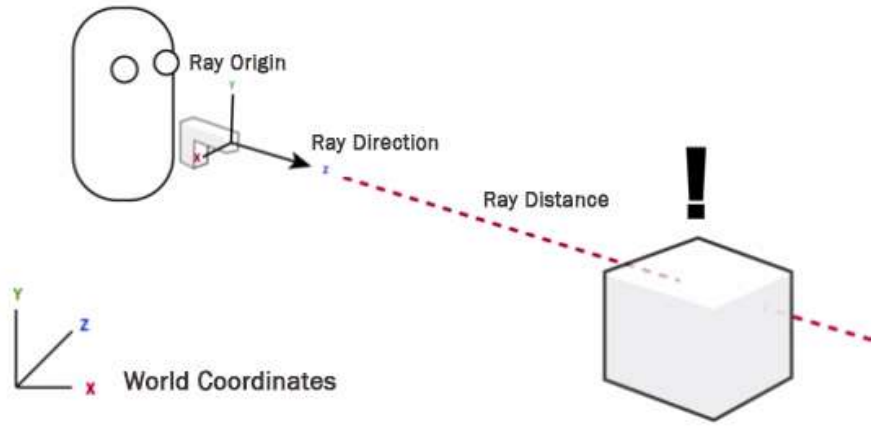
Figure 44 : Capture d'écran, Unity

Source : données de l'auteur

C'est un script qui s'occupe de la détection de surfaces planes. Ce script envoie un Raycast depuis le centre de notre écran jusqu'au sol. Il teste si la surface touchée est plane ou non et si c'est le cas il créé un AR Plane, un objet instancié virtuellement à cet endroit qui sera utilisé plus tard comme repère pour notre ancrage au sol.

Qu'est un Raycast ?

Figure 45 : Schéma raycast



Source : https://miro.medium.com/max/554/1*oXubxGtDjD6hXCi3VvDgxg.png

Un raycast est défini par un point de départ, une direction et une distance. C'est une ligne imaginaire dessinée à partir de ces paramètres. Si cette ligne, touche un objet on a en retour ce qu'on appelle un « hit ». Ce hit nous dit exactement ce qui a été touché et où (vector3 avec les coordonnées dans l'espace). C'est utilisé dans notre cas pour toucher le sol. Le SDK nous permet également ensuite de tester si ce « hit » est d'une surface plane ou non.

La deuxième partie de la fonctionnalité, l'ancrage au sol, fonctionne de la manière suivante :

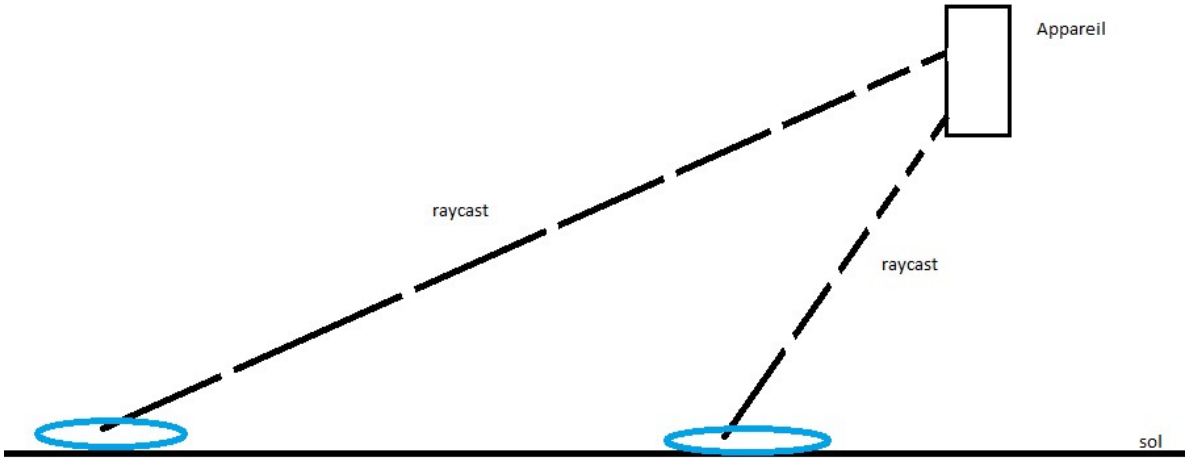
Figure 46 : Capture d'écran, Unity

Source : données de l'auteur

Il existe un objet dans notre scène appelé « ObjectSpawner » qui gère l'ancrage de nos objets. Un autre objet appelé « PlacementIndicator » gère le placement de notre helper pour l'interface utilisateur.

Le fonctionnement est simple, une fois que nous avons détecté des surfaces planes, nous faisons se déplacer notre helper sur celles-ci, en fonction du mouvement de la caméra.

Figure 47 : Schéma explicatif des raycasts



Source : données de l'auteur

L'appareil envoie des raycast au sol et selon la profondeur, le script déplace notre helper au point touché au sol, pour autant que celui-ci ait été détecté au préalable comme une surface plane.

Figure 48 : code snippet, méthode update

```
private void Update()
    {
        // shoot a raycast from the center of the screen
        List<ARRaycastHit> hits = new List<ARRaycastHit>();
        raycastManager.Raycast(new Vector2(Screen.width / 2, Screen.height / 2), hits,
TrackableType.Planes);

        // Check if we hit an AR plane, update the position and rotation
        if(hits.Count > 0)
        {
            if(enablePlanes==true)
                StartCoroutine(PlaneCoroutine(hits));

            transform.position = hits[0].pose.position;
            transform.rotation = hits[0].pose.rotation;


        }
    }
```

Source : données de l'auteur

Comme on peut le lire sur le code ci-dessus, les étapes sont dans l'ordre :

- Envoyer des raycast en continu depuis le centre de la caméra/écran.
- Si les éléments touchés sont considérés comme des « hits », des « Trackable » de type « Planes », mettre à jour la position du helper.

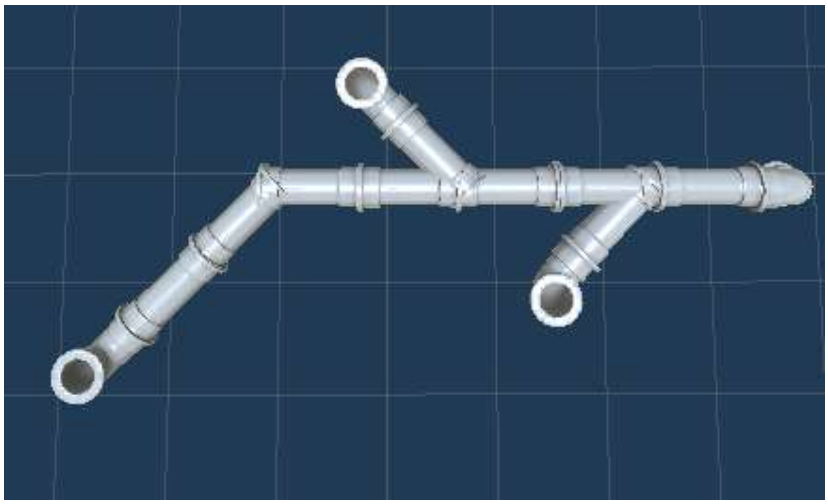
6.6 Sprint 5 : Interface graphique et technique de modélisation

Pour l'avant-dernier sprint, le but a été d'améliorer l'interface graphique qui jusque-là n'était que rudimentaire. Il a fallu également s'attaquer au problème de modélisation cité plus tôt. Nos modèles n'étant pas tout à fait fidèles à la réalité.

