Rapport de stage

Année universitaire 2011-2012

Développement d'applications mobiles

Application mobile et web de résolution d'objets astronomiques

Stagiaire: Julien Scheffmann

Maître de stage: André Schaaff

IUT de Saint-Dié-Des-Vosges

Observatoire Astronomique de Strasbourg

Remerciements

Je tiens à remercier en tout premier lieu, André Schaaff, ingénieur de recherche au CDS de l'Observatoire de Strasbourg pour la confiance qu'il ma accordée, l'encadrement qu'il m'a fournit et également pour sa sympathie, sa disponibilité et son aide au quotidien.

Je remercie également les informaticiens, documentalistes et astronomes travaillants au sein de l'Observatoire Astronomique, qui m'ont présentés leurs travaux respectifs. Merci également à eux toutes et tous pour leur sympathie et leur accueil mais aussi pour les cafés des vendredi matins.

Remerciement à Mr.Bourjij, professeur à l'IUT de Saint-Dié, pour m'avoir rendu visite dans les locaux du stage, à Strasbourg.

Merci aussi à Romain, stagiaire en DUT, avec qui j'ai réalisé une partie de la conception et réalisation lors du stage.

Introduction

Le stage a été effectué dans le cadre de ma licence professionnelle effectuée à l'IUT de Saint-Dié-Des-Vosges. Le stage proposé par André Schaaff m'a tout de suite intéressé, et j'ai eu grande joie à travailler sur sur ce projet.

L'IUT m'avait relayé par email cette proposition de stage, j'ai donc demandé rapidement un entretien avec André, qui s'est bien déroulé, étant donné ma présence actuelle. Et je suis plus qu'heureux de l'opportunité offerte.

Le contenu du stage est composé de deux parties. La première partie de ce stage porte sur le développement d'une application sur iOS. Le langage de programmation Objective-C et le SDK iOS m'intéressent grandement et j'aime développer sur la plateforme mobile d'Apple, aussi bien des applications iPhone que des applications plus spécifiques pour le grand écran de l'iPad.

La deuxième concerne la création d'un prototype de cette même application avec les technologies HTML5/CSS3/Javascript. Ayant vu durant l'année universitaire, et les deux année précédentes lors de mon DUT à Dijon, HTML5 mais également Javascript, pouvoir lier ces deux mondes est une parfaite occasion, d'autant plus qu'il ne s'agit pas simplement de developer deux applications sur deux types de plateformes. Il est d'autant plus appréciable que l'intitulé de ma licence est «Internet & Médias Mobiles», le développement iOS et le développement HTML5 correspondent donc tout à fait au sujet du stage effectué, une véritable chance donc d'allier à la fois les technologies porteuses, au coeur de ma licence, et au coeur de mon intérêt personnel.

Je vais dans ce rapport, présenter en premier lieu le service dans lequel j'ai effectué le stage. Je présenterai ensuite l'existant dans le domaine, suivi du contenu du stage. Je terminerai avec un aperçu des perspectives pour l'application et enfin la conclusion de ce rapport.

I. Présentation générale

1. Présentation de l'Observatoire

1.1. Présentation générale

L'Observatoire Astronomique est situé à Strasbourg, en Alsace. C'est un observatoire des Sciences de l'Univers et également une unité de Recherche du CNRS (Centre National de la Recherche Scientifique) et de l'Université de Strasbourg.

L'Observatoire fut fondé en 1881, il se situe dans le quartier universitaire historique de Strasbourg, dans la même zone géographique, on retrouve les facultés de sciences et de lettres ainsi que d'autres grandes écoles. Il se situe juste à côté du jardin botanique, ce qui lui confère un cadre de travail des plus agréables, ce dernier étant entouré de verdure, en plein centre de Strasbourg. Il est constitué de trois bâtiments qui abritent chacun des services particuliers, et est adossé au planétarium. Ce dernier fut autrefois rattaché à l'observatoire, avant de devenir autonome, il est à noter également que le planétarium accueil chaque jour plusieurs classes ainsi que de nombreux particuliers.

