Université de Nantes

Rapport de Stage

Développement d'un module de Réalité Augmenté

Réalisé par :
Allan RENOU

Encadrant :
Sebastien CAZALAS
Maitre de Stage
Vincent AUGERAU

13 juillet 2014

Remerciements

Je tiens à remercier Vincent Augereau pour son accueil au sein de l'entreprise Mobile Developpement, ainsi que pour les conseils et l'aide qu'il m'a apporté durant ce stage.

Je tiens également à remercier M. Cazalas pour son suivie et pour les conseils qu'il m'a donné tout au long de ces 10 semaines de stage.

Résumé

J'ai réalisé mon stage dans le société Mobile développement, une agence numérique spécialisé dans la conception d'applications mobiles. Mon rôle consistait à concevoir un module de réalité augmentée qui pourra être intégré dans une application de l'entreprise. Pour cela, j'ai utilisé le SDK métaio. Grâce à cela, j'ai pu créer le module en java, qui est le langage natif d'android. J'ai également comparé ce module en java, avec une version en JavaScripte afin de pouvoir utiliser le meilleur langage. D'autres missions m'ont été confiées, tel que la réalisation de graphiques en php, ou la conception d'une application destinée à un client.

Abstract

I did my internship in the company Mobile Development, a digital agency specializing in the creation of mobile applications. My job was to create an augmented reality module that can be integrated into an enterprise application. I used the SDK metaio. I could create the java module, which is the native language of android. I also compared this module in java with a version Javascript in the browser in order to use the best language. Other missions were entrusted to me as the realization of chart php, or creating an application for a client

Table des matières

1 Introduction

Dans le cadre du DUT informatique, j'ai été amené à faire un stage en entreprise pour une durée de 10 semaines à compter du 14 avril 2014. Celui-ci a pour objectif de confronter le savoir que j'ai acquis lors de ma formation à l'IUT, aux réalités de l'entreprise.

J'ai effectué mon stage dans l'entreprise Mobile Développement, à Doué-la-fontaine (49). Cette entreprise est une SARL, créée en octobre 2010 par Vincent AUGERAU, qui est son dirigeant. Elle est spécialisée dans le développement d'applications mobiles, notamment, des applications pour Android et IOS [1].

J'ai réalisé mon stage dans cette entreprise car j'ai déja eu l'occasion de créer des applications pour smartphone, d'une part dans le cadre de mon projet de première année, puis, par la suite, à des fins personnelles. J'ai trouvé le développement de ce type de programme très interessant, j'ai donc naturellement été séduit par la spécificité de l'entreprise.

Ma mission consistait à développer un module utilisant la réalité augmentée. Ce module sera réutilisé à la suite de mon stage, permettant ainsi à l'entreprise de proposer à ses clients, l'utilisation de la réalité augmentée dans leurs applications. Pour cela j'ai donc dû réaliser des recherches sur ce qu'était la réalité augmentée afin de bien comprendre le sujet ainsi que les tâches que j'aurais à effectuer. J'ai ensuite recherché ce qui existait déjà, et comment il était possible de réaliser l'application. J'ai enfin dû intégrer cette fonctionnalité à l'application, en tenant compte des contraintes liées aux différents type smartphones et de tablettes utilisées. J'ai du comparer les fonctionnalités proposées en langage natif (en java pour Android) et en JavaScript.

Ce rapport se divise en cinq parties :

— Dans un premier temps, je vais vous présenter l'entreprise Mobile Développement, ainsi que l'environnement dans lequel j'ai réalisé ce stage.

1. Android et IOS sont 2 systèmes d'exploitations pour smartphone. Ces 2 logiciels sont les plus utilisés actuellement. Android étant le systèmes des smartphone des marques HTC, Samsung, ... , et IOS est celui utilisé sur les IPhones

— Ensuite, nous verrons en quoi consistait ma mission,et quel en était l'origine
— Dans un troisième temps, nous allons voir les étapes par lesquelles je suis passé pour mettre à bien ma mission.
— Nous verrons le résultat de mon travail, les avantages de cette solution, et ce qui aurait pu être amélioré.
— J'aborderai ensuite les différentes missions qui m'ont été confiées en parallèle.

2 Environnement

2.1 L'Entreprise Mobile développement

2.1.1 Présentation de l'entreprise

Mobile Développement est une SARL, créée par Vincent Augereau en 2010. Elle concentre son activité dans le développement d'applications mobiles (pour les systèmes Android et IOS). Elle développe des applications pour tous les secteurs d'activités, malgré qu'elle soit spécialisée dans le tourisme. En effet, dès sa création, Mr Augereau a orienté son entreprise vers le secteur du tourisme avec la création de la marque "Touristic' Tour".

Figure 1 : Application Doué la fontaine Touristic tour et radio Monaco

2.1.2 Secteur touristique

La marque Touristic' Tour englobe plusieurs applications (par exemple Saumur Touristic' Tour). Les applications de cette marque proposent à ses utilisateurs de suivre des circuits touristiques en fonction du lieu où ils se trouvent. Pour cela, elle utilise la fonction GPS afin de localiser l'utilisateur, elle affiche ensuite sa position sur une carte ainsi que le circuit à suivre, et les différents points d'intérêts (POI) qu'ils proposent. Un point d'intérêt (POI) est généralement un lieu touristique à visiter, mais il peut également s'agir d'un hôtel, d'un commerce, d'un restaurant, ...

Pour chaque point d'intérêt, elle leurs propose des "fiches descriptives " des lieux à visiter, des photos, un lien vers la page wikipedia, si elle existe, ... De plus, cette application affiche également des informations pratiques, tels que des restaurants et des hébergements.

La marque "Touristic' Tour" ne s'arrête pas là, en effet elle est également à l'origine du logiciel se trouvant sur des bornes tactiles. Celles ci fournissent des informations touristiques sur la région.

2.1.3 Les autres secteurs

Bien que le tourisme occupe une partie des activités de Mobile Développement, cette société ne se limite pas à cela. Elle crée des applications pour IPhone et Android pour tous secteurs d'activités tel que la radio, avec les applications "RADIO MONACO" ou "COLLINES LA RADIO".Qui proposent d'écouter la radio en utilisant internet. Mobile Développement a également réalisé des applications pour le secteur bancaire, avec son application "CA Scan", qui permet, sans se déplacer, de transmettre vos justificatifs obligatoires d'identité et de domicile au Crédit Agricole. L'agence Crédit Agricole se chargera ainsi de mettre à jour votre dossier client.

2.2 Environnement physique

Mobile développement étant une petite entreprise, nous étions tous dans la même pièce. Cette pièce était à l'origine un appartement, qui a été aménagé afin que nous puissions tous travailler dans de bonnes conditions. L'entreprise ne possède pas d'autres batiments car étant un prestataire de service, elle n'a pas besoin d'endroit où stocker des marchandises, ou d'un endroit où accueillir les clients qui soliciteraient les services de l'entreprise. Cet appartement se situe dans une rue où se trouve principalement des habitations, et non dans une zone active économiquement.
Toutes les phases de réalisation d'un logiciel avaient lieu ici, une table était à notre disposition pour les phases théoriques qui sollicitaient plusieurs personnes. Nous l'utilisions également pour les réunions que nous faisions tous les lundis afin de faire le point sur l'avancement des différents projets, et pour fixer les objectifs de la semaine pour chaque personne.
Pour mener à bien mon projet, j'avais à ma disposition un bureau, à quelques mètres de celui de Vincent, mon maître de stage. Guillaume, un développeur de l'entreprise, travaillait également en face de moi. Je pouvais ainsi facilement demander de l'aide si un problème trop complexe m'empéchait d'avancer.

2.3 Environnement informatique

Afin de réaliser le module de réalité augmentée, j'avais à disposition un PC équipé de Windows Vista, auquel j'ai dû ajouter Eclipse et le SDK Android afin de pouvoir développer le module dans un environnement adapté. J'avais également à ma disposition différents smartphones ou tablettes, de taille d'écran et de version d'Android différents. Cependant, j'ai majoritairement travaillé avec une tablette samsung équipé d'un système Android 4.2.2. En effet, la réalité augmentée nécessitant d'utiliser le GPS, l'accéléromètre et la caméra, il était impossible de développer le module sur l'émulateur proposé par le SDK Android.
Lors de mon stage, j'ai également dû tester le module que je développais en conditions réelles. Pour cela, j'avais la possibilité de me rendre à des ruines qui se situaient à quelques mètres de l'entreprise, afin de tester la reconnaissance d'image et la façon dont les éléments virtuels s'intégraient aux ruines

3 Mission

3.1 Présentation

Ma mission consistait à développer un module de réalité augmentée. Ce module permet à l'entreprise de proposer à ses clients d'intégrer la réalité augementée dans leur application. Cette nouvelle fonction permettra à l'utilisateur des application Touristic' Tour, de suivre le circuit proposé par l'application à l'aide d'un système de réalité augmentée. Pour cela, l'application utilisera la fonction caméra du smartphone pour afficher ce qui se trouve devant l'utilisateur, et elle ajoutera à l'image des indications relatives au circuit (par exemple une flèche), ou aux lieux (nom, description, photo, lien vers une page wikipedia, ...).

3.2 Origine du besoin

Le besoin d'un tel module n'est pas né d'une demande explicite d'un client, mais en réponse à des demandes faites lors de salons. En effet, l'entreprise aimerait pouvoir proposer d'intégrer la réalité augmentée dans les applications de ses futurs clients, afin de se distinguer de la concurence.

3.3 Expression fonctionnelle du besoin

3.3.1 Fonction principale

La fonction principale de ce module est d'afficher des points d'intérêts (POI) proche de l'utilisateur. Pour cela, il faudra intégrer des éléments 2D (Images, vidéos, texte, ...) ou 3D au rendu de la caméra. Le rendu sera effectué sans intervention de l'utilisateur, uniquement avec des informations de l'environnement (position géographique et image de la caméra).

3.3.2 Fonctions complémentaires

Afin de rendre l'application plus complête, il faudrait rendre possible l'affichage de la distance qui sépare l'utilisateur de chaque POI. Il faudrait également que pour chaque point soit associé une image représentant la catégorie de ce point (hotel, restaurant, lieu touristique, ...). Enfin il faudra que le chemin que l'utilisateur doit suivre apparaisse à l'écran.

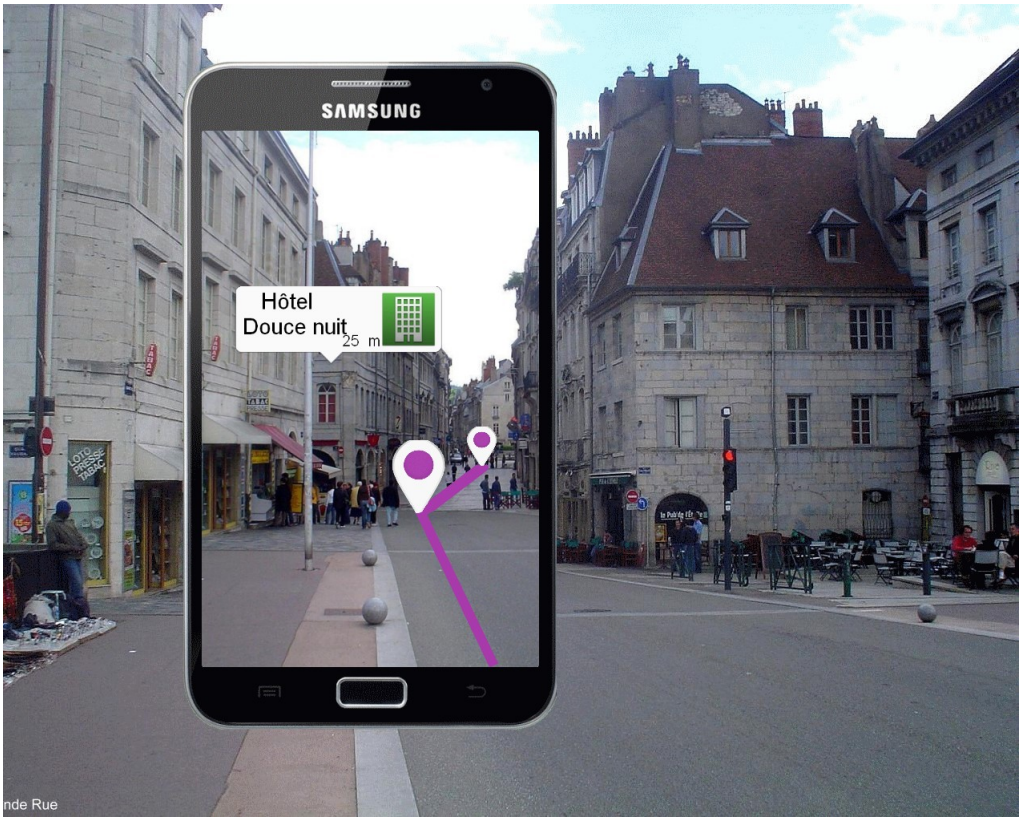


Figure 2 : Schema rendu souhaité

4 Travail effectué

4.1 Recherches préliminaires

Dans un premier temps, j'ai effectué des recherches sur ce qu'était vraiment la réalité augmentée, afin de pouvoir bien identifier le problème. Cela m'a amené à distinguer plusieurs types de réalité augmentée. Et enfin, j'ai cherché comment réaliser une application. Pour cela, j'ai listé les différents SDK [1] et API [2] qui existaient.

4.1.1 Définition de la Réalité Augmentée

Définition : C'est un système informatique qui rend possible la superposition d'un modèle virtuel (3D ou 2D) à une image filmée en temps réel grâce à un l'appareil photo d'un téléphone, ou à des "lunettes vidéos spéciales". Elle s'applique aussi bien à la perception visuelle (superposition d'images virtuelles aux images réelles) qu'aux perceptions proprioceptives comme les perceptions tactiles ou auditives.