Nous avons également essayé d'améliorer la fonctionnalité d'ancrage au sol. En effet, les modèles qu'on ancre au sol sont à échelle réelle. Cependant, il n'est quand même pas facile de se faire une idée de leur échelle. Après une suggestion de Mr. Widmer, nous avons intégré un quadrillage à cette fonctionnalité. Les carrés sont de 1m x 1m ce qui aide à se rendre compte de l'échelle plus facilement, nous avons également intégré des inscriptions de grandeur à nos modèles.

La première chose à faire a été de recréer nos modèles 3D. Jusqu'à ce stade nous nous étions basés sur des assets gratuits trouvés sur internet. Malheureusement ceux-ci ne couvraient pas complètement nos besoins en design.

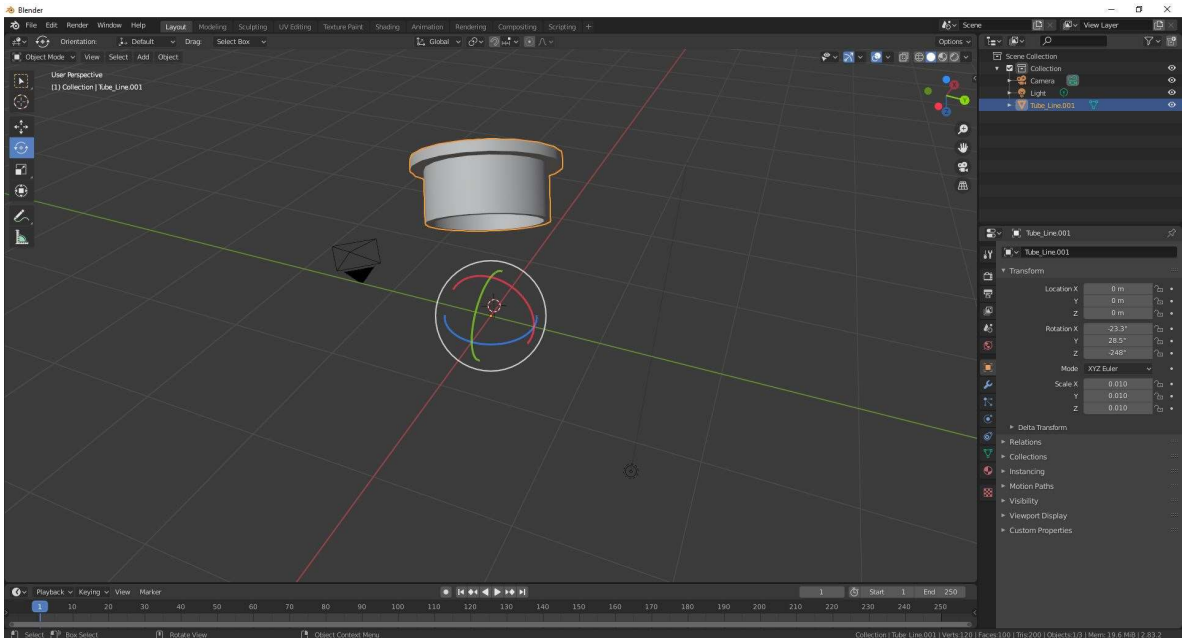
Figure 49 : modèle 3D araignée simple



Source : données de l'auteur

Afin de réaliser de nouveaux modèles, nous avons dû acquérir de nouvelles connaissances. En effet, nous n'avons pas de connaissances préalables en modélisation 3D. Afin d'avoir nos assets sur mesure nous avons dû utiliser Blender.

Figure 50 : capture d'écran, Blender



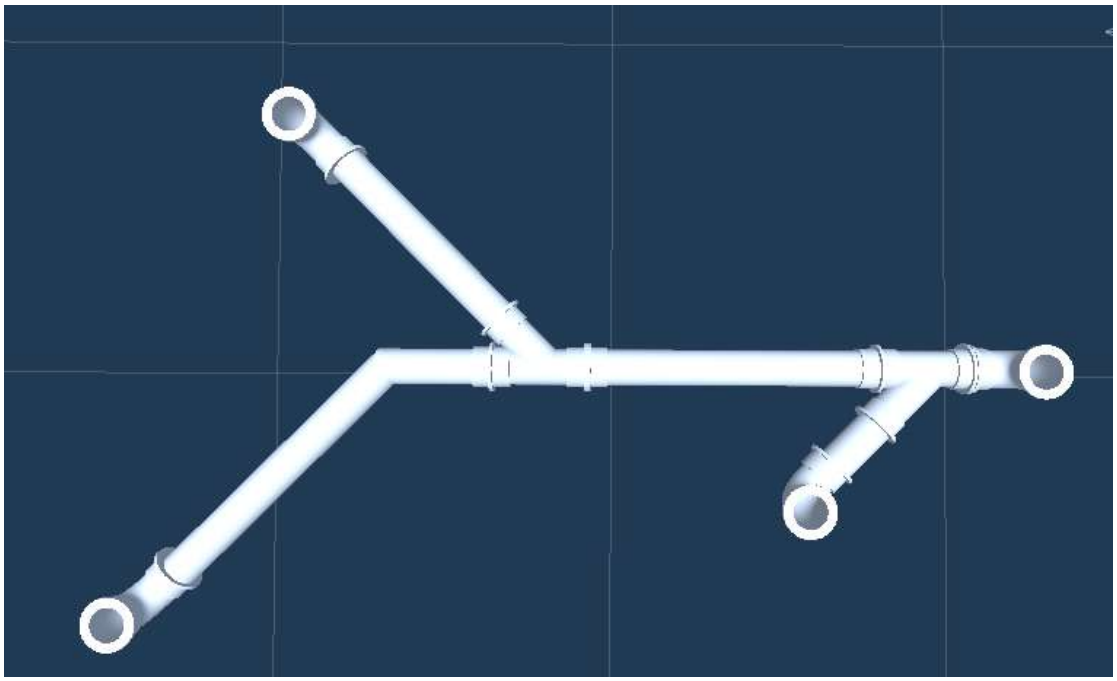
Source : données de l'auteur

<u>Qu'est Blender ?</u>

Blender est un logiciel de modélisation 3D. Il permet aux professionnels de créer les éléments graphiques nécessaires à la réalisation de jeux-vidéos, animations, etc.

Grâce à ce logiciel nous avons pu créer les éléments dont nous avions besoin pour créer nos tuyaux.