Les missions principales de l'Observatoire sont la recherche et l'enseignement. Il abrite également le CDS (Centre de Données de Strasbourg) la composante dans laquelle mon stage s'est déroulé, et dont je vais parler plus longuement plus tard. Il permet également la diffusion d'informations via tout un panel d'outils utilisés chaque jours par des centaines de personnes à travers le monde, outils que je dévoilerais au fil de cette présentation générale de l'Observatoire et de ses composantes.

L'Observatoire dispose de différents types de profils dans ses rangs. Dans la partie informatique de l'Observatoire, il y a en premier lieux il y a les ingénieurs de recherche qui travail sur un large panels de travaux, de développement informatiques

mais également une partie d'enseignement. Il y a ensuite les ingénieurs d'études qui assurent des missions techniques plus précises, et les ingénieurs assistants.

Dans la partie astronomie, ces derniers disposent de trois types de tâches. La première est la recherche, la deuxième est l'enseignement et ensuite la tâche de service. Cette dernière a la particularité d'être en étroite collaboration avec les autres composantes de l'Observatoire, comme par exemple la validation des informations astronomiques que les documentalistes saisissent, et dont nous parleront plus loin dans la description du CDS.

1.2. Les domaines de recherche

Comme indiqué dans l'introduction ci-précédent, l'Observatoire Astronomique de Strasbourg se décompose en quatre grandes équipes de travail, dont voici ci-dessous un descriptif de trois de ces quatre équipes. L'équipe du CDS sera explicité plus longuement ci-après, le stage s'étant effectué dans cette dernière composante.

A. Hautes énergies - XMM

L'équipe Hautes Énergies de l'Observatoire étudie les objets dégageant une énergie phénoménale tels que les trous noirs, les pulsars ou encore les naines blanches, la quantité d'énergie dégagée par ces objets et impossible à réaliser sur Terre. Ils étudient également les objets compactes dans toutes les galaxies visibles, et les objets stellaires jeunes. Ils effectuent des études observationelles mais également théoriques sur les objets stellaires, comme les trou noirs, dont on ne dispose que de théories.

Cet équipe se repose sur des instruments d'observations terrestres, et sur deux satellites qui utilisent les rayons-X, en orbite autour de la Terre, que sont «Chandra» et «XMM-Newton». Seuls les rayons-X permettent de voir et d'étudier les forces considérables émanant de ces objets, le spectre visible (via les satellites et outils d'observations optiques) étant parfois trop peu suffisant.

L'équipe Hautes énergies dispose également d'une importante participation au «Survey Science Center» (SSC) qui est un consortium auquel l'«ESA» («European Space Agency») a confié l'extraction, l'analyse, l'identification et la publication de toutes les sources issues du satellite «XMM-Newton». L'équipe participe donc en répertoriant les observations et analyses des objets capturés par les systèmes d'observations. Ils ont une forte implication dans la création et le maintien du système de base de données qui gère l'identification des sources, système officiel utilisé par le SSC. Ils travaillent également en forte collaboration avec le CDS qui fournit une importante base de données d'archives, d'images et d'informations qui permettent l'étude des objets observés.

B. Galaxies

L'équipe galaxie s'occupent de problèmes variés qui concernent la naissance et la formation des galaxies mais également l'étude d'objets stellaires constituant ces dernières.

Dans le détail, l'équipe étudie les caractéristiques et les propriétés physiques de nombres de galaxies plus ou moins lointaines, comme les mouvements de gaz et d'étoiles. L'objectif étant par la suite de combiner ces informations liées à l'histoire et l'évolution et la dynamiques des objets stellaires ceci afin de reconstituer les évènements clés de la vie de ces galaxies.

Un autre domaine de recherche est l'étude des amas stellaires (concentration locale d'étoiles) en construisant numériquement des amas virtuels via la combinaison de connaissances physiques sur les étoiles et l'évolution de ces amas stellaires. Ce la permettant de récréer le plus fidèlement possible leur évolution.

2. Présentation du CDS

Le CDS est une des composante importante de l'Observatoire Astronomique de Strasbourg, il joue un rôle important dans le monde de l'astronomie sur le plan international.