1. Un SDK (kit de développement) est un ensemble d'outils permettant aux développeurs de créer des applications de type défini.
2. API(Application Programming Interface) est un ensemble de classes, de méthodes et de fonctions qui servent de façade et qui offre la possibilité à un programme, de faire des actions plus évoluées.

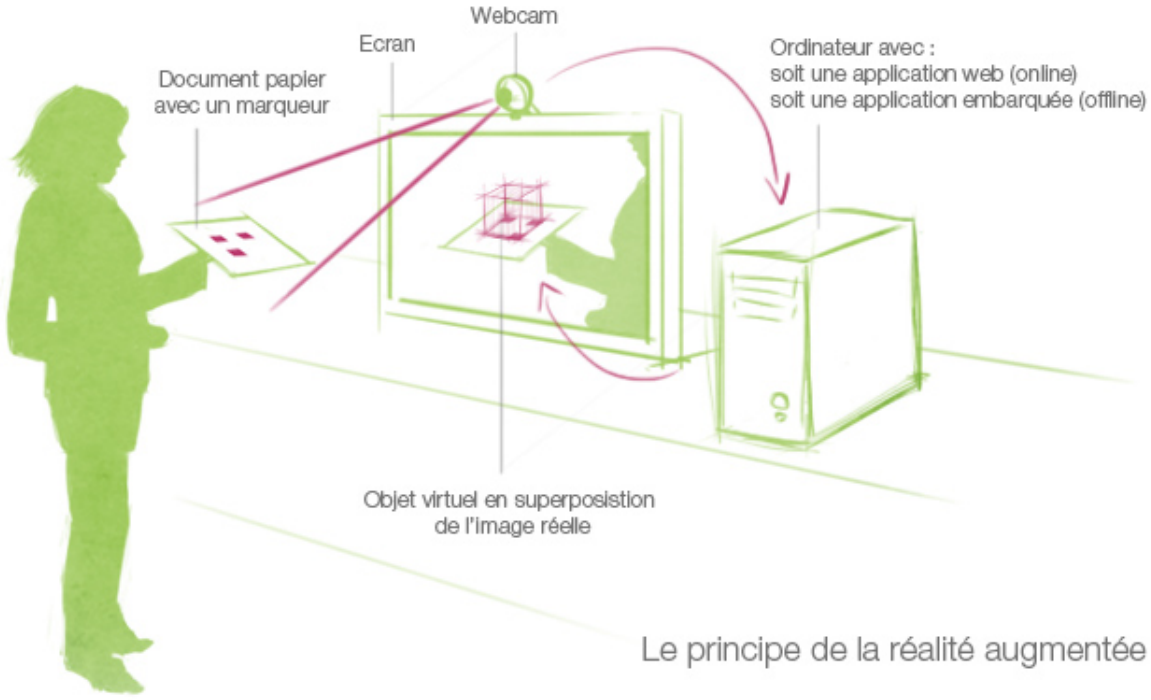


Figure 3 : Principe de la réalité augmentée

Les différents types de réalité augmentée On peut distinguer 3 types de réalité augmentée :

— Le premier utilise la caméra du smartphone. Le téléphone va analyser ce qui est filmé en temps réel. Il va ensuite identifier les éléments présents et effectuer une action en fonction de ce qui est vu. Le smartphone va ajouter des éléments à ce qui est vu au travers de la caméra de façon à ce qu'il semble s'intégrer au décor. Par exemple, le smartphone pourra reconnaître un visage, ou un bâtiment sous n'importe quel angle. Il ajoutera à l'écran un objet 2D (image, vidéo ou texte) ou 3D qui semblera être fixe dans l'environnement filmé.

Figure 4 : Application Aurasma

— Le second type de réalité augmentée utilise la caméra, la fonction GPS ainsi que l'accéléromètre[3]. Ici, la caméra ne sert qu'à afficher ce qui se trouve devant l'utilisateur. L'accéléromètre sert à gérer les mouvements de la caméra pour que l'image semble s'intégrer au décor. En d'autres termes, grâce à l'accéléromètre, si l'utilisateur pointe son appareil vers le haut, ce mouvement sera enregistré, et les éléments qui étaient affichés sortiront de l'écran. Les lieux qui se trouvent devant l'utilisateur sont reconnus en fonction de leurs coordonnées GPS, et non grâce à une analyse de l'image filmée. Il est donc possible d'afficher des objets en 3D, qui sembleront être fixes au travers de la caméra.

Par exemple, il est possible de placer un objet dans l'espace, de bouger la caméra, de faire sortir l'objet de l'écran... lorsque la caméra reviendra sur cet objet, il semblera ne pas avoir bougé.

Figure 5 : Application Métro Paris

— Le dernier type de réalité augmentée est un mélange des 2 points évoqués ci-dessus. L'image est stabilisée (détection de la rotation, et de la translation du

3. L'accéléromètre est un outil équipé sur les smartphone qui permet de connaitre l'inclinaison du téléphone. Il actualise cette information en temps réel

téléphone pour adapter le rendu) grâce à l'accéléromètre, et c'est bien une analyse de l'image capturée en temps réel grâce à la caméra, qui permet de reconnaître ce qui se trouve devant l'utilisateur.
Il est également possible d'utiliser le GPS. De cette façon le téléphone va analyser l'image filmée, cependant, s'il reconnait quelque chose, le résultat rendu peut varier en fonction de l'endroit ou se trouve l'utilisateur. Par exemple si l'utilisateur se trouve à Paris et que l'application repère un symbole, une Tour Eiffel en 3D pourrait apparaître à l'écran. Si l'utilisateur de trouve à Londre, et que l'application repère le même symbole, une reproduction en 3D de Big Ben pourrait apparaître à l'écran.

4.1.2 Recherche d'applications existantes

Afin de réaliser ce à quoi ressemblerait le module, j'ai effectué des recherches sur ce qu'il était possible de faire. Pour cela j'ai recherché des applications mobiles utilisant différents types de réalité augmentée. Voici une liste d'applications utilisant la réalité augmentée de différentes façons :

- WordLens Cette application analyse l'image et opère une modification sur les mots qui sont filmés (inversion de l'ordre des lettres pour la version gratuite, traduction pour la version complette)(Annexe A).
- Aurasma Grâce à cette application, vous pouvez faire apparaitre des objets 3D sur des supports personnalisés. (Annexe B)
- Junaio Cette application regroupe tous les types de réalité augmentée. Elle peut aussi bien vous indiquer où se trouve certains lieux grâce au GPS, que faire apparaitre des objets 3D à l'écran. Les objets 3D peuvent être affichés en fonction d'une image reconnu par l'application, ou en fonction de coordonnées GPS.(Annexe C)
- GéoCam Vous pouvez,grâce à cette application, placer des points sur une carte, et les voir apparaitre en réaliter augmentée.(Annexe D)
- Cinéma Gaumont Pathé Vous pouvez voir lieux et les heures des prochaines scéances d'un film, en scannant son affiche.(Annexe E)

4.1.3 Recherche et comparatif des SDK disponnibles

Après avoir identifié ce qui était à faire, je me suis penché sur la question comment réalisé ce module. J'ai donc cherché s'il existait déja des SDK ou des API sur lesquelles je pourrais m'appuyer pour réaliser le module. Pour cela j'ai listé différents SDK et API que j'ai pu trouver. J'ai essayé de garder celles qui fonctionnaient sous java, javascript, html.

SDK/API	GPS + Boussole	Analyse l'image	Android	iOS	Natif	Web	Fonctionne hors connexion	Affiche objets 3D	prix (en euros)
D'Fusion	x	x	x	x			x	x	0
Métaio (SDK)	x	x	x	x	x	x	x	x	5989.5, 3569.5 ou 0
Vuforia		x	x	x	x		x	x	0
Wikitude	x	x	x	x		x	x	Uniquement au format wt3	0
AR Viewer	x	?	x		x			x	0
ARToolKit		x	x	x	en C		?	x	0

Au vu de ce tableau, on peut constater que le SDK métaio est le plus complet malgré qu'il soit payant. Je me suis alors penché sur les différences existantes entre la version gratuite et les versions payants. Il ne sagit que de l'apparition d'un texte indiquant que le SDK métaio a été utilisé pour faire ce rendu. Il n'y a aucune restriction liée à ce qu'il est possible de faire ou à la qualité du rendu.
J'ai décidé d'utiliser ce SDK car il permet d'utiliser tous les types de réalité augmentée (ce qui pourrait s'avérer utile pour des applications futures). Je l'ai également choisi car il peut fonctionner en langage natif (java pour Android) mais aussi en Java Script/HTML grâce à la plate forme AREL (Annexe F).

4.2 Prise en main du SDK Metaio

J'ai d'abord dû prendre en main le SDK. Je me suis servi des exemples d'applications fournis avec le SDK pour comprendre sa structure.
Dans un premier temps, il faut charger la configuration de suivi. Pour cela on utilise la methode "metaioSDK.setTrackingConfiguration(trackingDataFile) ;" . Ici, trackingDataFile est un fichier xml contenant les informations sur l'objet à afficher. On retrouve entre autre dans ce fichier le chemin de l'image qui sera reconnu et sur laquelle l'objet 3D apparaitra, sa position par rapport à l'image, sa taille par défault, ... (un exemple est présent dans l'Annexe F) . Il est possible de mettre "GPS" en paramètre de la méthode si on souhaite utiliser le second type de réalité augmentée. Grâce à cela, l'application affichera des objets en fonction de coordonnées GPS .

4.3 Solution GPS choisie

J'ai finalement choisi d'utiliser le second type de réalité augmentée (Utilisation du GPS pour afficher les objets en fonction de leur position géographique). En effet, le premier type aurait permis de reconnaitre les lieux en analysant l'image, mais cela consommerait beaucoup de ressources, de plus l'utilisateur devrait se trouver devant le lieu pour avoir accès à ses informations. En utilisant le GPS, je peux afficher les POIs se situant autour de l'utilisateur, de façon à ce qu'il sache ce qui se trouve autour de lui (hotel, restaurant, lieu touristique, ...) pour ensuite pouvoir s'y rendre. De plus ce type de réalité augmentée consomme moins de ressources puisqu'il ne réalise pas une analyse de l'image constante.

4.4 Etapes de réalisation

4.4.1 Ajout de points personnalisés

Dans un premier temps j'ai repris l'exemple fournit avec le SDK. De ce faite je n'avais pas de problèmes liés à la tructure de mon projet : toutes les méthodes necessaires sont appelées au bon moment. J'ai donc commencé par afficher des POI [4] avec des coordonnées que j'ai choisi moi même. J'ai ainsi pu constater la précision du SDK, et la variation d'échelle des points affichés. En effet, plus un point est loin, plus il apparaitra petit sur l'écran.

4.4.2 Ajout de POI à partir d'un fichier

4.4.2.1 Lecture du fichier

Tous les POIs ne peuvent pas être chargés en dûr dans l'application, du au faite qu'il peut y en avoir un très grand nombre, et seulement ceux proches de l'utilisateur seront affichés. Vincent, mon maitre de stage m'a alors fournit un fichier texte contenant les points à afficher. L'application, une fois le module intégré, me fourrira une chaine de caractères organisés de la même façon. Cette chaine de caractères sera constituée comme ceci :

4. Un POI (point d'interêt), est un lieux où se trouve quelque chose à voir ou à faire. Ce lieux est repéré par ses coordonées GPS.

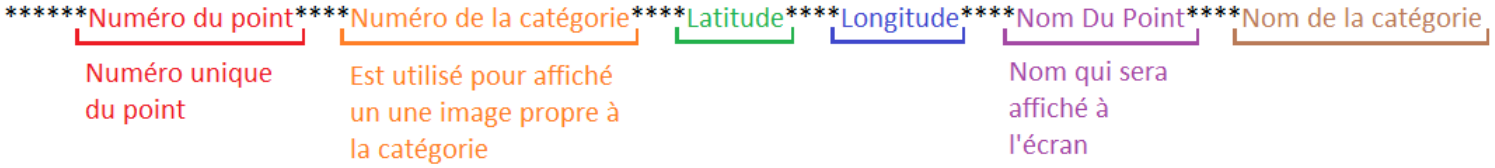


Figure 6 : Structure d'un point

4.4.2.2 Création d'une liste de points

Après quelques recherches dans la documentation de la classe String, j'ai découvert la méthode split, qui permet de découper une chaine de caractère. J'ai donc pu diviser ma chaine de caractère en sous chaines, représentant chacune un point. Pour chacune de ces sous chaines j'ai refait un split pour avoir, pour chaque point, un tableau contenant chaque attribut du point. J'ai stocké le résultat dans une liste afin de ne plus avoir à gérer le nombre de points donnés. Je me retrouve ainsi avec une liste de points, où chaque point est représenté par un tableau de chaines de caractères.

4.4.2.3 Restriction du nombre de points affichés

A partir de cette liste, j'ai donc pu afficher tous les points présents. Cependant le nombre de points étant important il a fallu trouver un critère de sélection pour ne pas tout afficher. Ma solution a été d'afficher seulement les points se situant à une certaine distance de l'utilisateur, cette distance étant stocké dans une variable.

4.4.2.4 Encodage des caractères spéciaux

J'ai ensuite eu un problème lié à l'encodage. En effet, les caractères spéciaux et accentués n'étaient pas reconnus. J'ai essayé de les changer de format mais sans résultats. Le problème venait visiblement du fichier texte que l'on m'avait donné, or, lorsque le module serait intégré, les points ne seraient pas obtenus par un fichier texte mais directement envoyés par l'application. Donc, pour la phase de développement, j'ai stocké le contenu de la chaine de caractère dans une variable String, en tant que constante.