Figure 51 : modèle 3D, araignée simple



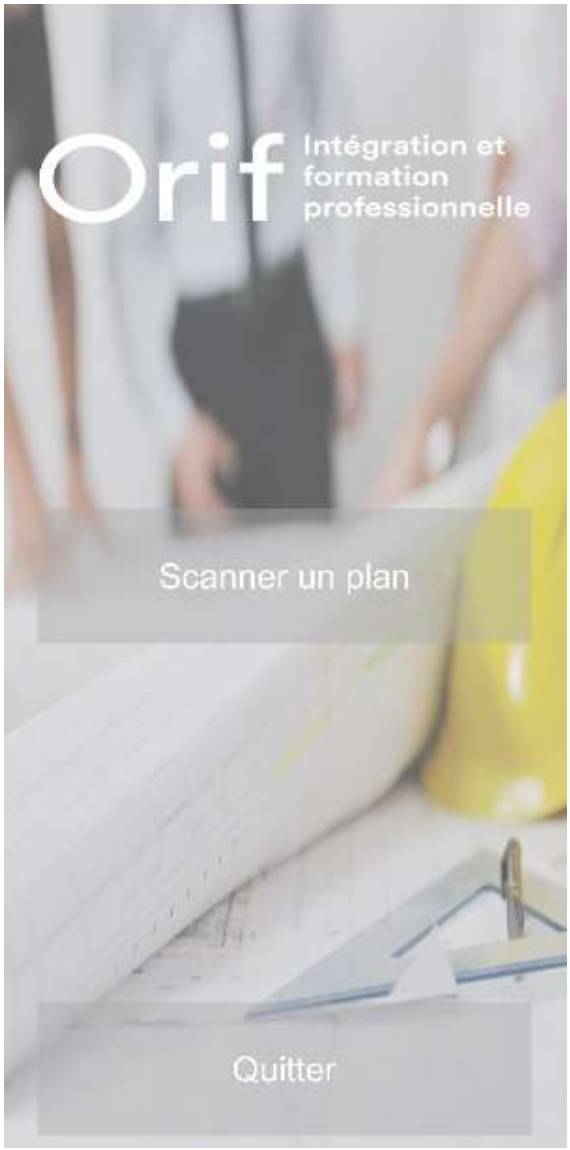
Source : données de l'auteur

Nos nouveaux tuyaux ressemblent à ceci. Les joints sont maintenant aux bons endroits et le reste du tuyau est « lisse ». Cet élément de tuyauterie correspond mieux à nos besoins et convient mieux également à l'apprentissage des étudiants de l'Orif.

Le dernier point à travailler durant ce sprint était l'interface utilisateur, dont voici le résultat :

- L'écran d'accueil :

Figure 52 : interface utilisateur, écran n°1



Source : données de l'auteur

Lorsqu'on appuie sur le bouton « Scanner un plan » de l'écran précédent on arrive sur l'écran qui permet de scanner des plans, celui-ci est en deux parties :

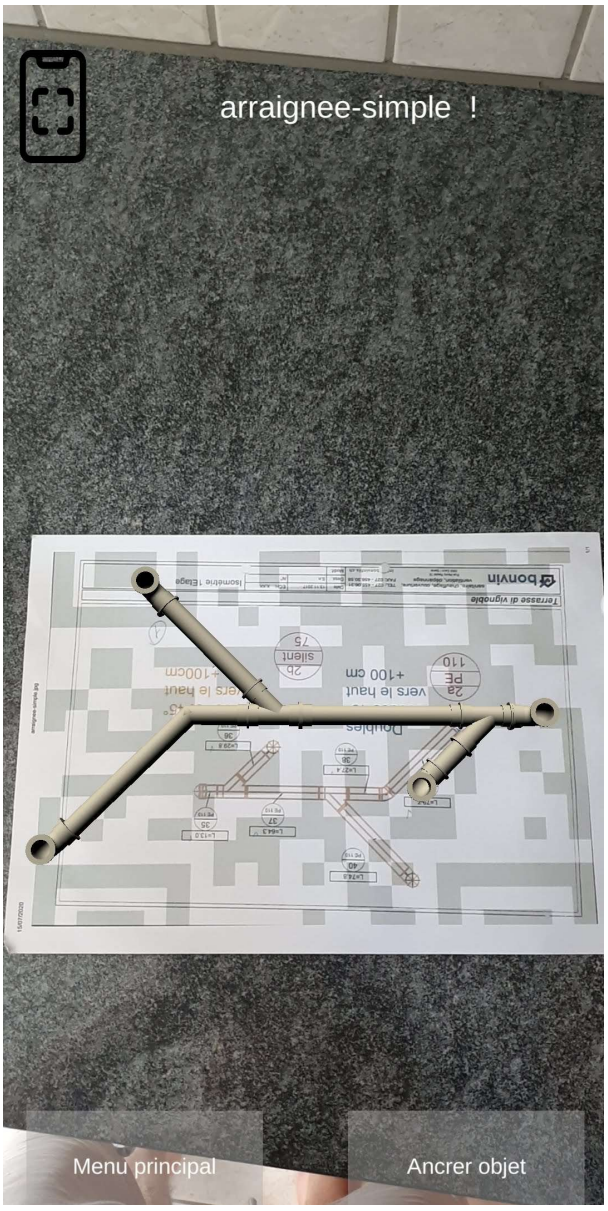
Figure 53 : interface utilisateur, écran n°2

Source : données de l'auteur

Dans un premier temps le programme est à la recherche d'images à reconnaître, à ce stade on ne peut que retourner au menu principal via un bouton ou juste approcher un plan devant la caméra afin que le programme le reconnaisse.

Une fois que le programme a reconnu un plan, il change et devient comme ceci :

Figure 54 : interface utilisateur, écran n°2



Source : données de l'auteur

Nous avons maintenant un bouton supplémentaire qui apparaît : « Ancrer objet ». Le texte en haut de l'écran a également changé et affiche le nom attribué au modèle 3D. Nous pouvons également apercevoir ce modèle s'afficher sur l'image au milieu de l'écran, sur le plan.

Si on appuie sur « Ancrer objet » on arrive sur le dernier écran, qui lui est aussi découpé en plusieurs parties.

Figure 55 : interface utilisateur, écran n°3

Source : données de l'auteur

Dans un premier temps, le programme est à la recherche de surfaces planes. Afin qu'il puisse les trouver il faut diriger sa caméra vers le sol ou vers toute autre surface plate, comme une table.

Une fois que le programme a détecté suffisamment de surface, l'interface utilisateur se met à jour afin de nous faire comprendre qu'il est prêt à ancrer un objet au sol.

Figure 56 : interface utilisateur, écran n°3

Source : données de l'auteur

Un Helper apparaît au milieu de l'écran pour nous aider à placer le centre de notre objet. Une fois qu'on est satisfait de son positionnement il nous suffit de cliquer sur l'écran pour que notre modèle 3D apparaisse.

Figure 57 : interface utilisateur, écran n°3

Source : données de l'auteur

Nous pouvons maintenant inspecter notre élément de tuyauterie dans tous les sens. On peut apercevoir également sur cette image le quadrillage au sol réalisé durant ce sprint, ainsi que les inscriptions sur nos divers éléments de tuyauterie. Ces inscriptions donnent la longueur exacte de chaque tronçon de tuyau.

7. Conclusion

Durant ce projet nous avons pu mener jusqu'au bout un développement de software sous la méthodologie SCRUM. Nous avons réalisé pour l'Orif le support software nécessaire afin de faciliter l'apprentissage de leurs apprentis.
Cependant, certains aspects de ce travail ont été plus difficiles abordés dû à la situation actuelle liée au Corona virus. En effet, nous avions pour but de faire tester l'application à des apprentis et d'avoir leur feedback avant la fin de l'échéance du projet, mais cela n'a pas été possible.

Selon l'Orif, cette application les aidera grandement en tant que support de cours supplémentaire pour leurs étudiants. Selon nous, le but est atteint. Nous sommes parvenus à aider des étudiants en difficulté d'apprentissage via la réalité augmentée.

D'un point de vue plus concret, ce projet a été une grande source de savoir puisque nous n'avions pas de connaissances au préalable liées à ce sujet. Nous avons appris à réaliser une application en réalité augmentée et à modéliser des objets en trois dimensions. Nous avons également eu une expérience concrète du développement software SCRUM.

Annexes

Annexe I : tutoriel modélisation

Tutoriel : Comment modéliser un nouveau plan et l'intégrer à l'application.

Ce tutoriel a pour but de permettre à une personne tierce de reprendre le projet et d'y ajouter de nouveaux plans. Ces plans pourront être scannés par l'appli, il sera aussi expliqué ici comment modéliser un plan en trois dimensions et intégrer le modèle à l'application.