2.1. Présentation générale

Le Centre de Données astronomiques de Strasbourg (CDS) fut créé en 1972 et est composé d'environ une trentaine de personnes, décomposé environ d'un tiers d'astronomes, d'un tiers d'informaticiens et d'un tiers de documentalistes. Il dispose aujourd'hui d'une mission scientifique et d'une place importante pour la communauté astrophysique internationale. sPour synthétiser le rôle de ce dernier, il permet la collecte, l'analyse et la diffusion d'informations et d'outils gratuitement, le tout au bénéfice de la communauté. Pour ce faire le CDS dispose de ses propres locaux et également de se propre salle de serveur.

Le CDS est une référence mondiale dans le domaine de l'astronomie. Depuis la fin des années 2009, il fait partis des TGIR (Très Grandes Infrastructures de Recherche). Les TGIR sont gérés par le CNRS, par le Ministère de la Recherche et agit en collaboration avec ses partenaires européens et internationaux, le tout pour le bénéfice de l'ensemble de la communauté scientifique.

Ils concernent des chercheurs de toutes les disciplines comme l'astronomie, la biologie ou encore la physique et permet ainsi à tout ses domaines d'avoir accès aux équipements les plus performants dans un environnement scientifique international via l'utilisation de télescopes, d'accélérateurs, de lasers ou encore de moyens de calculs intensifs. Faire partie des TGIR permet au CDS d'obtenir des financements

complémentaires qui permettent donc au CDS de financer des contrats et d'investir dans de nouveaux équipements.

Le CDS est dans une situation paradoxale car il fait partit des TGIR mais son infrastructure est de taille beaucoup plus réduite, et les budgets lui étant alloué sont eux de dix à plus de cent fois inférieur à ceux généralement accordés à d'autres TGIR, preuve qu'avec peu de moyens, l'Observatoire et le CDS en particulier permettent de réaliser un travail très important et reconnus. Il est à noté que le travail du CDS est un travail reconnu et utilisé tous les jours à travers le monde, le CDS ayant acquis une solide réputation dans la communauté internationale de l'astronomie par la multitude d'outils de qualité proposés et par sa forte implication dans de nombreux projets avancés, et dans des activités de coordination d'ordre internationales.

Et pour finir en ajoutant que dans les années 2000, le CDS a participé à l'émergence de ce qu'on appel l'«Observatoire Virtuel», dont la description va suivre, via sa forte implication dans de nombreux projets européens (Euro VOTECH, EuroVO AIDA, EuroVO ICE...) tous en liaison avec l'«Observatoire Virtuel». Ce dernier sera plus longuement décris ci-après la description des services offerts par le CDS.

2.2. Les services proposés

Le CDS a développé au cours de son existence, de nombreux services et outils permettant d'exploiter de manière automatisé ou de manière plus visuelle, les données émanant de ses services, et mis à disposition de la communauté internationale. En n'oubliant pas de préciser que tout ses services ont été crées et sont maintenus par le CDS avec ses propres infrastructures.