4.4.3 Modification de l'affichage des POIs

A ce stade du projet, j'étais capable d'afficher des points avec leur nom. Cependant, il fallait que j'affiche également une image, propre à la catégorie ainsi que la distance qui

sépare l'utilisateur du lieu marqué par le POI. J'ai donc totalement changé ce qui était affiché pour localiser les POI.

4.4.3.1 Création des images

Je me suis vite rendu compte que le SDK ne permettait pas d'afficher un texte. Il permettait l'affichage d'images, d'objets 3D, et de vidéos. J'ai donc du réaliser une méthode qui, à partir du titre du point, et du numéro de sa catégorie, allait générer l'image qui allait s'afficher. Cette methode sauvegarde l'image en cache, et renvoie le chemin de l'image, de façon à afficher des POI distinct. Pour réaliser cette méthode, j'ai du créer l'image qui servirait de fond. J'ai ensuite du gérer l'affichage du nom du POI sur une ou deux lignes en fonction de sa taille, en veillant à bien effectuer le retour à la ligne au niveau d'un espace quand cela est possible. Pour l'affichage d'images liées à la catégorie du point, on m'a fourni un ensemble d'images, que l'application utilisait déja.

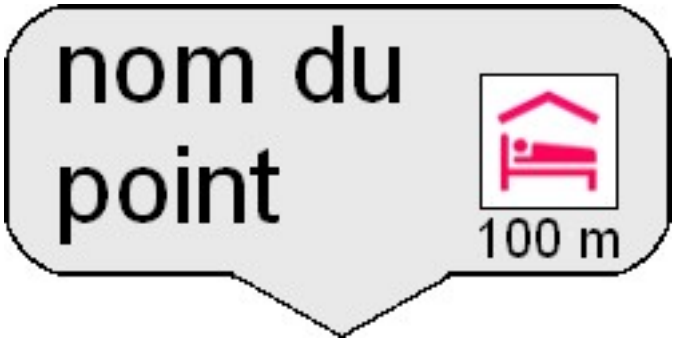


Figure 7 : Affichage d'un point

4.4.4 Affichage de la distance

Afin de rendre les informations affichées plus complètes, j'ai cherché à afficher la distance qui sépare l'utilisateur du lieux indiqué. En lisant la documentation du SDK, j'ai découvert une fonctionnalité qui permettait de mesurer la distance entre 2 points géographiques. Une fois la distance connue, j'ai dû intégrer à la méthode de génération des images, l'ajout d'une distance. Le problème qui s'est alors posé, est que j'affichais une image, et donc les distances n'étaient pas actualisées.

4.4.5 Actualisation de l'affichage

4.4.5.1 Première solution : on charge tout

L'affichage des POIs à l'écran étant géré par le SDK, j'ai pu me concentrer sur le contenu à afficher. Le premier problème auquel je me suis confronté, mais aussi le plus important, est que l'on ne peut pas créer ni modifier l'image des POIs pendant l'exécution de l'application, car seul le thread qui génère les POIs peut le faire. Il me semblait donc impossible de modifier les distances qui séparaient l'utilisateur des lieux indiqués. J'ai donc créé une première méthode pour actualiser l'affichage. Dans ce système, tous les POIs étaient affichés, mais sur ceux qui étaient suffisamment proche de l'utilisateur étaient visibles. Les inconvénients étaient que la machine devait gérer tous les points, ramenlissant ainsi le rendu. Le second inconvénient était que les distances n'apparaissaient pas à l'écran.

4.4.5.2 Seconde solution : on recharge que le nécessaire

J'ai alors fait des recherches sur les forums dédiés à ce SDK , pour savoir si d'autres personnes ont été confrontées à ce problème et comment elles l'ont résolu. J'ai finalement découvert qu'en implementant la métode MetaioSurfaceView.queueEvent, on pouvait recréer des POIs. Cette solution résolvait à elle seule tous mes problèmes. En effet, j'ai pu créer une seconde méthode pour actualiser, qui supprimait tous les points présents et qui créait les POIs proches de l'utilisateur. Ainsi, la machine n'avait plus à gérer tous les POIs, et à chaque fois que cette méthode est appelée, les distances seraient actualisée elles aussi.

4.4.5.3 Différentes méthodes d'appel envisagées pour la fonction actualiser

Je me suis ensuite posé la question de la façon dont la méthode actualiser serait appelée. Le première solution que j'ai implémentée est de créer un thread dédié à l'appel de cette méthode. J'ai d'abord essayé d'appeler la méthode à intervalle de temps régulier. Cette méthode était fonctionnelle, mais pouvait ralentir l'application sans que les POIs n'aient changés. J'ai donc implémenté une seconde solution, qui actualisait l'affichage, lorsque l'utilisateur a parcouru une certaine distance (5m). Cette méthode était également fonctionnelle mais moins précise que la solution précédente, d'autant plus que le GPS du téléphone n'est pas précis à 1 mètre près.

4.4.5.4 La méthode retenue

Je me suis ensuite rendu compte que je pouvais implémenter la méthode onDrawFrame, qui est la méthode qui dessine ce qui se trouve à l'écran. J'ai donc cherché un moyen de l'utiliser, de façon à ne pas faire gérer à la machine, un thread supplémentaire. J'ai dans un premier temps, naïvement placé ma méthode actualisée dans la méthode onDrawFrame, pour constater le résultat. Les points étaient affichés, mais on pouvait distinguer un clignotement de l'image. J'ai alors revu une troisième fois ma méthode actualisée. Mon idée cette fois était d'utiliser la vitesse d'affichage de la machine. En effet, à chaque fois que l'écran est actualisé, on passe une fois dans la méthode onDrawFrame (pour que l'écran soit fluite il faut passer au moins 24 fois par seconde). Cette méthode actualisée n'allait pas traiter tous les points affichables mais un seul à la fois. Ainsi, la diminution de la vitesse d'affichage n'était pas visible à l'oeil nu. Après quelques tests, j'ai calculé que les 150 points étaient actualisés en 2.6 secondes, sans nouveau thread, et sans ralentir l'affichage.

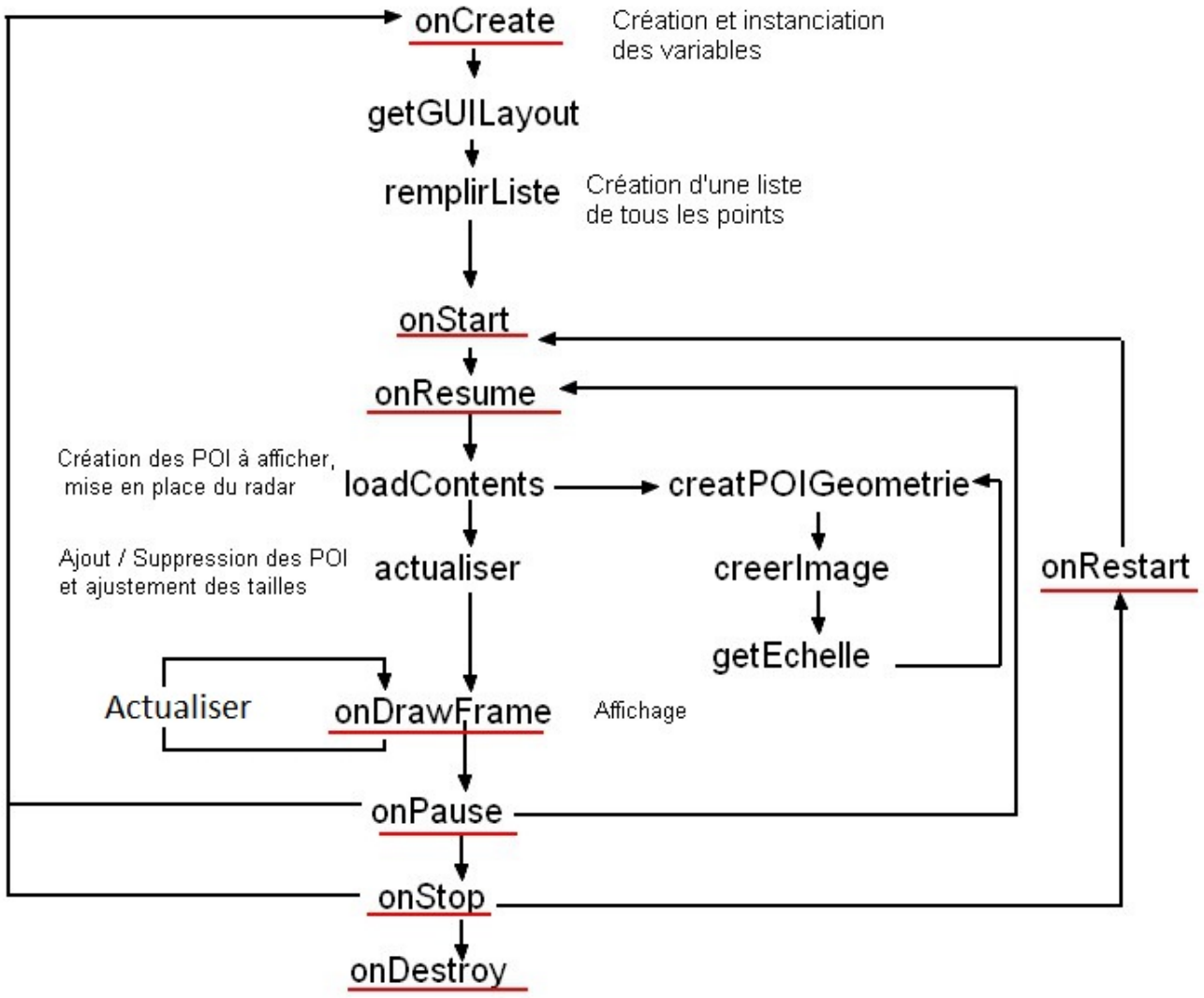


Figure 8 : Fonctionnement du module

4.4.6 Affichage du circuit

L'affichage des POIs étant fait, il ne me restait qu'une fonctionnalité complémentaire à implémenter : l'affichage d'un circuit.
Cette fonctionnalité doit pouvoir indiquer à l'utilisateur la route qu'il doit emprunter pour parcourir un circuit. Pour cela, la route à suivre doit être mise en évidence.

4.4.6.1 Le fichier contenant un circuit

Dans un premier temps, j'ai eu besoin de savoir comment était enregistré un circuit. Vincent m'a alors fournit un fichier contenant un exemple. Ce fichier était constitué d'une liste de coordonnées géographiques de points. Pour chaque point, la latitude, la

longitude et l'altitude était séparé par un virgule, et chaque point géographique étaient séparés par un point-virgule.
Afin de pouvoir visualiser le circuit, j'ai placé les points sur une carte. Je me suis alors aperçu que les points étaient disposés au niveau des intersections, et des virages.

4.4.6.2 Le résultat souhaité

J'ai donc recherché de quelle façon je pouvais indiquer le circuit à suivre. Vincent m'a alors dit que la solution idéale serait d'avoir un trait au sol, qui indiquerait à l'utilisateur le chemin à suivre.

4.4.6.3 Traçage du chemin

Etant capable d'afficher les points à partir de coordonnées GPS, j'ai recherché de quelle façon je pouvais relier 2 points affichés à l'écran. Le problème fut alors que le SDK ne possédait pas cette fonctionnalité. Je suis donc vite arrivé à la conclusion que pour intégrer l'affichage du circuit à l'écran, je devais gérer l'affichage de ce circuit en parallèle de celui géré par le SDK. D'après quelque notions que j'avais sur le dessin en java, je savais qu'il était possible de réaliser un dessin basique sur l'écran, à partir de coordonnées de l'écran. Le second problème a alors été de connaître les coordonnées de l'écran qui correspondaient aux POIs.

4.4.6.4 Recupérer les coordonnées de l'écran, des POIs

A ce moment 2 solutions s'offraient à moi : je pouvais calculer moi même les coordonnées des points. Cependant, cette solution nécessitait de connaître des caractéristiques propres à chaque appareil, tel que l'angle d'affichage de la caméra, les dimensions de l'écran, l'inclinaison verticale et horizontale du téléphone, ...La seconde solution était d'utiliser le SDK afin d'obtenir les coordonnées des points à l'écran . De cette façon je pouvais utiliser la qualité et la fiabilité du SDK. Pour cela, j'ai recherché dans la documentation de toutes les classes que propose le SDK, une méthode qui permettrait de récupérer ces coordonnées. J'ai finalement trouvé la méthode getScreenCoordinatesFrom3DPosition. Cependant, après de multiples essais, les coordonnées affichées étaient toujours 0.
J'ai donc été sur le forum dédié à ce SDK afin de connaître la cause de mon problème. J'y ai appris que cette méthode ne fonctionne correctement que sur les Objets 3D. Or ici, je n'affichais que des images. J'ai donc testé la méthode avec un objet 3D, et j'ai pu récupérer les coordonnées de l'objet. A ce moment, j'étais donc capable de récupérer les

coordonnées à l'écran, d'un point repéré géographiquement.