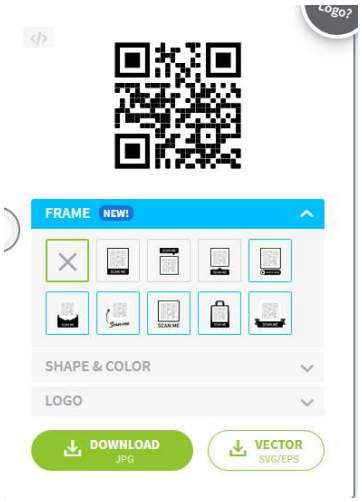
Première étape : intégrer un QR code à un plan afin que sa qualité d'image soie supérieure. Comme expliqué dans le rapport, il faut intégrer un QR code à un plan afin que la reconnaissance de celui-ci soit efficace.

Pour se faire, rendez-vous sur internet sur un site permettant de générer des QR codes. Pour notre part, nous avons utilisé :

https://www.qr-code-generator.com/

Ce site permet de générer des QR code à partir de texte, url ou toute autre information. Insérez un texte unique par exemple le nom du plan afin que le QR Code ne soit pas le même que celui d'un autre plan.

Figure 58 : qr code generator

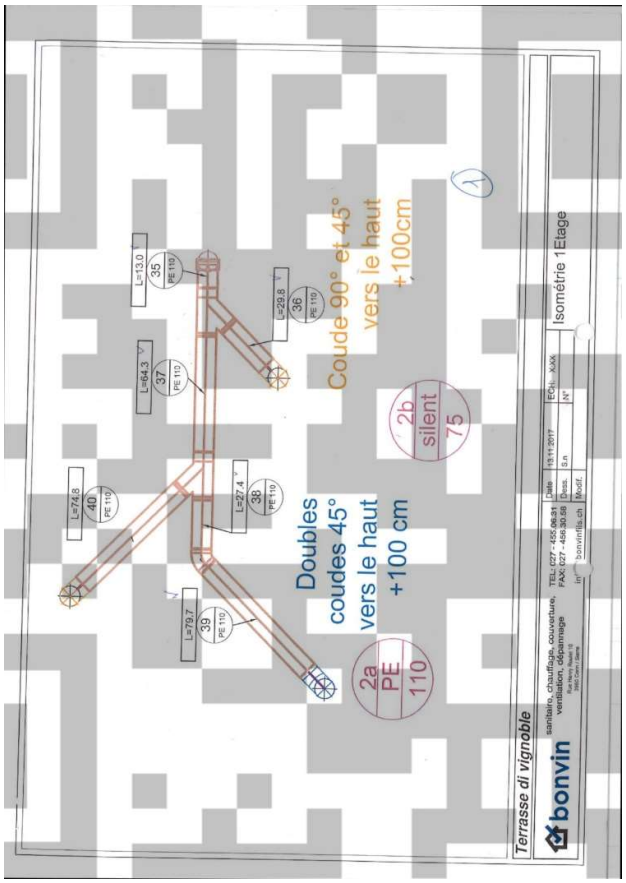


Source : https://www.qr-code-generator.com/

Une fois téléchargé, il faut intégrer le QR code à l'image du plan avec un logiciel de traitement d'image, pour notre part nous avons utilisé GIMP 2.0.

Le résultat final doit ressembler à ceci.

Figure 59 : plan-QR code araignée simple



Source : données de l'auteur

Ce n'est pas grave si le QR code ne figure pas complètement dans l'image, l'important est que les motifs (contrastes) se répètent tout au long de l'image.

Une fois que c'est fait, nous pouvons intégrer cette image à notre base de données. Ouvrez l'application sur Unity avec le .zip fourni avec le projet.

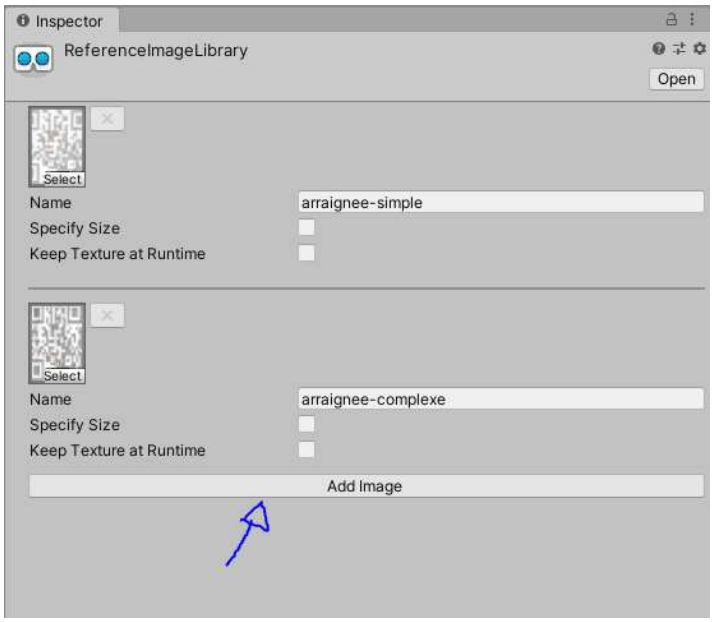
Figure 60 : capture d'écran, Unity

Source : données de l'auteur

Rendez-vous dans le dossier « Scripts » qui se trouve dans l'arborescence suivante : Assets > Scripts. Une fois à cet endroit, cliquez sur l'objet ReferenceImageLibrary comme sur l'image ci-dessus.

L'inspecteur s'ouvre à droite de l'écran et nous pouvons maintenant introduire notre nouveau plan.

Figure 61 : capture d'écran, Unity



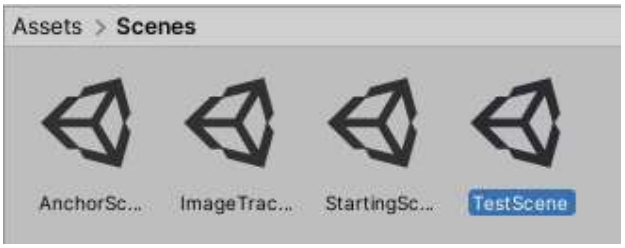
Source : données de l'auteur

Il faut donner un nom à notre image, à noter que ce nom doit être le même que pour le modèle 3D généré pour ce même plan afin que le programme fonctionne correctement.

Une fois que notre plan a été ajouté, nous devons maintenant modéliser nos tuyaux. Afin de savoir comment lire un plan pour le modéliser au mieux, référez-vous au chapitre « Préparation à la phase de développement » à ce sujet dans ce document.

Rendez-vous dans l'arborescence Assets > Scenes afin de commencer la modélisation, vous y trouverez une scène nommée « TestScene » qu'on peut utiliser pour cela.

Figure 62 : capture d'écran, unity



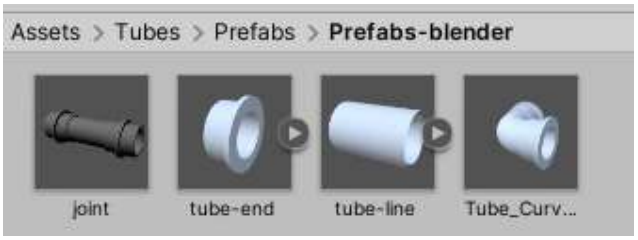
Source : données de l'auteur

Dans cette scène, nous devons créer 2 Prefabs afin qu'ils fonctionnent dans notre application. Les deux doivent avoir le même nom et par conséquent devront se trouver dans des dossiers différents. Ces différentes étapes sont expliquées ci-dessous mais nous avons également effectué une vidéo tutoriel disponible dans les annexes du projet.