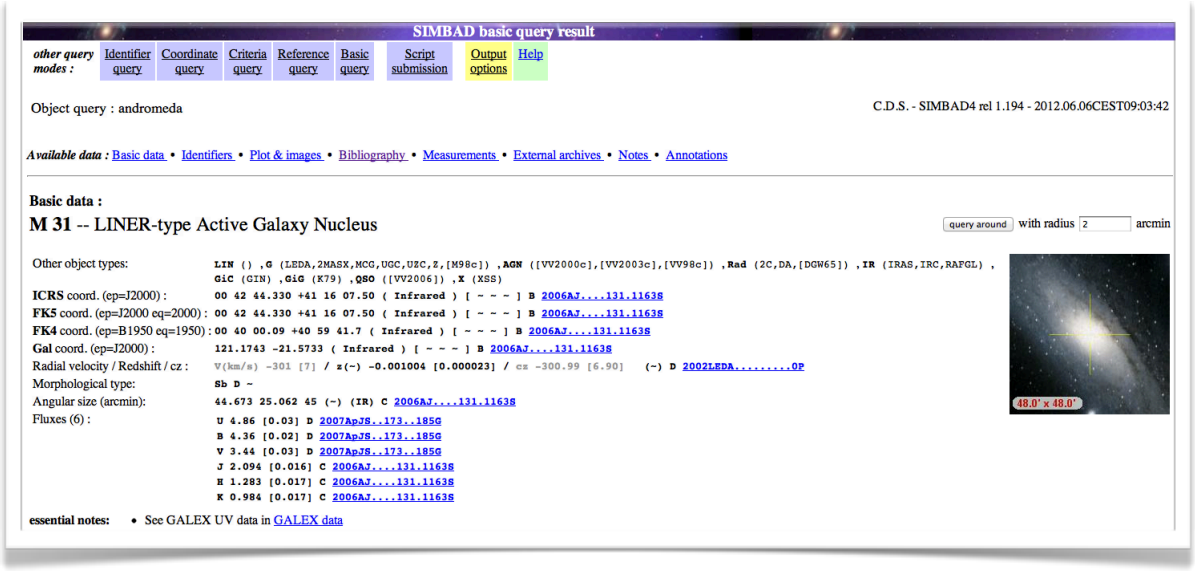
A.Simbad

Simbad est une base de données de référence d'objets astronomiques. Ces dernières son organisé pour permettre la recherche via le nom propre d'un objet, comme la galaxie d'Andromède, mais uniquement en anglais - Andromeda, ou bien par son identifiant, dans le cas d'Andromède c'est M31, ou bien par un de ses 28 autres identifiants secondaires.

Simbad est une base de donnée qui est accessible via une interface web (http://simbad.u-strasbg.fr/simbad/) mais également interrogeable via des services de type REST, pour permettre la création d'outils de recherche et de requête sur les serveurs Simbad de manière automatisée ou de les intégrés dans d'autres applications.

La recherche via Simbad permet de récupérer l'image de cet objet, prise avec différentes techniques et outils selon l'objet, toute une série de cordonnées mais également d'inter-agir avec les autres outils comme nous allons le voir avec VizieR et Aladin.

La base de données de Simbad contient à l'heure d'écriture de ces lignes, presque 7 millions d'objets et quelque 269000 références bibliographiques concernant les objets répertoriés. Tout ce travail d'alimentation de la base de données de Simbad s'effectue avec le travail des documentalistes, qui sont une dizaine, et qui répertorie et corrigent chaque jour plusieurs milliers d'objets, en plus de l'ajout des bibliographies et des articles liés aux objets de cette base de données. Ce premier maillon est primordial pour les autres équipes, qui s'appuient sur cette base de donnée, alimentée par le travail de l'équipe de documentalistes.



Exemple de résultat de recherche sur l'objet M31 (Andromède) via Simbad en ligne. (Figure 1)

B.VizieR

VizieR est une base de données contenant des références de plus de 10000 catalogues d'objets astronomiques, ces derniers étant des observations relevées de mission spatiales et d'observations.

VizieR dispose d'outils permettant à l'utilisateur de sélectionner les valeurs souhaitées, d'en extraire et de formater les résultats selon les critères de recherche donnés. VizieR permet également un accès tout particulièrement optimisé à de très larges catalogues comme «2MASS» qui est le plus utilisé à travers le monde. Il est possible d'interroger VizieR en lui fournissant des critères tel que la longueur d'onde avec laquelle un objet a été observé ou encore via la mission correspondante à cet objet.

Page d'accueil du moteur de recherche de VizieR. (Figure 2)