4.4.6.5 Gestion des points du circuit à afficher

Afin de ne pas avoir un rendu illisible, il m'a paru évident que je ne pouvais pas afficher tous les points du circuit en même temps. Il m'a fallu trouver une solution pour avoir un rendu clair et fonctionnel. Les points du circuit étant situés à chaque intersection et à chaque virage, si l'utilisateur se place sur un point, il devrait voir l'endroit où se situe le point suivant, ainsi que le point précédent.
En partant de cet hypothèse, j'ai intégré au module, la possibilité de trouver le point le plus proche de l'utilisateur, lors de l'initialisation. Je sauvegardais ensuite l'index à laquelle le point le plus proche se situait dans le tableau qui contient tous les points du circuit. J'ai ensuite intégré à ma méthode "actualiser" la possibilité de savoir si l'utilisateur s'était rapproché du point suivant, ou du point précédent du circuit, auquel cas, l'index du point le plus proche augmentait ou diminuait.
Ce schéma résume la façon dont les points sont gérés :

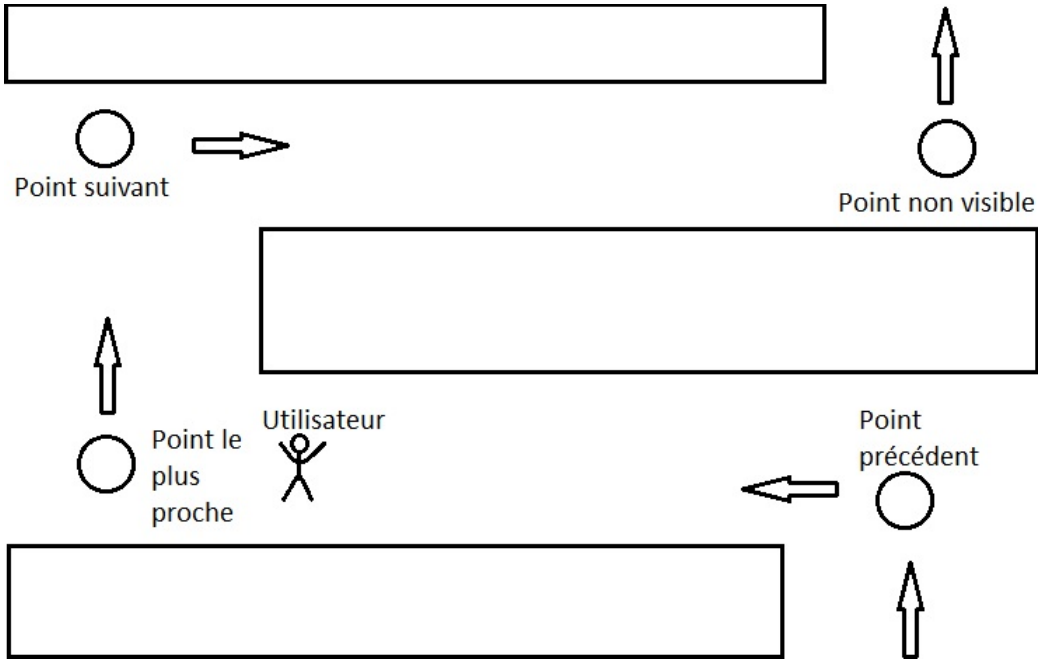


Figure 9 : Evolution des points 1

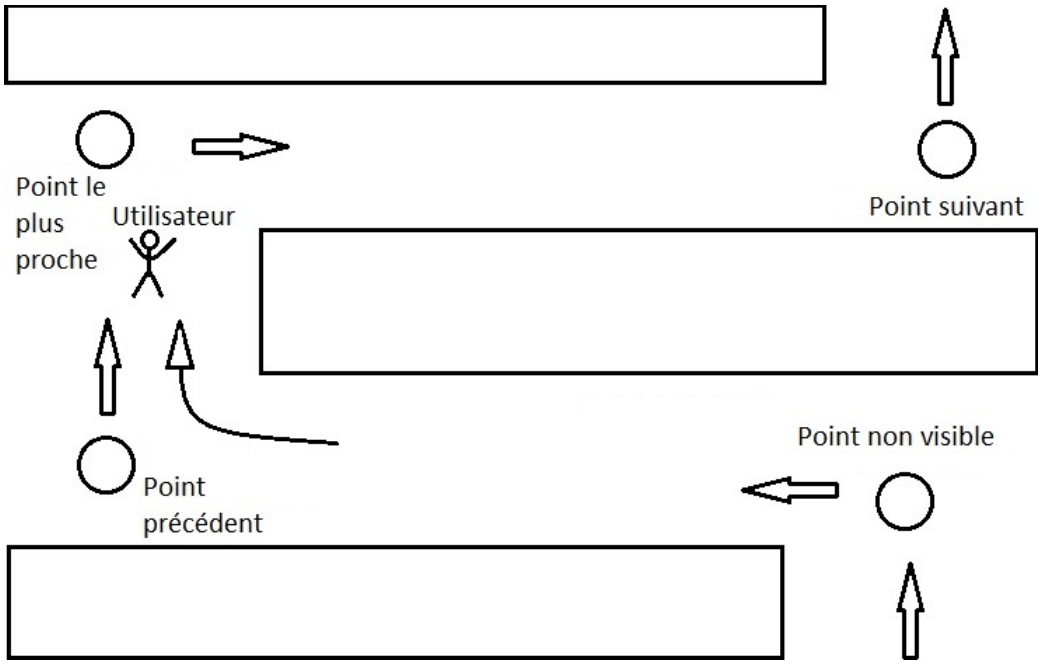


Figure 10 : Evolution des points 2

A ce stade du projet, j'avais donc mis au point un système capable de gérer quels points devaient être affichés, en fonction de la position de l'utilisateur.

4.4.6.6 Dessiner le circuit

La phase suivante était donc la phase du dessin. J'avais à ma disposition les coordonnées de l'écran du point le plus proche, du point suivant, et du point précédent. Je n'avais alors plus qu'à les relier.
N'ayant pas suffisamment de connaissance en matière de dessin en java, j'ai été me documenter sur internet afin de comprendre comment faire.
J'ai tout d'abord essayé de dessiner sur la surface que le SDK utilisait, cependant, seul le thread utilisé par le SDK pouvait réaliser cette action.J'ai ensuite eu l'idée de créer une nouvelle surface transparente que je mettrais au premier plan. Grâce à cela, je pouvais afficher la forme que je dessinais par dessus l'affichage du SDK, cependant, l'image affichée était statique. Il me fallait implémenter la méthode onDraw() de l'objet sur lequel je dessinais. Cette méthode étant "private", je ne pouvais pas l'appeller. J'ai donc créer une nouvelle classe, dans laquelle j'ai redéfini la méthode onDraw. Après affichage, l'image était toujours statique. J'ai alors recherché comment redessiner continuellement la surface, de façon à ce qu'elle s'actualise. J'ai trouvé dans la documentation, la méthode "invalidate" qui efface l'affichage et appelle la méthode onDraw. J'ai placé cette méthode dans la méthode onDraw, de façon à constamment changer ce qui est affiché.A partir

de ce moment, le module était capable d'afficher la route que l'utilisateur devait suivre pour avancer dans le circuit.

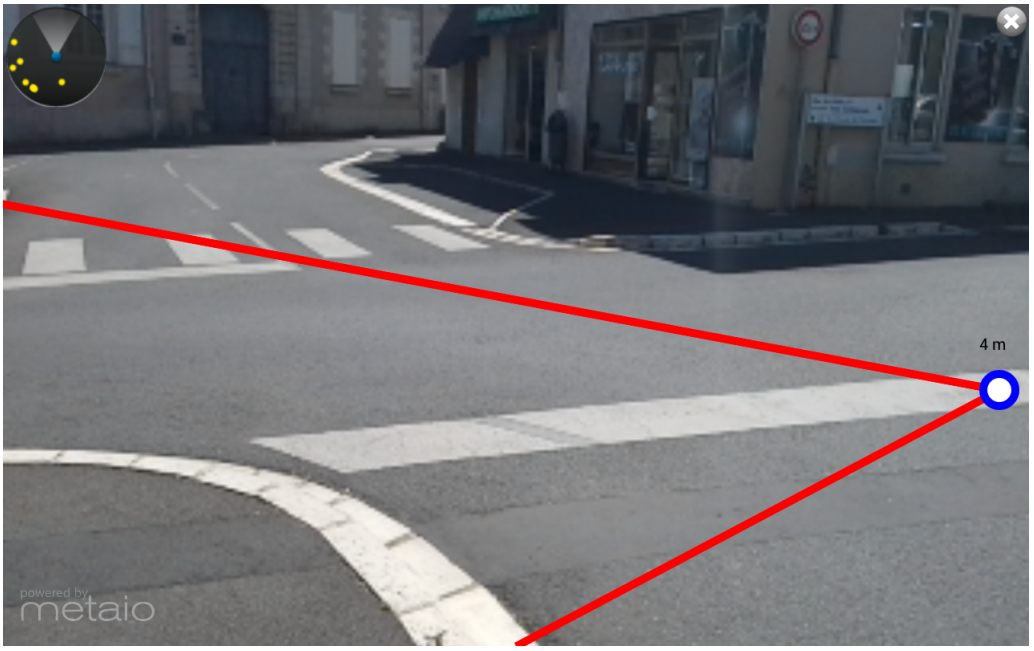


Figure 11 : rendu de l'affichage d'un circuit

4.4.6.7 Test en situation

J'ai voulu tester le bon fonctionnement de mon module en situation réelle. Je suis alors sorti dehors et je me suis laissé guider par ce que je voyais à l'écran. Certains problèmes sont alors apparus.

Le premier vient de l'affichage des points à partir du GPS. En effet, le rendu du SDK ayant une précision de quelque mètres, les points n'apparaissaient pas exactement au même endroit sur une carte et sur le module. Ce décalage a eu pour effet de placer les points sur ou derrière des murs, rendant ainsi l'affichage peu lisible.

Le second problème est lié au SDK. En effet, lorsqu'un point est sorti de l'écran , et que ses cooordonnées sont trop grandes, elles passent à -10000, -10000. Pour que le rendu soit plus agréable pour l'utilisateur, lorsqu'un point avait ces coordonnées, le trait qui le reliait à un autre point n'était pas affiché. Le problème fut alors que lorsqu'un utilisateur se trouve sur un circuit, il se dirige vers un point, mettant ainsi un autre point dans son dos. Or si un point se trouve dans son dos, le trait qui indique le circuit à suivre n'est plus affiché. Pour résumer, si un utilisateur se trouve sur un circuit, ce dernier n'est pas affiché.

4.4.6.8 Résolution des premiers problèmes

Ce schema résume la solution théorique trouvée

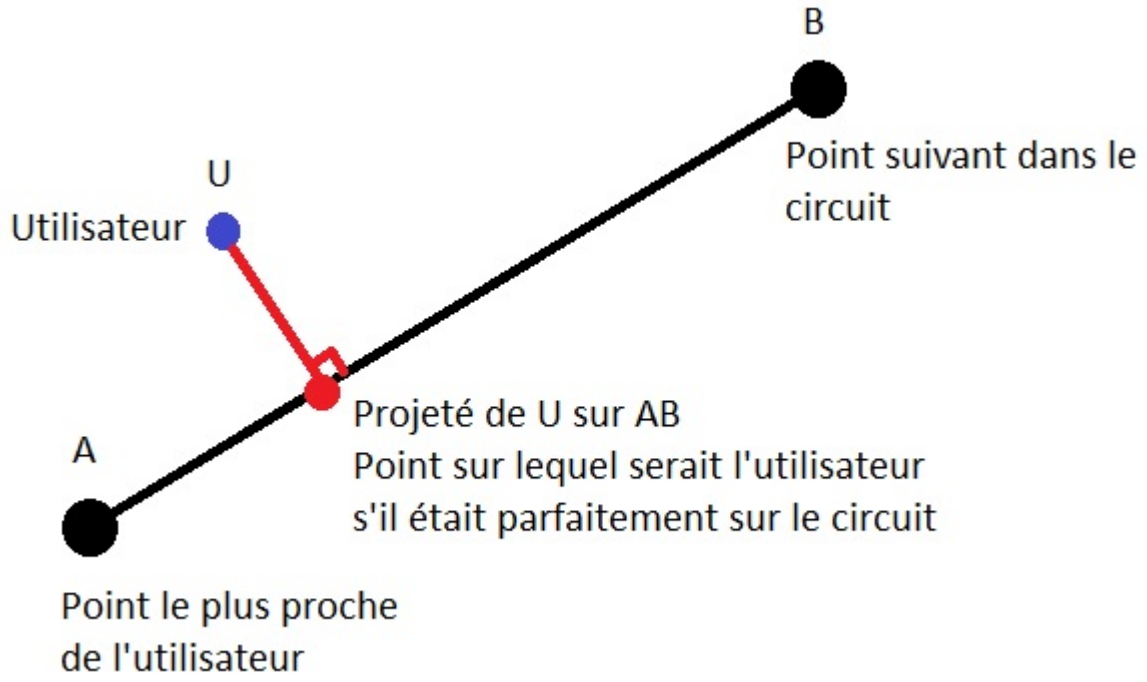


Figure 12 : Résumé de la solution

Le premier problème venait du SDK, je ne pouvais donc pas le résoudre, cependant, je pouvais limiter son impact. En effet, l'utilisateur se trouve toujours entre le point le plus proche d'un circuit, et, soit le point qui le suit, soit celui qui le précède. Or ces 3 points étaient affichés. Il y a donc 1 point qui a de fortes chances d'être derrière un mur, du fait que les points soient positionnés au niveau des virages. J'ai donc decidé de n'afficher que les points se situant devant et derrière l'utilisateur.
Pour le problème d'affichage des traits, j'ai tout d'abord eu l'idée d'afficher un trait qui part du bas de l'écran du téléphone (des pieds de l'utilisateur), jusqu'au point affiché à l'écran quand il est impossible de relier les 2 points. Après quelques tests, j'ai remarqué que cette solution était claire quand le SDK plaçait bien les points, sinon, les traits étaient entièrement sur un mur, ou coupaient un trottoir, mais ils n'étaient jamais parallèle à la route.
J'ai alors eu l'idée de placer un point supplémentaire sur le circuit. Ce point serait le point sur lequel serait l'utilisateur s'il était parfaitement sur le circuit. De cette façon, si l'utilisateur se décalle , il verra toujours une ligne à côté de lui, qui corresponderait au circuit. Mathématiquement, cela se traduit par "le projeté orthogonal d'un point (l'utilisateur) sur une droite (le circuit)". Après quelque rappels de mes cours de mathématique

de lycée, je me suis rendu compte que le calcul de ce point dans un environnement à 3 dimensions augmenterait de beaucoup le temps de calcul, j'ai donc cherché une autre solution pour obtenir ce point.
J'ai tout d'abord eu l'idée de séparer le segment à afficher en 10, et de prendre le point le plus proche de l'utilisateur.J'ai finalement utilisé une méthode utilisant un ratio. En effet, si l'utilisateur est représenté par un point U, et qu'il veut se rendre d'un point A à un point B. J'ai fait le calcul suivant :

$$ratio = \frac{distanceAU}{distanceAB}$$

J'obtenais donc une valeur entre 0 et 1. Les coordonnées du projeté de U sur AB sont donc :

$$(xa + ratio\ *(xb - xa),\ ya + ratio\ *(yb - ya))$$

Ainsi, j'ai pu placer un point sur ma droite, qui évoluait en fonction de la position de l'utilisateur. Grâce à ce point il m'était possible d'afficher, dans toutes circonstances, le circuit que l'utilisateur devra suivre.