Sans quitter la scène « TestScene », rendez-vous à l'arborescence suivante :

Assets > Tubes > Prefabs > Prefabs-blender

Figure 63 : capture d'écran, unity

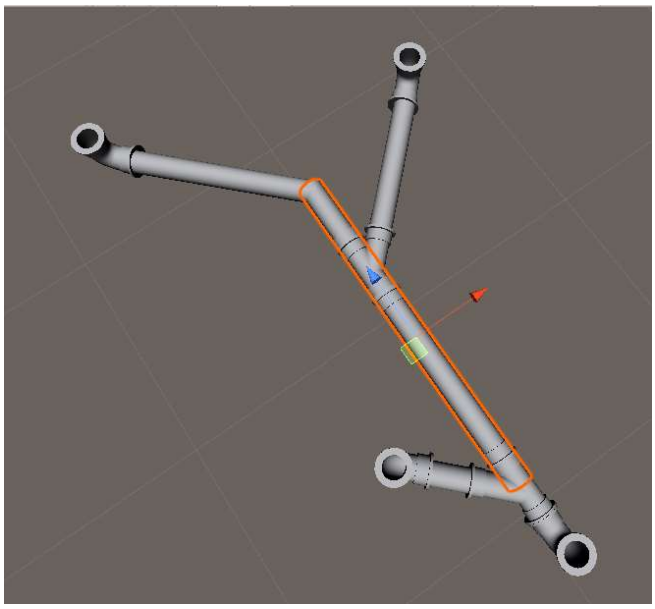


Source : données de l'auteur

A cet endroit, vous trouverez les différents éléments que nous avons créés sur Blender pour nous permettre de modéliser nos tuyaux.

Il faut les assembler sur la scène de manière qu'ils ressemblent à la forme voulue ainsi qu'ils atteignent la bonne échelle :

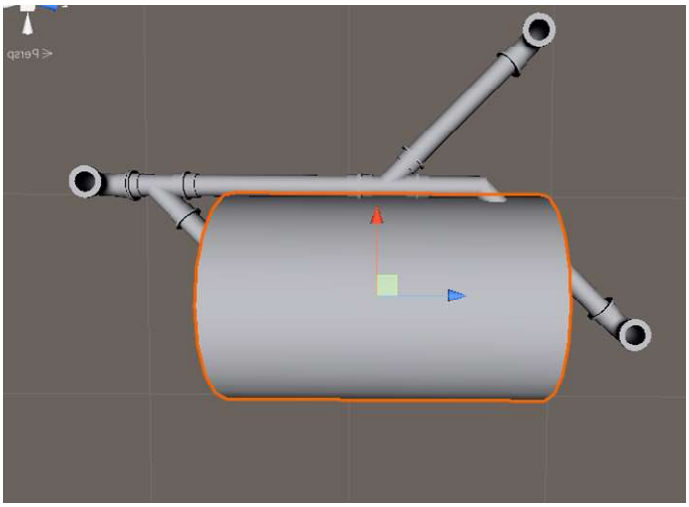
Figure 64 : capture d'écran, unity



Source : données de l'auteur

A noter que 1 carré dans l'éditeur correspond à 1m x 1m dans la réalité.

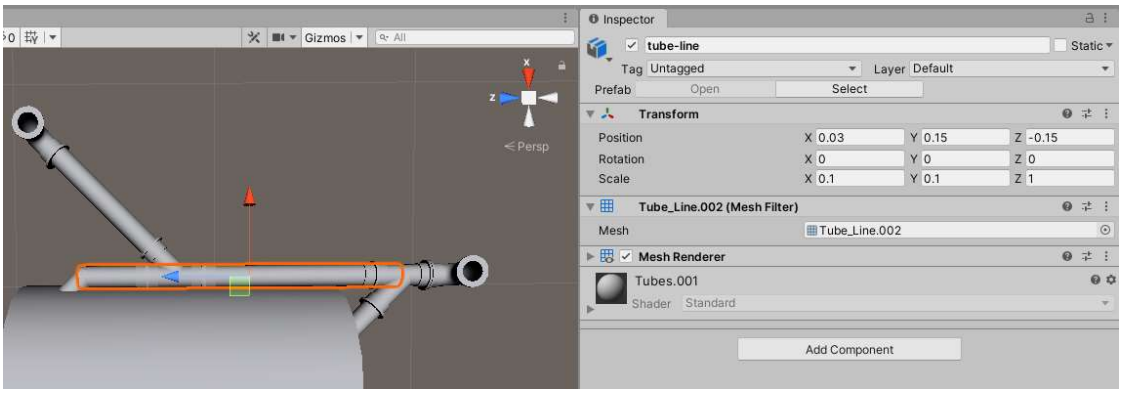
Figure 65 : capture d'écran, unity



Source : données de l'auteur

Un élément tube line par exemple ressemblera à ceci lorsque vous l'afficherez sur la scène, il faut modifier ces échelles dans l'inspecteur afin qu'il corresponde à ce qu'on veut.

Figure 66 : capture d'écran, unity



Source : données de l'auteur

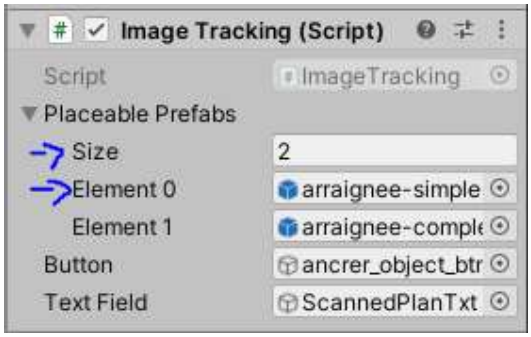
Par exemple, pour lui donner 10cm de diamètre, il faut se rendre dans l'inspecteur de l'objet et baisser les scales X et Y à 0.1, l'échelle Z gérant sa grandeur, il faut également la modifier afin d'atteindre la grandeur voulue, laisser la valeur 1 sur l'échelle Z, correspond à environ 1m70.

Cette étape est également détaillée à travers une vidéo jointe aux annexes du projet, merci de vous y référer pour avoir de plus amples explications.

Une fois que la modélisation est terminée, vous avez créé 2 prefabs, un pour l'image tracking et un pour l'ancrage d'objet.

Afin qu'ils soient fonctionnels, il suffit d'aller dans la scène « ImageTrackingScene », cliquer sur l'objet « AR Session Origin », l'inspecteur s'ouvre sur le côté droit et on n'a plus qu'a incrémenter la taille du « Placeable Prefabs » de 1 et d'y introduire le prefab pour la tracking scene.

Figure 67 : capture d'écran, unity



Source : données de l'auteur

Pour l'ancrage d'objet, c'est exactement le même procédé, il suffit de se rendre dans la scène « AnchorScene », cliquer sur l'objet « ObjectSpawner », incrémenter la taille de « Prefabs To Spawn » de 1 et d'y introduire le prefab pour l'ancrage d'objet.

Figure 68 : capture d'écran, unity

Source : données de l'auteur

Le tutoriel est terminé, nous avons intégré un nouveau plan à l'application ainsi que ses modèles 3D.