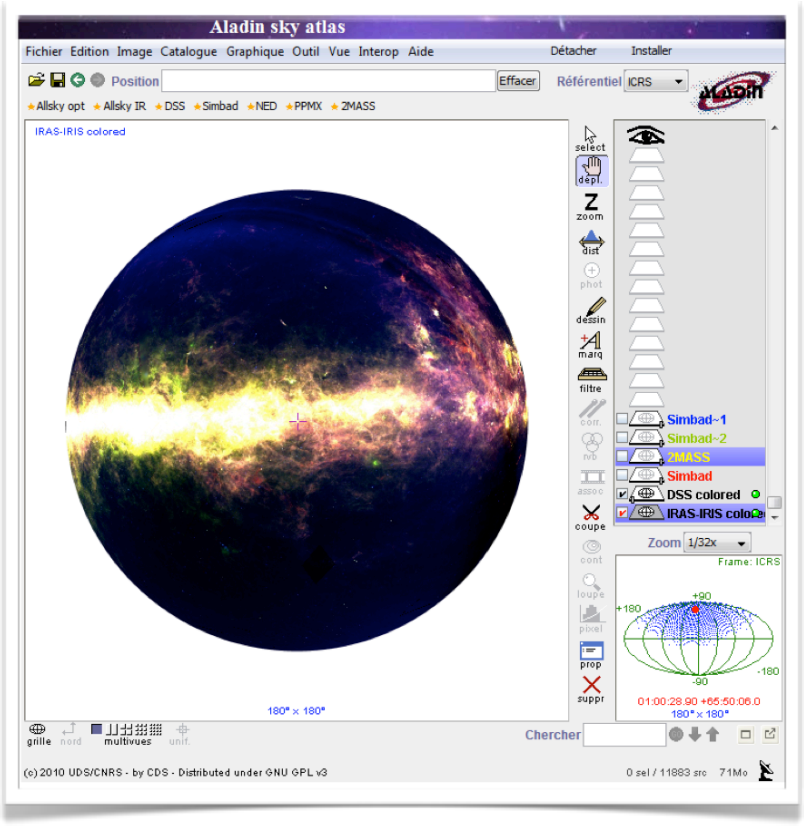
C.Aladin

Aladin est un double système. C'est à l'origine un serveur d'image disposant de sa propre base de donnée. Mais c'est aussi un logiciel développé par le CDS.

Il y a une quinzaine d'années, des membres du CDS ont développé une interface d'accès à cette base de donnée en Java, cette application remplissait correctement son utilisation et le développement de cette dernière a ainsi été poursuivit, devenant un client Aladin. Cette application a pour particularité d'être un atlas interactif du ciel, qui récupère les images d'Aladin mais également d'autres base d'images astronomiques, pour recréer la carte du ciel.

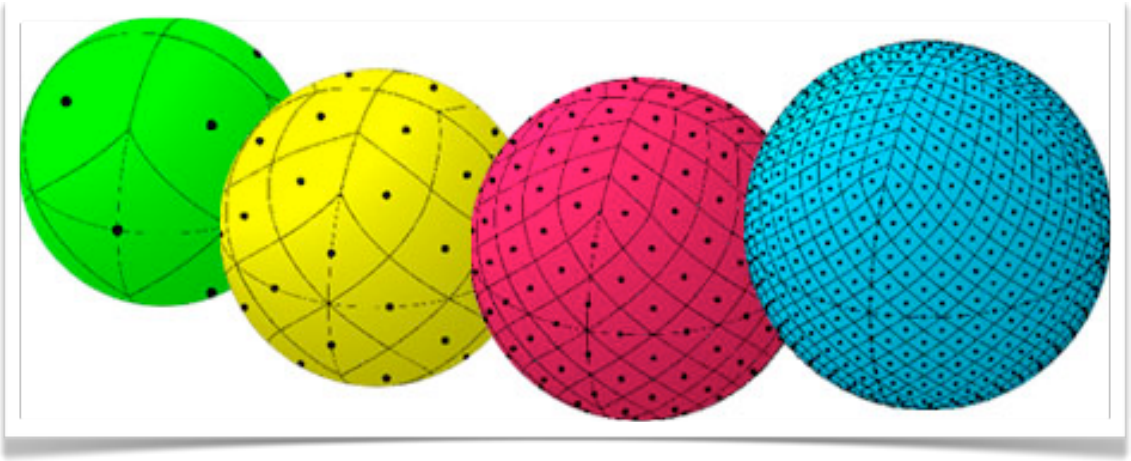
Cet atlas interactif permet de visualiser le ciel mais dispose également de fonction comme «AllSky» permettant de visualiser le ciel sous forme d'un globe. Il permet aussi de visualiser ce ciel en utilisant différents moyens d'observation comme les infra-rouges, les rayons-X ou encore l'optique classique. Il permet de visualiser les objets et en cliquant sur un de ces objets, de voir toutes les informations liées à cet objet.

Il permet également de pouvoir comparer des relevés du ciel, de superposer différentes images sur différents relevés avec différentes longueurs d'ondes et dispose également d'une grande interaction avec d'autres outils logiciels du CDS et d'autres organisations via l'utilisations de standard comme le SAMP. Ce dernier est un standard qui a été créer par l'IVOA («International Virtual Observatory Alliance» - dont je vais expliciter la fonction plus loin dans ce rapport) pour permettre une plus grande compatibilité entre les application développées au sein d'organismes hétéroclites.