5 Résultats

5.1 Ce que fait le module

Le module que j'ai développé pendant ces 10 semaines est capable de lire une liste de lieux ainsi qu'une liste de points (qui constituent un circuit).
Il récupère les images filmées à la caméra, et les affichent. Il va ensuite localiser les lieux (d'après la liste donnée) les plus proches. Pour chacun de ces lieux, il créé une image contenant le nom de ce qui se trouve à cet endoit, un logo symbolisant le type de lieu auquel cet endroit appartient, et enfin la distance que l'utilisateur devra parcourir pour s'y rendre. Le module intègre ces images au rendu de la caméra, de façon à ce que lorsque l'on pointe l'objectif de la caméra dans une direction, les images créées précédemment apparaissent sur le lieu qu'elles représentent.(Annexe I).

Figure 12 : Rendu final

Le module est également capable d'afficher le chemin qu'un utilisateur doit emprunter pour suivre un circuit. Et d'actualiser les points à afficher en fonction de l'avancement de l'utilisateur dans le circuit.

5.2 Avantages de cette solution

Cette solution a pour avantage d'afficher un contenu de façon clair, et lisible. En effet, il n'est pas nécessaire d'afficher les points qui représentent des lieux trop éloignés. Les informations affichées sont suffisamment grosses pour pouvoir être lues pendant que l'utilisateur est en train de marcher.

5.3 Pour aller plus loin

Il est possible de trouver quelque points à améliorer sur ce module. Par exemple, j'ai fait des recherches sur la façon de faire apparaitre des objets 3D, mais ils ne sont pas utilisés. Il aurait été interessant de pouvoir voir en 3D ce à quoi ressemblait une ruine il y a quelques siècles. Ce module n'utilise pas de reconnaissance d'image, ce qui aurait pu être utile pour avoir des informations plus complètes en arrivant sur un lieux. L'intégration de vidéos aurait également pu être une option interessante.Cette fonctionnalité est également présente dans le SDK. Une autre objection faisable est le fait que 2 images peuvent se superposer lorsqu'elles représentent 2 lieux proches l'un de l'autre.

6 Missions Annexes

En plus du module de réalité augmentée, j'ai eu à réaliser des missions annexes.

6.1 Google Annalytics

6.1.1 Objectif

L'objectif de cette mission était de réaliser une page web qui récupérerait les informations d'un compte Google Analytics, et qui les afficherait sans que l'utilisateur n'ai à fournir un identifiant et un mot de passe. L'intéret étant de proposer aux clients des statistiques liées aux applications créées par l'entreprise.

6.1.2 Recherches des API

Dans un premier temps, j'ai effectué des recherches sur les différents moyens qu'il existait pour récolter ces informations. J'ai ainsi repéré 2 API : Google Analytics API, qui permet de se connecter à google analytics et de récupérer des informations générales. Je n'ai pas réussi à obtenir des informations précises avec cet API. J'ai également utilisé gapi. Cette api prend en paramètre l'adresse mail, le mot de passe, le numéro de l'application [1], ainsi qu'une dimension et une métrics [2], et retourne un tableau contenant toutes les informations demandées.

6.1.3 Affichage des données

Afin de faire un affichage propre de ces informations, j'ai recherché une API permettant de faire des graphiques en PHP. J'ai trouvé l'API pChart, elle permet de générer tous

1. Sur un compte google Analytics, chaque application/ site enregistré est identifié par un numéro.
2. Les métrics correspondent aux informations que l'on souhaite (exemple nombre de nouveaux utilisateurs, nombre de sessions ouvertes, ...) alors que les dimenions indique en fonction de quoi les informations sont triées (exemple, la date, les région, ...)

types de graphiques de façon entièrement personnalisable, et de les enregistrer au format png.
Pour chaque graphique, j'ai créé une fonction qui récupère le nombre de nouveaux utilisateurs ou de session, en fonction des systèmes d'exploitation et des heures, des pays, des régions, ou des mois. Ces informations sont stockées dans un tableau php à double entrées, du type tab[heure][OS].
Une seconde fonction récupère ce tableau pour créer un graphique. L'utilisateur peut trier les valeurs à afficher par années et par OS (Android ou IOS). S'il demande des graphiques d'années précédentes, les images des graphiques sont stockés sur le serveur, les graphiques ne sont pas générés (car les informations ne peuvent plus changer, et pour gagner du temps de chargement). Cependant si l'utilisateur demande les informations de l'année actuelle, les graphiques seront sistématiquement recréés.

6.2 Mediaclap

6.2.1 Objectif

Mediaclap est une société qui réalise des supports multimédias pour les cours de catéchisme. Elle nous a demandé de réaliser une application qui servirait de support aux scéances de cathéchisme, et que les familles pourraient utiliser chez elles pour les approfondir.
Cette application devra pouvoir télécharger du contenu, lire des sons, des vidéos (en streaming ou en local), elle devra également proposer des petits jeux.

6.2.2 Recherches

Dans un premier temps, j'ai dû faire des recherches sur la façon de réaliser ces différentes fonctionnalitées. Je me suis aperçu que Android proposait par défaut des outils très utiles. Par exemple ma classe MediaPlayer : elle permet de lire un son. Pour cela, il faut seulement lui spécifier le son à lire et lancer la lecture. J'ai cependant dû créer une nouvelle classe de façon à ajouter des fonctionnalités tels que l'arrêt, la pause, et de boucler un son un nombre de fois donné.
Il est également possible d'afficher une page web, ce qui permet de gérer l'aspect graphique de l'application en HTML et non en Java. Cela permet aussi d'utiliser du JavaScript. J'ai fait des recherches sur l'API IUI, qui est spécialisé dans l'affichage d'une page HTML dans une application Android. Le fait de faire une application en HTML diminue le temps nécessaire pour chaque test. De plus il existe une fonctionnalité similaire sur IOS, une même page HTML peut donc servir pour la version Android et IOS d'une application.

6.2.3 Lecture de vidéos

Pour la lecture de vidéos, il existe sur android l'objet VideoView, qui fonctionne de façon similaire à MediaPlayer. ViedoView gère la lecture de vidéos en local et en streaming
J'ai mis en place une barre de chargement de façon à ce que l'utilisateur puisse savoir où il en est dans la vidéo, et qu'il puisse avancer ou reculer, comme il est possible de le faire dans tous les lecteurs de vidéos.
J'ai ensuite mis en place un bouton, qui permetterait à l'utilisateur de choisir si il veut voir la vidéo avec des bandes noirs sur les côtés, ou s'il préfère que la vidéo soit rognée, mais qu'elle occupe tout l'écran. Cette étape a été plus compliquée car elle nécessite de connaître les dimensions d'origine de la vidéo, de façon à la lire sans la déformer. Cette fonctionnalité n'étant disponible sur aucun lecteur de vidéo, j'ai dû importer la classe MediaMetadataRetriever qui lit la vidéo et génère une image au format bitMap. Grâce à cette image, j'ai pu déterminer les dimensions d'origine de la vidéo.
Il a ensuite fallu que je recherche de quelle façon il était possible de rogner la vidéo.
En effet, par défaut, android ne le permet pas. Après quelques recherches dans la documentation, j'ai découvert qu'il existait une méthode qui permet de définir les coordonnées de l'écran sur lesquelles la vidéo sera lue.
Pour H, la hauteur de l'écran
L, la largeur de l'ecran
DH, la hauteur de la vidéo par défault
DL, la largeur de la vidéo par défault
NH, la nouvelle hauteur de la vidéo
NL, la nouvelle largeur de la vidéo

$$NL = \frac{DL*H}{DH}\ NH = \frac{DH*L}{DL}$$

Les coordonnées à lécran de la vidéo (coin en haut à gauche, et en bas à droite) sont donc :
Pour voir la vidéo sans déformation, avec des bandes noirs sur les côtés :

$$(\ \frac{NL-L}{2},\ 0)(L + \frac{NL-L}{2}\ ,\ H)$$

Pour voir la vidéo sans déformation, en plein écran, mais en rognant l'image :

$$(\ 0,\ \frac{H-NH}{2})(L,\ NH + \frac{H-NH}{2})$$

6.2.4 Télécharger un fichier

Pour le téléchargement, Vincent m'a fournit une classe qu'il avait déja mis au point , qui permet de télécharger des données à partir d'un serveur ftp, et de les dézipper si besoin. J'ai un peu modifié cette classe de façon à ce qu'elle s'adapte au projet, et également qu'elle gère les dossiers présents dans l'archive (à l'origine, si l'archive contenait un dossier, il était enregistré comme un fichier, une exception était lancée lors de l'écriture des fichiers contenus dans ce dossier). J'ai ajouté une barre de chargement qui indique à l'utilisateur où en est le téléchargement. L'un des problèmes que j'ai eu est pour actualiser cette barre. Dans un premier temps, je la faisais s'actualiser à chaque fois que l'on recevait des données. Cependant, avec cette méthode, le téléchargement était plus long, j'ai donc lancé un thread qui actualise la barre de chargement une fois toutes les secondes. Ainsi, pour un fichier de 12Mo, le temps de téléchargement passe de 5 minutes, à 15 secondes.

6.2.5 Mini-jeu "Des images et des sons"

L'application devra contenir des jeux. L'un d'entre eux consiste à faire écouter un son, l'utilisateur doit séléctionner l'image qui correspond. S'il a séléctionné la bonne image, le son suivant est joué, ...
J'ai dû réaliser cette application en HTML/Javascript. La seule chose que devra faire l'application java, sera de jouer les sons, tout le reste est fait en javaScript.
Dans un premier temps il a fallu faire en sorte que les images affichées occupent tout l'espace de l'écran. Il fallait qu'elles soient rognées si l'écran est trop large, ou trop haut. Il a ensuite fallu écrire le code javascript qui identifiait si l'image cliquée était la bonne, et qui envoyait le son à jouer à java.

6.2.6 Mini-jeu "Les 4 différences"

Ce jeu affiche 2 images, contenant 4 différences. Lorsque l'utilisateur touche l'écran, au niveau d'une différence, une image apparait (un petit 'v' vert), pour signaler que cette différence est trouvée. Ce jeu devait lui aussi être créé en HTML/Javascript.
Les 2 images contenant les différences devaient occuper tout l'écran, elles étaient donc rognées de la même façon que pour le jeu précédent. Le problème qui s'est posé a été le placement des zones sur lesquelles l'utilisateur pouvait appuyer pour qu'une différence soit validée. En effet, les dimensions des images, ainsi que le rognage, dépendent des dimensions de l'écran. J'ai donc écrit un algorithme qui calcule la taille des images sans

le rognage qui est fait sur chaque bord, et ainsi, j'ai pu déterminer à quelle distance du bord se trouve chaque zone de détection.

6.2.7 Les transitions

L'interêt de faire ces minis-jeux en HTML/Javascript est de limiter le nombre de chose que le code natif aura à faire (Java pour Android) , et ainsi limiter la quantité de travail à fournir pour passer une application pour Android, à une application pour IOS.
Il a donc fallu gérer l'enchaînement des jeux et des vidéos en javascript. Pour cela, j'ai créé un fichier "transition.js", qui gère cet aspect de l'application. Au début de l'application, un scénario est donné à javascript sous la forme d'une chaîne de caractères. Cette chaîne de caractères est constituée d'un mot représentant le premier évènement (un jeu ou une vidéo), suivi d'une virgule, puis d'un paramètre si besoin. A cela on y ajoute un point-virgule, puis un mot représentant le second évènement, avec son paramètre, ...Voici un exemple de scénario : "vidéo, introduction.mp4 ;jeu1 ;jeu2 ;vidéo, conclusion-Lecon1.mp4".
Lorsque javascript obtient ce scénario, il l'enregistre dans une variable de session, de façon à ne pas perdre cette information quand la page suivante sera chargées. Quand un jeu est fini, le premier élément du scénario est retiré de la chaîne de caractères, celle-ci est alors enregistrée dans la variable de session, et le jeu suivant est lancé.

7 Conclusion

J'ai pu mettre au point un module permettant de repérer des lieux à partir de leur coordonnées GPS. J'ai appris à créer des images "à la volée" en Java. Cette experience à également été l'occasion de me familiariser avec les programmations de système embarqués, ainsi qu'aux contraintes que cela impliquent.
A travers ce rapport, j'ai essayé de vous expliquer les différentes étapes que j'ai traversées afin de mettre à bien ma mission. Je n'ai pas pu vous expliquer la totalité des problèmes que j'ai rencontrés car j'ai préféré me concentrer sur les problèmes majeurs, qui avaient un impacte sur le rendu final.
Ce stage m'a permis de mieux connaitre l'architecture d'une application Android, notamment au niveau de la gestion des données stockées dans l'application. En effet cette gestion est différente que celle utilisé sur ordinateur.
J'ai également appris à effectuer des recherches de façon précises, ce qui m'a appris à m'adapter à des SDK inconnues de façon rapide afin de pouvoir l'utiliser dans un autre contexte.