Annexe II : Product Backlog

Hes·so VALAIS WALLIS
Haute École Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

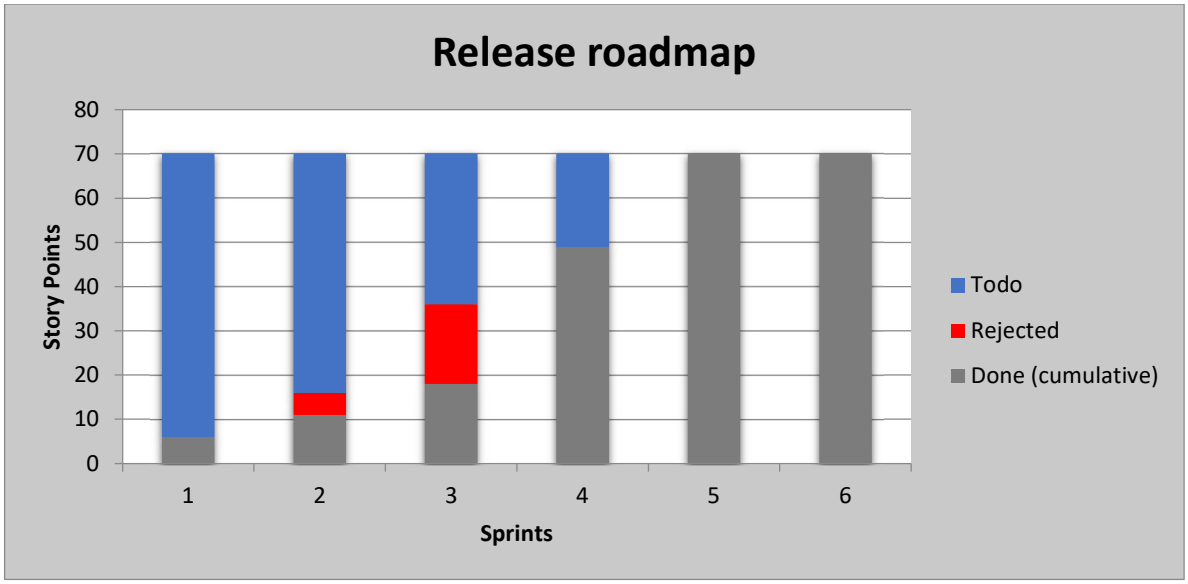
Nelson Ribeiro Teixeira

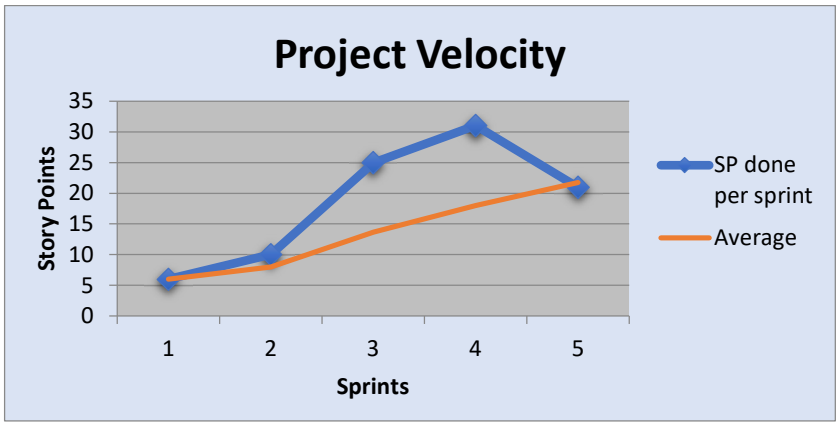
TB : ORIF AR

US Nr	Theme	User Stories			Acceptance Criteria	Status	Story Points	Sprint	Comment
		En tant que...	... je veux ...	... afin de ...					
1	App Management	Utilisateur	pouvoir insérer un plan	afin que le programme puisse le reconnaitre	BDD avec les plans (jpg)	3	1	1	
3	Feature	Utilisateur	scanner un plan déjà connu par le programme	d'avoir une version 3D de celui-ci	Afficher le nom d'un plan scanné	3	3	1	
10	Feature	Utilisateur	que le modèle 3D généré par l'ordinateur soye placé au bon endroit	pouvoir mieux l'observer	Modèle 3D placé sur le plan (physique)	3	2	1	
7	App Management	Utilisateur	pouvoir utiliser l'application dans n'importe quel smartphone	afin de ne pas être limité dans le choix de mes appareils	AR Foundation	3	18	3	
4	App Management	Utilisateur	être capable de pouvoir lire un plan que l'appllication scanne	afin de pouvoir reconnaitre ce qui est dessiné sur le plan	Amélioration scan QR code	3	2	3	
5	User Management	Utilisateur	avoir une interface graphique user friendly	afin d'être capable d'utiliser l'application facilement	UX	3	8	5	
6	User Management	Utilisateur	avoir une expérience d'utilisation fluide	afin d'être dans les meilleures conditions d'apprentissage possibles	Amélioration scan QR code	3	13	4	
9	User Management	Utilisateur	pouvoir observer des modèles 3d correspondant au mieux à la réalité	afin d'apprendre plus facilement la lecture de plan et l'assemblage de tuyaux	Blender elements	3	13	5	
11	Feature	Développeur	J'aimerais être capable de reprendre le projet facilement	afin de pouvoir ajouter des nouveaux plans et modèles 3D à l'application	Tutoriel	3	2	2	
12	Feature	Utilisateur	J'aimerais avoir un modèle affiché à l'echelle (1:1)	afin de pouvoir me faire un ordre d'idée de la taille réelle de l'objet	Ancrage d'objet	3	3	2	
13	Feature	Utilisateur	J'aimerais pouvoir ancrer un objet au sol	afin de pouvoir tourner autour et l'inspecter	Ancrage d'objet	3	5	2	
							70		

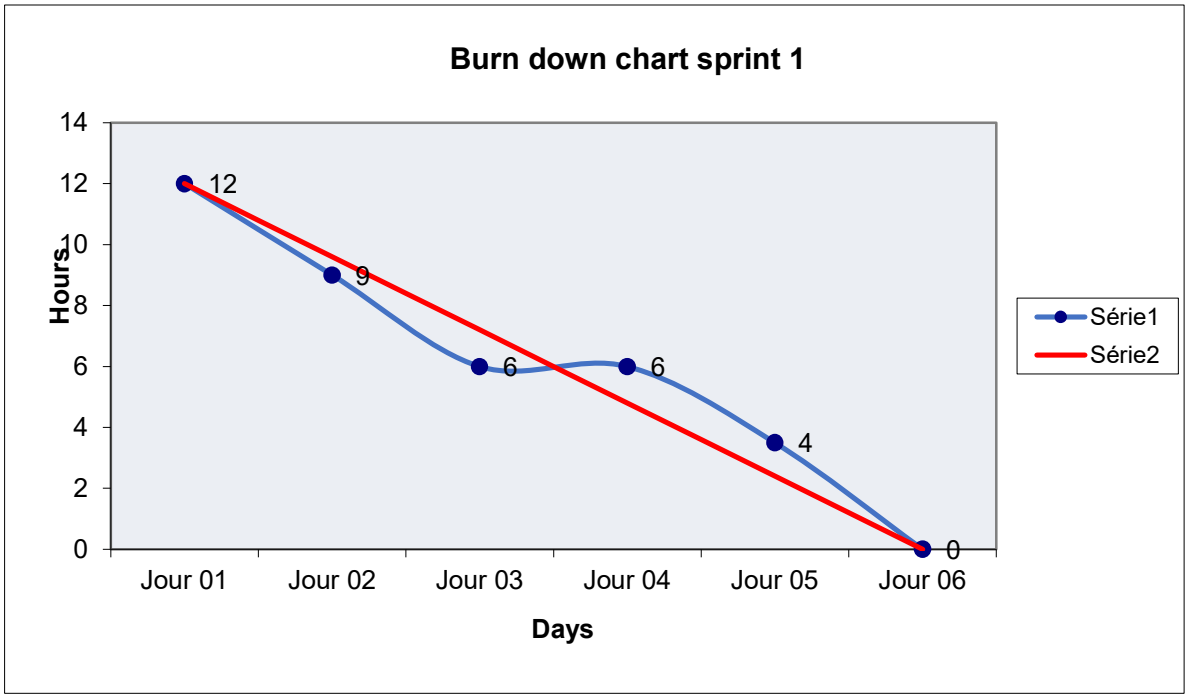
Annexe III : Release roadmap and project velocity