Capture d'écran du logiciel Aladin avec une vue «AllSky». (Figure 3)

Aladin utilise également «HEALPix» qui est développé par la NASA (Agence Spatiale des Etats-Unis) et qui permet à partir d'image planes, et via l'utilisation d'algorithmes, de re-créer une sphère par projection, à partir de ces images.



Représentation graphique de la découpe des images sphériques. (Figure 4)

3.Présentation de l'Observatoire Virtuel

L'Observatoire Virtuel fait parti des importantes participations de l'Observatoire Astronomique de Strasbourg, auquel il participe activement via son penchant français («VO France») mais il est également un moteur important de ce dernier.

3.1.Présentation générale

Il existe dans la communauté astronomique de nombreuses bases de données et de multiples applications clients/serveurs qui diffusent toutes des informations non uniformisées. L'observatoire Virtuel est une collection d'archives, de services et d'outils informatiques permettant de créer un environnement scientifique international plus uniforme pour un partage et un accès à l'information simplifié et plus efficace. L'observatoire virtuel existe en tant que différents projets à travers le monde, comme par exemple l'«EuroVO» qui est le projet d'observatoire virtuel européen ou encore l'«US National Virtual Observatory» américain.

Le but de l'Observatoire Virtuel et justement d'unifier les données échangées par la communauté, et le CDS prends une part importante dans la construction de standards d'échanges de données, de création d'outils d'interrogations des bases de données, d'extraction de ces données, afin de permettre une homogénéisation des échanges à travers le globe.

3.2.L'IVOA

Pour synchroniser l'action des différents observatoires virtuels, un consortium a été créer en juin 2002 et qui se nomme l'«International Virtual Observatory Alliance» (IVOA). Il fut créé afin de faciliter la coordination internationale et de permettre la collaboration nécessaire au développement et au déploiement d'outils, de systèmes et de structures organisationnelles nécessaires à l'utilisation au niveau international d'archives astronomiques au sein d'un observatoire virtuel interconnecté. L'IVOA comprends actuellement 20 projets d'observatoires virtuels à travers le monde, dont la France.

Il se concentre sur le développement de standards et encourage à leur implémentation pour le bénéfice de la communauté astronomique internationale. L'IVOA dispose d'un groupe de travail qui développe des standards et des technologies sur le modèle du «W3C» («World Wide Web Consortium» - le consortium chargé de définir les standards du web) dans lequel des groupes définissent des brouillons qui obtiennent par la suite le statut de proposition, avant d'être finalement des recommandations. Comme tout standards, la vie de ces derniers ne tient qu'au fait que la communauté utilise ce standard à grand échelle. C'est pourquoi c'est sur les étudiants en astronomie que des efforts sont portés afin que ces derniers appliquent ces recommandations à leurs méthodes, et fournissent par la suite des informations dont la forme correspond aux recommandations de l'IVOA.

3.3.L'action du CDS au sein de l'IVOA

L'action du CDS est directement liée à sa participation à «VO France» qui existe depuis plus de 40 ans, et qui est largement impliqué dans les collaborations internationales. Le CDS collabore dans de nombreux projets, et le cadre du stage actuel rendre dans ce cercle, ou le CDS développe des systèmes pour le bénéfice de la communauté, et en accord avec les recommandations de l'IVOA.

Le CDS participe chaque année à des rendez-vous où des centaines de personnes venues du monde entier viennent présenter les nouveautés et amélioration à la communauté. Durant la dernière conférence à Chicago, une pré-version du résultat de mon travail de stage et de mes collègues a été présenté par André.