Il est encore possible d'améliorer ma solution, avec par exemple la possibilité d'accéder au site des lieux indiqués en cliquant sur le POI affiché, ou encore en modifiant la portée des éléments à afficher en fonction du lieu où on se trouve(si l'utilisateur est au milieu de la campagne, il faudrait afficher des éléments plus éloignés que si il est en ville).

Cette expérience n'aura pas seulement été une source d'apprentissage de connaissance technique. Ca a également été une experience humaine, grâce aux personnes que je côtayais tous les jours, et qui ont fait de ce stage, une formidable éxperience.

Bibliographie

ERIC SARRION, *Développement Web pour mobiles iUI : installation et première application*, EYROLLES, 2010

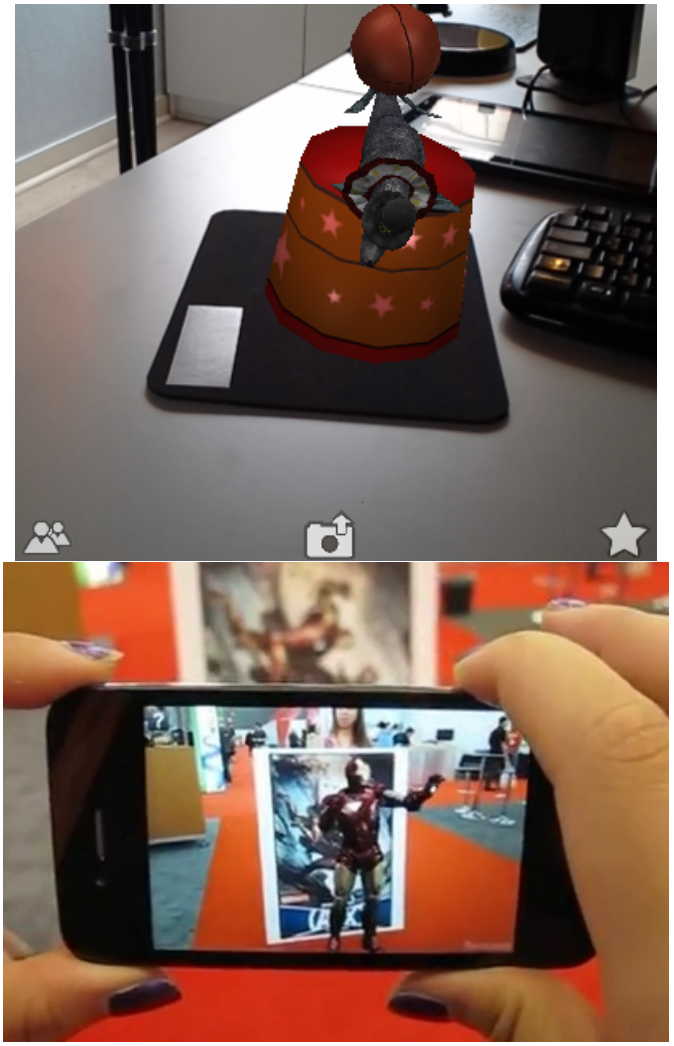
http ://helpdesk.metaio.com/
http ://dev.metaio.com/sdk/tutorials/location-based-ar/

A Application Word Lens

Application Word Lens

Cette application analyse les mots qui s'affichent à l'écran, et place les lettres à l'envers, tout en conservant la couleur de fond, ainsi que la police d'écriture. Cette application est également capable de traduire les mots qu'elle analyse, et d'afficher leur traduction. De cette façon, l'utilisateur aura l'impression que tous les mots affichés par la caméra sont dans sa langue.

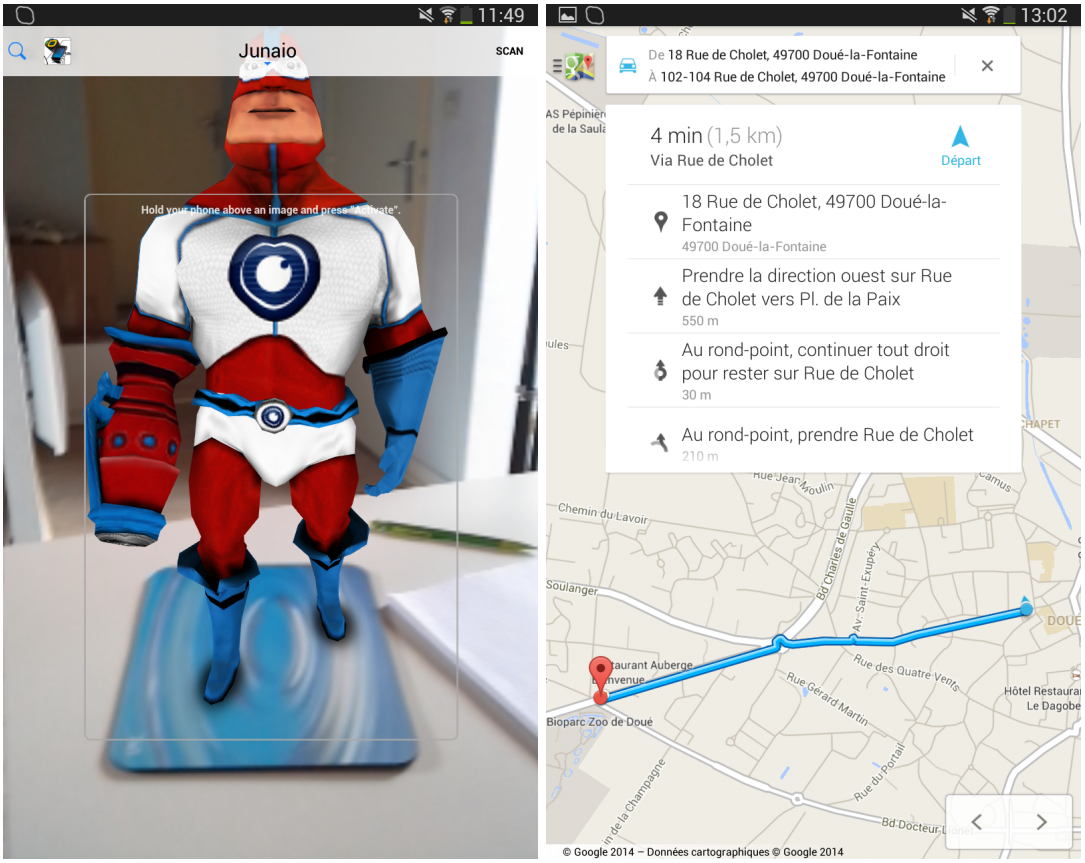
B Aurasma



Application Aurasma

Aurasma est une application qui reconnaît les éléments filmés, et affiche un objets 2D ou 3D en fonction de ce qui est en face de l'utilisateur. Il est possible d'ajouter des cibles, qui feront apparaître des objet 3D. Cette application est également capable de reconnaître une affiche de cinéma, et renvoie l'utilisateur vers la page wikipedia de ce film.

C Junaio

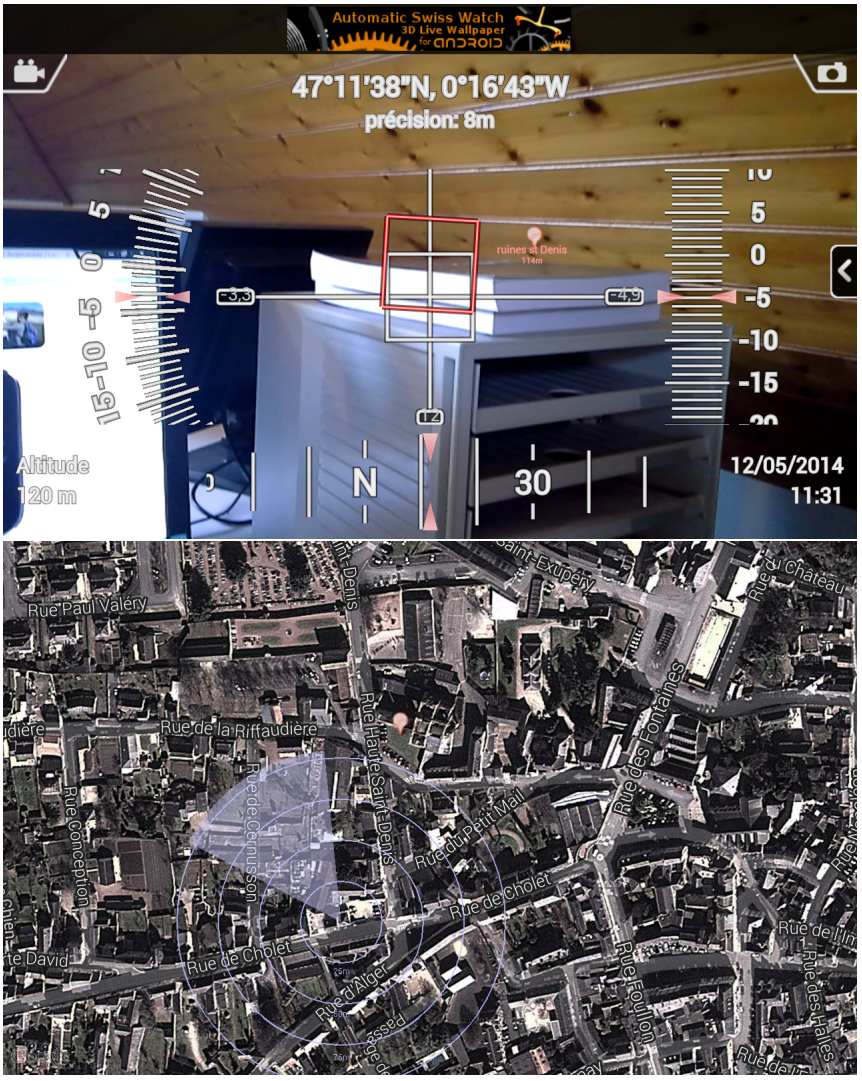


Application Junaio

Junaio est une application qui permet d'utiliser tous les types de réalité augmentée.
Il est possible de définir un "marqueur" grâce à une simple photo. Lorsque l'application reconnaitra ce marqueur, elle affichera un objet en 3D.
Cette application peut également utiliser la fonction GPS, et nous indiquer des points d'intérêts dans un rayon allant de 0 à 50km (modifiable pas l'utilisateur). Ces points d'intérêts sont affichés en utilisant la caméra ou sur une carte. L'utilisateur peut alor sélectionner l'un de ces points, Junaio peut ensuite indiquer sur une carte un itinéraire à suivre pour s'y rendre, la distance à parcourir et le temps, en fonction du moyen de

locomotion à la disposition de l'utilisateur
Cette application utilise le SDK Metaio

D GéoCam Free



Application GéoCamm Free

Cette application est la version gratuite de GéoCam. Elle permet de donner des informations relatives à la position de l'utilisateur , et à l'orientation du smartphone. Elle permet également de localiser l'utilisateur sur une carte, de créer des marqueurs personnalisés, qui seront affichés en réalité augmentée.

E Les Cinémas Gaumont Pathé

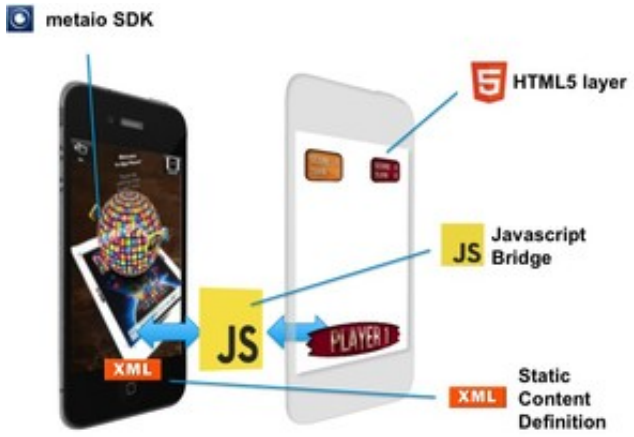
Il s'agit d'une application pour s'informer sur les films, horaires et offres des cinémas, gérer son compte carte de fidélité et acheter ses places online. L'application propose son service en réalité augmentée.

Elle renseigne sur la prochaine séance dans le cinéma le plus proche, ou dans une liste de cinémas favoris .

F AREL

Avec le SDK de metaio il est possible de créer une plate-forme indépendant de Réalité Augmentée avec AREL, au lieu d'utiliser la plate-forme des langages de programmation spécifiques (Java pour le SDK Android, Objective C pour iOS et C + + pour Windows).

Comment foncitonne AREL



Application Aurasma

AREL est constitué des composants suivants :

Un fichier XML La partie XML définit les objets virtuelles dont le contenu doit être chargé, comme les modèles 3D ou les panneaux d'affichage. Il définit également les propriétés initiales de ces objets comme la taille, le système, les transformations, les coordonnées, ...

Un fichier HTML 5 Il fournit l'interface graphique de l'utilisateur et interagit avec le SDK de metaio, en utilisant le "pont JavaScript". Il est possible de le concevoir comme un site Web mobile, à l'exception que le fond est transparent pour laisser l'utilisateur voir l'image de la caméra et les éléments de réalité augmentée ajoutés par dessus.

Le fichier javascript Ce fichier fait le lien entre les fichiers html et xml.