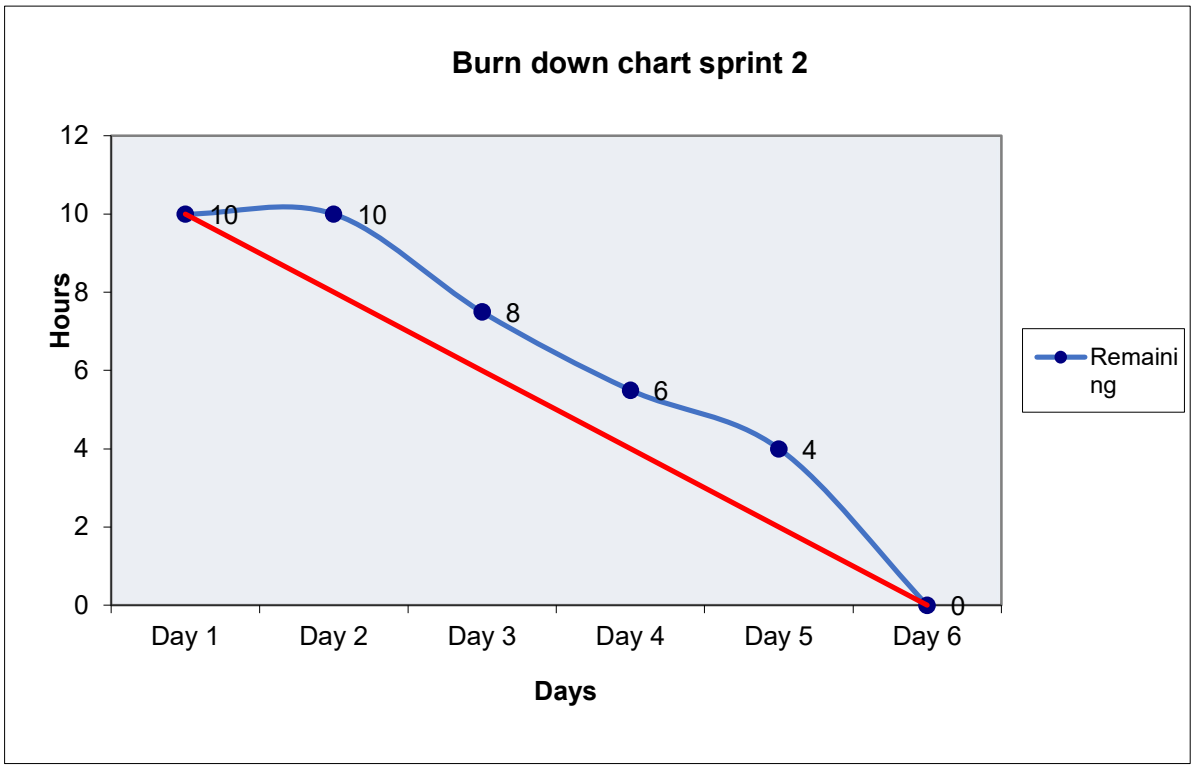
	Sprints					
	1	2	3	4	5	6
SP done per sprint	6	10	25	31	21	0
Average	6	8	13.7	18	21.75	19.3
Todo	64	54	34	21	0	0
Rejected	0	5	18	0	0	0
Done (cumulative)	6	11	18	49	70	70