Voici un exemple de code XML

```
<?xml version="1.0" encoding="UTF-8"?>
<results>
  <arel>simpleAREL.html</arel> <!-- HTML GUI that should be loaded -->
  <object id=\Tiger\>
    <assets3d>
     <model>Assets/tiger.md2</model>
     <texture>Assets/tiger.png</texture>
     <transform>
      <translation>
        <x>10.0</x><y>0.0</y><z>0.0</z>
      </translation>
      <rotation type="eulerdeg">
        <x>90.0</x><y>0.0</y><z>0.0</z>
      </rotation>
     </transform>
     <properties>
      <coordinatesystemid>1</coordinatesystemid>
     </properties>
    </assets3d>
  </object>
</results>
```

Le code HTML :

```
<html>
<head>
   <!-- Integrates the arel javascript bridge -->
   <script type="text/javascript" src="http://dev.junaio.com/arel/js/arel.js"></scrip
   <!-- Includes application logic -->
   <script  type="text/javascript" src="logic.js"></script>
</head>

<body>
</body>
</html>
```

Et le fichier javascript :

```
function trackingHandler(type, param)
{
 // called when a pattern is tracked / lost
}

arel.sceneReady(function()
{
 // scale up loaded 3D model by factor 2
 arel.Scene.getObject("Tiger").setScale(new arel.Vector3D(2, 2, 2));

 // set a listener for tracking events
 arel.Events.setListener(arel.Scene, trackingHandler);
});
```

G Cahier des charges

Cahier des charges fonctionnel		
Structure émettrice	Mobile développement	
Emetteurs		email
Destinataires		

- **1. Présentation générale du problème**
    - **1.1 Projet**

L'objectif de ce projet est d'être capable de développer un module pour un application Android utilisant un système de réalité augmentée. L'objectif étant de pouvoir proposer aux clients la possibilité d'intégrer ce module à leur application.

    - **1.2 Contexte**

La réalité augmentée est une demande faite par de plus en plus de clients (entre autre pour les applications Touristic' Tour). Une fois réalisé, ce module pourra, si besoin, être intégré à n'importe quel autres projets de l'entreprise.

Aucune étude préalable n'a été faite. Il faudra lister différentes applications existantes sur Android ou Iphone, utilisant la réalité augmentée, afin de bien comprendre ce qu'il est possible de faire. Il faudra ensuite rechercher comment réaliser cette application. Dans un premier temps il faudra chercher les différents SKD et les différentes API existantes. Une étude de faisabilité sera à effectuer afin de déterminer quelle solution serait envisageable d'un point de vue technique (puissance des machines nécessaire, complexité du SDK, …) et financier (coût du SDK).

Le besoin d'une tel application est née en réponse à de demandes faites par des clients, notamment dans des salons.

    - **1.3 Énoncé du besoin (finalités du produit pour le futur utilisateur tel que prévu par le demandeur)**

Cet fonctionnalité de l'application devra permettre à ses utilisateurs d'avoir des informations, et de suivre un circuit déjà enregistré dans l'application. Elle utilisera la caméra du smartphone et de la réalité augmentée.

- **1.4 Environnement du produit recherché**

Ce module devra fonctionner sur des applications Android, et sur tous ses supports matériels. La version minimale d'Android n'es pas définie. Le module devra fonctionner sur les différents supports. Android étant un système embarqué, il y aura des contraintes, notamment au niveau des performances du matériel, ainsi qu'à la quantité de mémoire vive disponible .

- **2. Expression fonctionnelle du besoin**

- **2.1 Fonctions de service et de contrainte**

La fonction principale de ce module est d'afficher le circuit sélectionné par l'utilisateur grâce à un système de réalité augmentée. Pour cela, il faudra intégrer des éléments 2D (Images, vidéos, texte, …) ou 3D au rendu de la caméra. Le rendu sera effectue sans intervention de l'utilisateur, uniquement avec des informations de l'environnement (position et image de la caméra).

De plus, ce module devra être capable d'afficher des « points d'intérêt » proches de l'utilisateur. Un point d'intérêt étant un lieu (repéré grâce à sa position géographique), et auquel on peut associer un nom, une image, …

Cependant, le fait de pouvoir afficher beaucoup d'informations peut constituer un problème. C'est pour cela qu'il faudra aussi veiller à se que le contenu affiché reste lisible.

- **2.2 Critères d'appréciation (en soulignant ceux qui sont déterminants pour l'évaluation des réponses)**

Le module devra permettre de repérer un lieu ou une image et d'afficher les éléments voulus.

- **2.3 Niveaux des critères d'appréciation et ce qui les caractérise**

Le module devra absolument repérer une cible dès qu'elle se présente devant lui. Les éléments ajoutés devront apparaître dans une échelle adaptée.

Si possible, les éléments ajoutés (notamment ceux en 3D), devront tourner en même temps que l'utilisateur pour donner un effet de réalisme. Et les échelles de ces derniers devront s'adapter aux mouvements de l'utilisateur. Il sera également possible de faire une version du projet fonctionnant sur une plat forme web.

- **3. Cadre de réponse**
    - **3.1 Pour chaque fonction**
        - **3.1.1 Solution proposée**

Afin d'afficher le circuit que l'utilisateur devra suivre (fonction principale), il faudra tout d'abord rechercher les SDK existant. Pour chacun d'eux, il faudra lister ses avantages et ses inconvénients afin de faire un premier trie. Une fois ce trie effectué, il faudra tester chacun des SKD restant pour sélectionner lequel sera utilisé pour le développement.

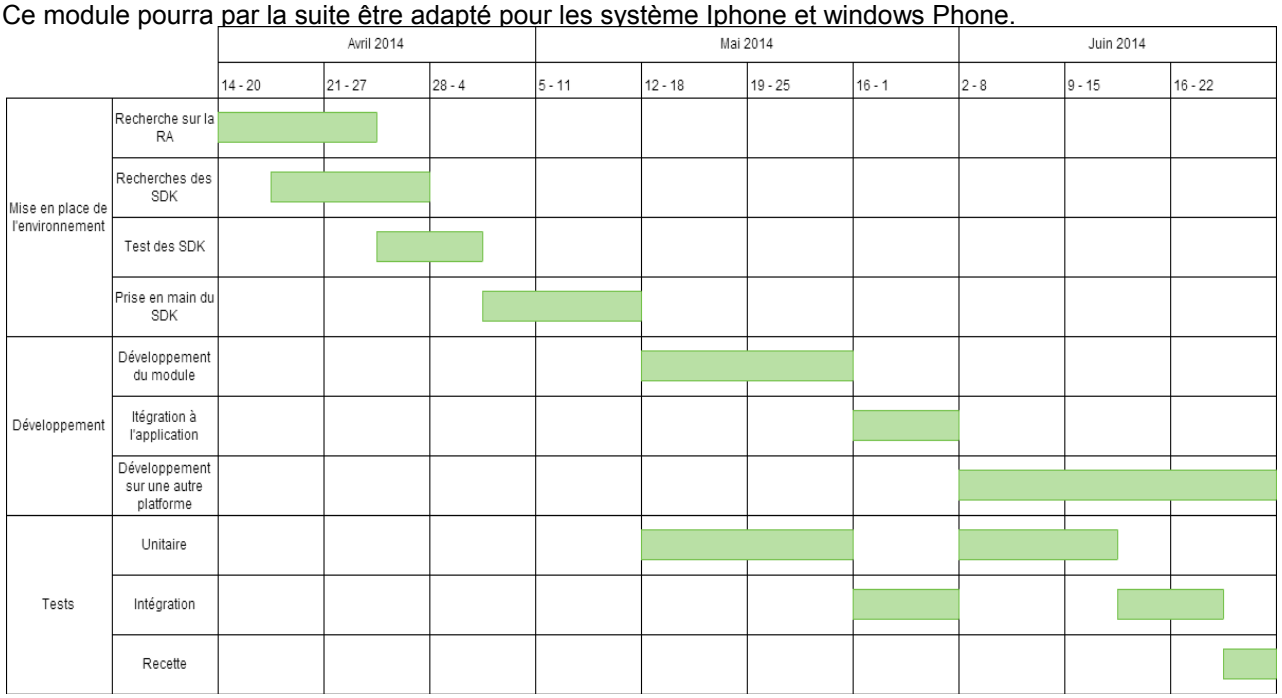
Ensuite, il faudra se documenter plus en détail sur les différentes fonctionnalités de ce SDK pour pouvoir le prendre en main. Une fois cette phase terminée, il va falloir réaliser le module pour France Touristic. Si besoin, il faudra créer une interface facilitant l'utilisation du SKD pour des utilisations futures.

- **3.2 Pour l'ensemble du produit**

Le coût de la réalisation de ce module sera fait en fonction de celui du SDK sélectionné. En effet, les pris des SKD vont de 0 à 6000€ . Il faudra essayer de limiter les coûts liés à son achat (pour cela il faudra bien prendre en considération le prix par rapport aux avantages apportés lors de la phase de sélection du SDK).

Le module ne sera pas décomposé en sous ensemble étant donné qu'il se base sur un SDK qui devrait permettre de réaliser des fonctionnalités variées.

Ce module pourra par la suite être adapté pour les système Iphone et windows Phone.



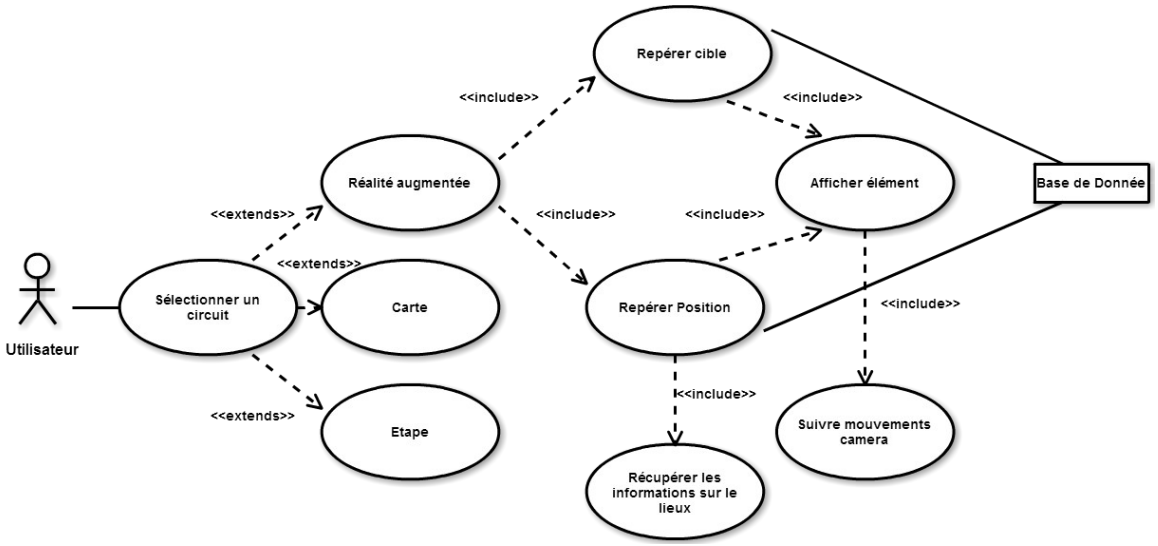


Diagramme de Grantt

(Ici, c'est un exemple du fonctionnement du module dans l'application France Touristic. L'application permet également de visionner les étapes à partir d'une carte ou de la liste des étapes. Ces parties ne faisant pas partie du projet, elles ne sont pas développées)

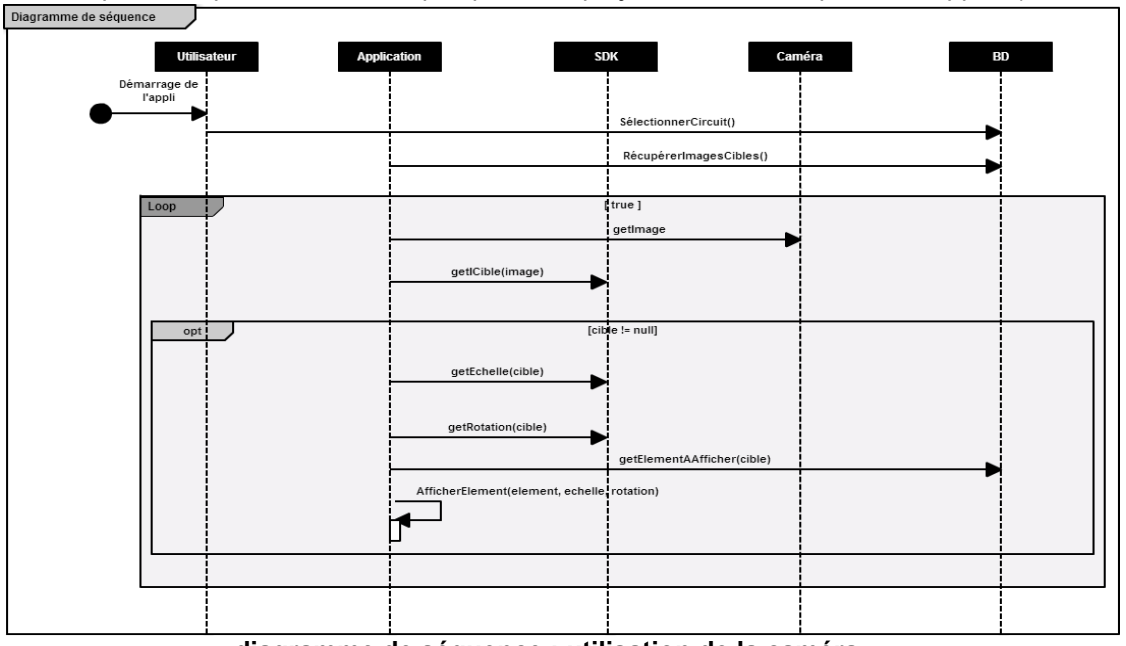


diagramme de séquence : utilisation de la caméra

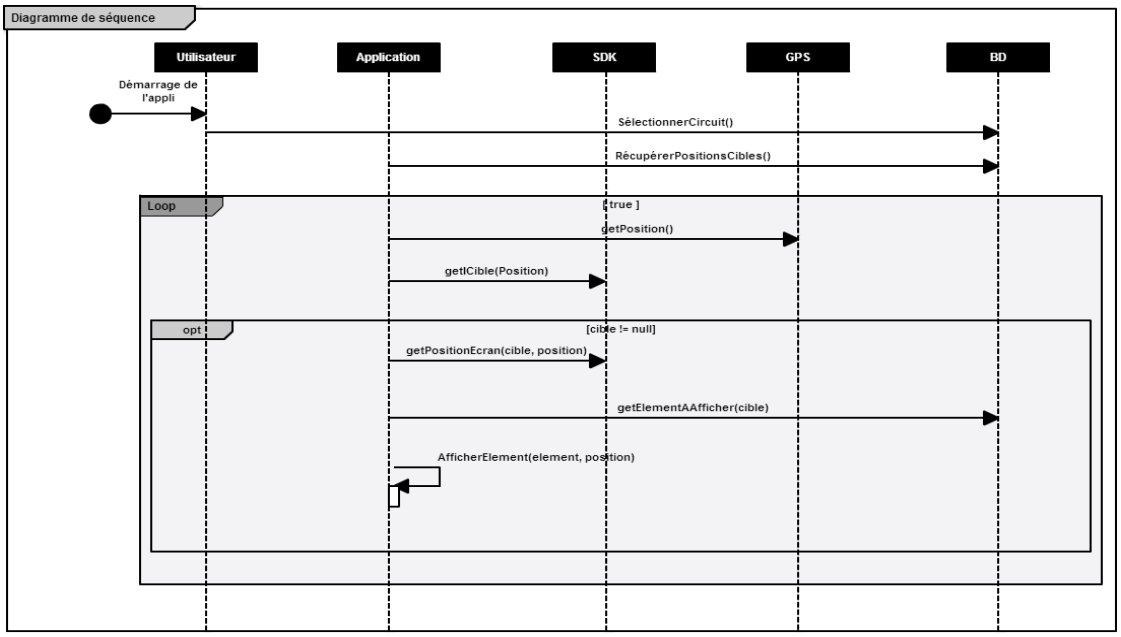


diagramme de séquence : utilisation du GPS

H Journal

Journal de Stage

14/4/2014
Réunion pour prendre connaissance de la mission et de l'entreprise Recherche sur la réalité augmentée + exemple Documentation sur l'entreprise Recherche sur comment récupérer les infos de google analytics

15/4/2014
Recherche sur les différents SDK existants. Trie des SDK Installaion d'eclipse tentative d'installation du SDK Android Remplissage des fiches des sites de tourisme

16/4/2014
Recherchedes API existantes ...

Début d'un tuto pour utiliser Google API

17/4/2014
Recherches sur les API sélectionnées Installation de eclipse et le SDK Android Avancé dans le tuto Google API Installation de FileZilla (Serveur ftp)

18/04/2014
Recherche sur comment utiliser wikitude (API RA) sans installer l'application Recherche sur le SDK ma... (téléchargement d'exemple) Installation du SDK Android

22/04/2014
Synthèses des informations recueillies durant la semaine précédente(definition, applications existantes, SDK et API) Remplissage de fiches

23/04/2014
Recherches de avantages et inconvéniants des SDK Reprise à zéro du tuto Google Analytics pour le faire correctement Réussite de connexion à google, décolte de données générales

24/04/2014
Test des SDK Metaio fonctionne Vuforia et Wikitude bug Essais d'échanges de donnée avec Google Analytics Remplissage de fiches

25/04/2014
Test des SDK Débuggage du SDK vuforia ARViewer bug Suite tuto Google Analytics Réunion Rédaction du cahier des charges pour l'IUT

28/04/2014
Diagrammes pour le cahier des charges Test de compatibilité avec les différents formats de 3D, pour le SDK métaio Mise au point de certains diagrammes Avancé dans Google Analytics

29/04/2014
Recherche sur comment utiliser le SDK metaio Probleme avec la structure des exemple et les protocoles internes Avancée dans Google Analytics, j'affiche beaucoup d'information pour avoir une vue large de ce que je peux faire.

30/04/2014
J'ai compris la structure et le protocole interne du SDK, j'ai donc pu créer un projet avec une seul classe qui peut utiliser la réalité augmentée J'ai avancé dans googl Analytics, j'ai trouvé certaines informations, et j'ai une vue d'ensemble sur la structure de l'API google.

2/05/2014
J'ai rechercher comment utiliser les POI, j'ai chercher à gérer la taille , la position... Avancée dans google analytics, je peux afficher des infos dans toutes les catégorie, je ne peux pas encore afficher le contenu des tableaux

5/05/2014
J'ai cherché comment changer l'affichage du texte J'ai commencé à refléchir à comment repérer des cibles qui se trouve dans une autre cibre. J'ai donc testé en géolocalisant Il faut que je réfléchisse comment faire en analysant Test d'un autre API pour google analytics, plu simple d'utilisation, mais je ne peux pas afficher le nombre de personnes connectés en ce moment

6/05/2014
J'ai continuer à chercher comment afficher le texte à la place des pointeurs des POI, mais le texte est une image créée à la volée. J'ai mis en place un système pour afficher une liste de POI a partir de'une chaine de caractères, J'ai fait une méthode actualiser mais le radar bug, et je ne sais pas comment l'ajouter à la boucle principale Tentative de fusion des 2 API pour pouvoir avoir accès au nombre d'utilisateur actuellement connectés

7/05/2014
J'ai identifié le problème de la méthode actualisé sans pour autant l'avoir résolu. J'ai mis le texte de chaque POI à la place des marqueurs. J'ai découvert comment changer l'image, et commencé à créer une autre image. J'ai cherché, découvert et pris en main une API pour faire des graphiques en php.J'ai également commencé le rendu du site final

19/5
Recherches API pour lire une vidéo (j'ai trouvé une API qui permet de lire une vidéo en ligne mais pas en locale) Utilisation de l'objet videoView présent par défault . Ajustement des détails de la page de stats (de Google Analytics) Recherches pour temps réel.

20/5
Affichage des vidéos au format mp4 et webm Affichage plein écran et dans une autre activity Création d'une classe Media pour gérer les sons Lancement d'un son Création des bouton de teste des fonctionnalités

Recherches sur comment afficher les utilisateurs (avec Iframe) sans appuyer sur connexion)

21/5
Changement de l'interface graphique médiaclap pour utiliser une webView Mise en plein écran des vidéos Recherches sur la façon de rogner les bords de la vidéo pour ne pas la déformer Recherche sur le fonctionnement du SDK metaio JS : 1 fichier avec des données, mais aucun changement pour la modification des images. (Pb : le fichier xml de configuration n'est plus trouvé)

22/5
Modification de l'ancrage/Plein écran, grâce à un bouton Début du téléchargement des fichier en ftp Identification des problèmes (les points ne s'affichent pas/ js n'est pas toujours chargé) Recherche sur la façon de changer des fonds des picots.

23/5
Téléchargement de fichier par ftp (Pb : La méthode de dézip ne voyait pas le fichier créé par la méthode de téléchargement Affiche d'une barre de progression pour le télécharger : faire s' actualiser la barre pendant le téléchargement (Pb : AsynchroneException)

26/5
Affichage de la barre de chargement des téléchargement Modification de la classe Media de façon à pouvoir boucler (Pb : Il n'y a pas de Listener, on ne peut pas faire une boucle sans bloquer le système) Changement de l'interface pour y intégrer le bouton chrono Rédaction de la javadoc Dessin d'un schéma de l'enchaînement des méthodes Ajout de commentaires dans le code pour toutes les méthodes importantes Ajout de la méthode

Actualiser dans la boucle onDrawFrame (Pb : l'image saccade) J'ai donc créé un 3ème méthode actualiser qui cherche 1 élément de la liste de points et qui le traite . Ainsi à chaque tour de boucle de ma méthode onDrawFrame, un unique point est traité (temps de traitement des 150 points est de 2.6s)

27/5

Mise en place de la boucle pour le chrono Modification des boutons Affichage d'une barre de progression pour la vidéo (Pb : le thread qui la gère le lance avant que la durée de la vidéo ne soit connue) Adaptions des dimensions de la vidéo (avec marges/rognée) en fonction de l'écran, et non de façon static RA : Recherche sur la façon d'afficher un trajet (possibilité de récupérer les coordonnées des POIs sur l'écran) Recherche sur la façon de trouver la position des points à l'écran en fonction de leur coordonnées GPS (calcule des angles, des longueurs, . . .)

2/6

Affichage d'une vidéo stocké en local : La vidéo est trouée grace à son identifian R.id.video Adaptation de la taille des boutons en fonctions de la taille de l'écran Optimisation du temps de téléchargement : la vidéo mettait 5 minutes à se télécharger à cause de l'actualisation de la barre de chargement à chaque octet téléchargé L'actualisation se fait maintenant toutes les secondes Création d'un apk médiaclap Mise en ligne de l'apk Suppression de l'affichage du nombre d'octets téléchargés Lecture du livre "XHTML/CSS et JS pour web mobile" Augmentation de la largeur et la hauteur des boutons en pourcent

3/6

lecture et début de la prise en main des fonctionnalités de IUI (Pb : difficultés d'utilisation des fonctions qui détectent si l'écran a été touché) RA Affichage des 2 POIs : l'un pour le points qui suit le point le plus proche(dans le circuit), et un autre pour celui qui le précède. Test en ville de l'application Récupération des coordonnées de l'écran d'un POI

4/6

Prise en main du multi-touch avec IUI (déplacement , rotation et zoom) (Pb : rotation et zoom ne fonctionnent pas) Début de la conception d'un mini jeu : création de la page HTML et du fichier java (Pb au nouveau du lien entre java et javascript, java ne reçoit rien) RA : Affichage d'une ligne fixe par dessus le rendu du SDK (grâce à cela il sera possible de lier 2 points de façon à afficher le circuit à suivre)

5/5

Commencement du jeu mediaclap en javascript (Pb : afficher en plein écran : le tableau prend toute la largeur mais pas la hauteur) Les sons ne se jouent pas automatiquement j'ai adapté la classe Media (qui permet de lire un son), pour que l'on puisse lire un ficher

à partir de son chemin RA : Affichage d'un trait entre 2 points + distance (Pb : les coordonnées passent à -10000 quand le point est trop loi en dehors de l'écran / ! Le GPS n'est pas actualisé en intérieur

6/5
Le JS gère la gestion de tous les sons Problème d'affichage des images, elles sont rognées en largeur mais pas en hauteur (Pb : afficher 2 images en background) RA : Recherches sur la façon d'afficher des traits sur les routes pour guider les utilisateurs Test de l'application

10/6
Fin du jeu 4 de médiaclap : mise en transparence des images validées (Pb : le son continue à être joué lorce que l'activité est fermée), affichage de la consigne au milieu de l'écran (Pb : superposition de la consigne, centrer la consigne, et capter le touché de l'utilisateur pour la faire disparaître) RA : Début de la finalisation de la RA Test en situation(Pb : les lignes disparaissent quand on est sur le circuit. Les points disparaissent quand il n'y a plus de lignes)

11/6
Finalisation du jeu4 : mise en place de l'opacité sur les éléments touchés valides (Pb : si on touche une image déjà validé, l'opacité repasse à 1) Mise en place d'un ordre aléatoire des sons Début du jeu des différences Affichage de balises div sur les différences (Pb : les div doivent bouger en prenant en compte le rognage de l'image) Recherche d'une solution pour l'affichage en pourcentage Changement du menu de l'appli de façon à ce que seul les jeux soit affichés

12/6
Les différences sont affichées en pourcentage de l'écran et non en px Affichage d'une image quand une différence est trouvée Un son est joué à chaque fois que l'écran est touché Correction d'un bug qui permettait à un son de continuer à jouer même si le jeu a été quitté Début de la création du fichier HTML du quizz RA : Teste de remplacement des points d'un circuit par une image(Pb : la translation, rotation et mise à l'échelle doivent être gérés) Débug du Pb des traits entre les points qui s'affichent alors que les points ne sont plus affichés Les rond apparaissent sur les traits Si 1 des 2 points qui définissent un trait est absent, le trait se fera entre le point et le milieu du bord bas de l'écran

13/6
Finalisation du quizz (Pb : mettre un background dégradé sans utiliser d'images) Recherches sur la façon d'enchainer les jeux en JS

16/6

regroupement de tous les jeux existant en un seul fichier. Création de fichier .js et .css pour chaque jeux Gestion des transitions(Pb : certains jeux passent tout seul au suivant, au bout de 0.5 secondes) Création d'une classe Java "jeux", qui gère tous les types de jeu. Tentative de correction du bug de transition. Création de la page du jeu "mot masqué" avec une animation CSS(Pb : j'ai du mal à gérer les animation css en javascripte)

17/6

Gestion de l'animation en JS Intégration du jeu3 au système de transitions Test sur smart phone (Pb : le résultat sur PC est différents de celui sur téléphone) RA Recherche et implémentassions d'une solution pour afficher le circuit de façon parallèle à la route, et pour qu'il y ait toujours le chemin d'affiché Test du resultat

18/6

Résolution du problème de double transition Création d'une fonction JS qui met le jeu en portrait Possibilité de mettre une vidéo dans le scénario Lecture de la bonne consigne pour chaque jeux RA Ajout de point intermédiaires entre la projection de l'utilisateur, le le point le plus proche, ainsi qu'entre la projection et le second point le plus proche Affichage en transparence de l'étape suivante quand on s'approche du point le plus proche La projection de l'utilisateur et les points intermédiaires sont affichés selon le point qui est "visé" par la caméra.

I Rendu sans affichage du chemin

Affichage des POIs