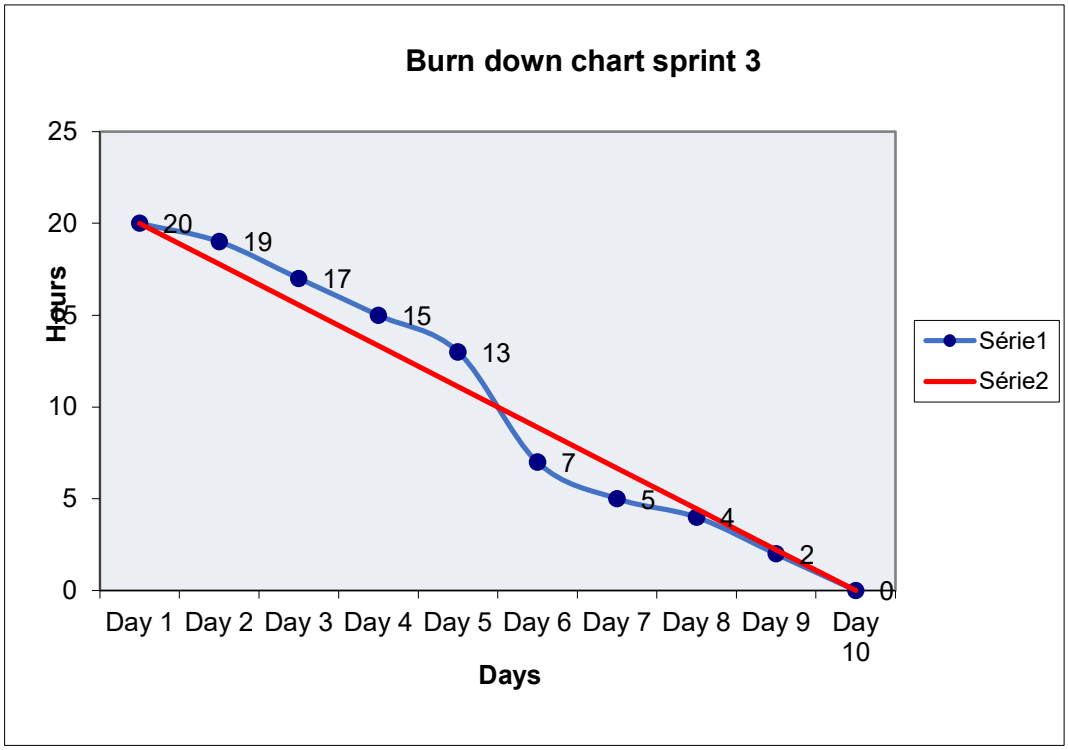


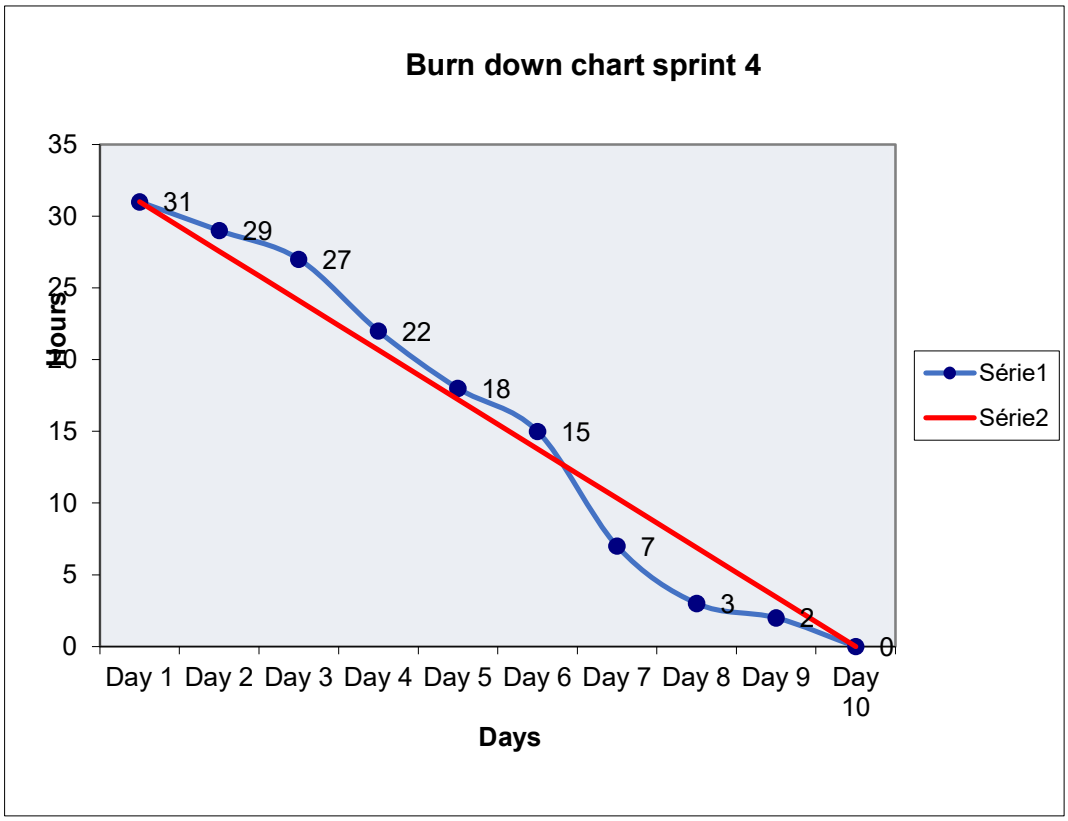


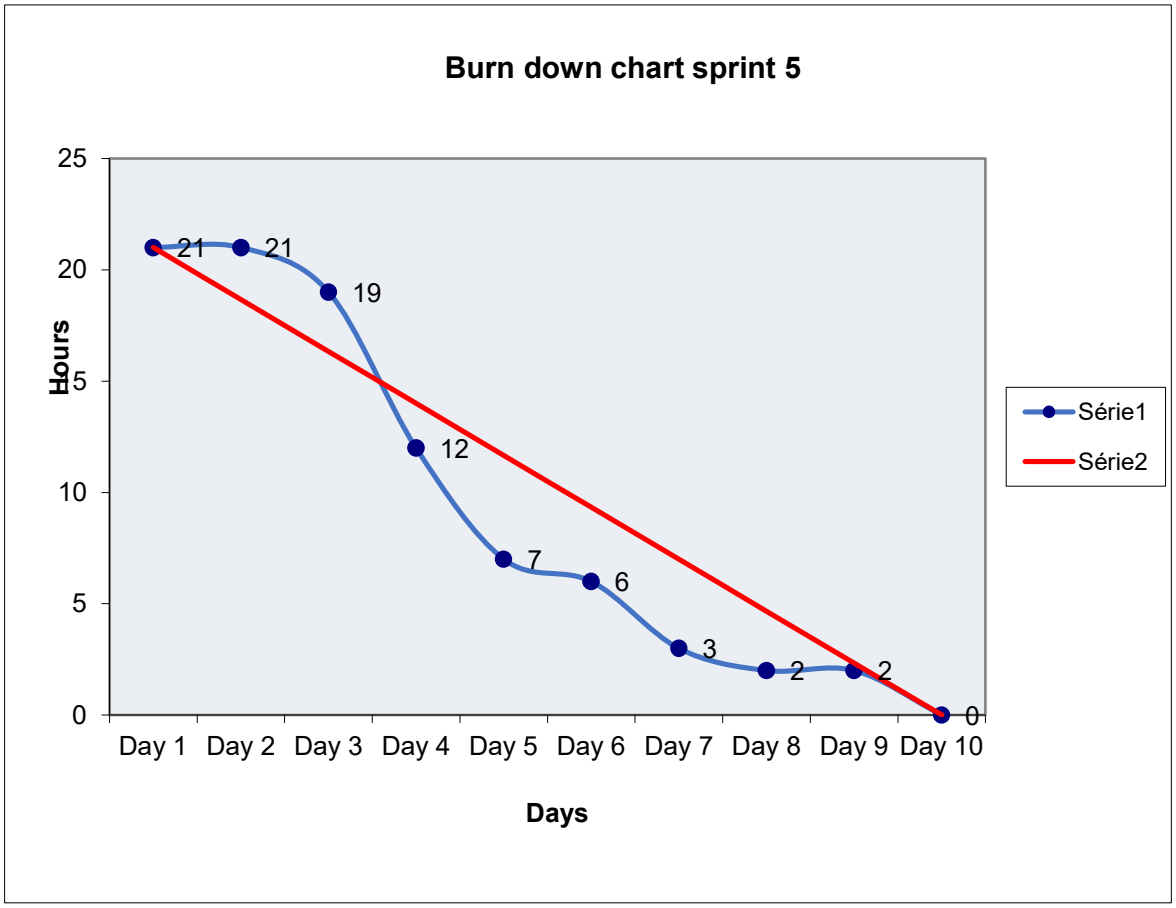
Annexe IV : Burndown charts






Burn down chart sprint 3
25
20
15
10
5
0
Hours
20
19
17
15
13
7
5
4
2
0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Day 8
Day 9
Day 10
Days
Série1
Série2


Burn down chart sprint 4
35
30
25
20
15
10
5
0
Hours
31
29
27
22
18
15
7
3
2
0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Day 8
Day 9
Day 10
Days
Série1
Série2


Burn down chart sprint 5
25
20
15
10
5
0
Hours
21
21
19
12
7
6
3
2
2
0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Day 8
Day 9
Day 10
Days
Série1
Série2

Références

SEEABLE. (date pas indiquée). *Scan > model > visualize in augmented reality*. Consulté le mai 06, 2020, sur https://seeable.co.uk/scan-model-visualise-augmented-reality/.

Eduardo Souza. (2019, avril 14). 9 *Augmented Reality Technologies for Architecture and Construction*. Consulté le mai 06, 2020, sur https://www.archdaily.com/914501/9-augmented-reality-technologies-for-architecture-and-construction/ .

Darfdesign. (date pas indiquée). *An Augmented Reality Application to Visualise Architecture*. Consulté le mai 06, 2020, sur https://www.darfdesign.com/arki-pses.html/.

Susan Fourtané. (2019, avril 15). *Augmented Reality: The Future of Building*. Consulté le mai 08, 2020, sur https://interestingengineering.com/augmented-reality-the-future-of-building/ .

Construction Citizen. (2019, avril 14). *How Augmented Reality Can Enhance Your Building Projects*. Consulté le mai 08, 2020, sur https://constructioncitizen.com/blog/how-augmented-reality-can-enhance-your-building-projects/1904241/ .

Vuforia . (date pas indiquée). *About us*. Consulté le mai 09, 2020, sur : https://developer.vuforia.com/

Wikipédia . (2019, septembre 12). *Vuforia Augmented Reality SDK*. Consulté le mai 09, 2020, sur : https://developer.vuforia.com/

Wikipédia. (2020, juillet 24). *Unity (game engine)*. Consulté le mai 13, 2020, sur https://en.wikipedia.org/wiki/Unity_(game_engine) .

Wikipédia. (2020, juin 02). *Scrum (développement)*. Consulté le mai 18, 2020, sur https://fr.wikipedia.org/wiki/Scrum_(développement) .

Wikipédia. (2020, avril 15). *Réalité augmentée* . Consulté le mai 18, 2020, sur https://fr.wikipedia.org/wiki/Réalité_augmentée .

ViewSonic. (2019, septembre 26) *6 Benefits and 5 Examples of Augmented Reality in Education*. Consulté le mai 18, 2020, sur https://www.viewsonic.com/library/education/6-benefits-and-5-examples-of-augmented-reality-in-education/ .

Unity. (date pas indiquée). *AR Foundation* . Consulté le juillet 06, 2020, sur https://unity.com/unity/features/arfoundation/

Hagry, C. (2012*). Les enjeux de la réalité augmentée*. Haute école de gestion de Genève(HEG-GE).