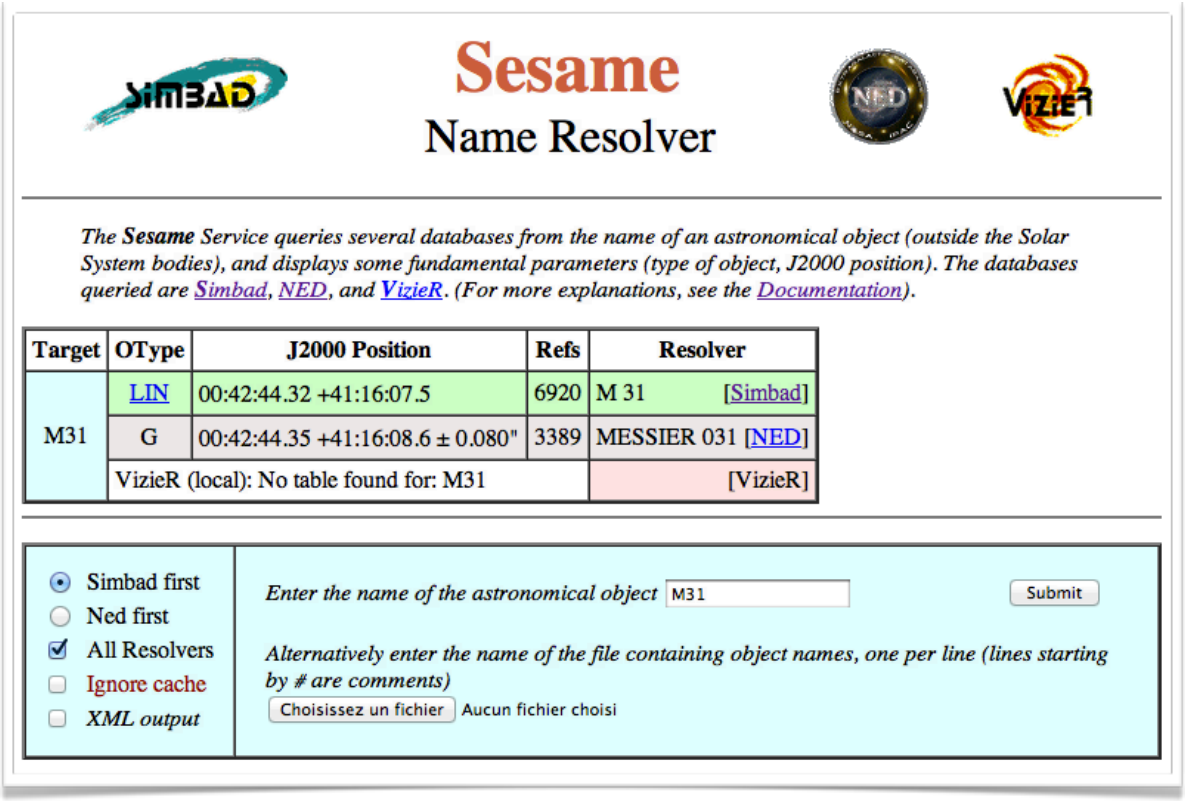
II.L'existant dans le domaine

La présentation du cadre du stage étant faite, je vais présenter l'existant dans le domaine de mon stage. Dans cette section je vais expliciter les services qui existaient à l'origine et qui sont le point de départ de mon stage.

1.Sésame

Sésame est un service en ligne créé, hébergé et maintenu à jour par le CDS. Il s'agit d'un service permettant, à partir d'une chaine correspondant à la dénomination d'un objet astronomique, d'obtenir sa position dans le ciel est quelques autres détails sur ce dernier. Sésame se charge d'aller requêter dans différents «resolvers» pour retourner les informations. Nous verrons ci-dessous ce que son ces «resolvers».

Sésame dispose de deux interfaces d'accès. La première est un formulaire en ligne, qui à partir du nom d'un objet ou un de ses identifiants permet de retourner un affichage très succincts des informations de cet objet comme présenté ci-dessous:



Page de présentation de résultat dans le formulaire en ligne de Sésame. (Figure 5)

Il est possible à partir de cet affichage de sélectionner un des liens s'affichant sous la partie du tableau intitulé «Resolver» pour accéder à l'interface web du service lié. Ici par exemple en sélectionnant «Simbad» sous «Resolver» on accède à la page sur Simbad contenant les informations sur l'objet demandé. Ces informations sont de ce fait beaucoup plus complètes que via l'affichage classique que propose Sésame. Car le but de Sésame n'est pas de dupliquer la fonction de recherche de Simbad, ça ne ferait que réduire son utilité et réduire la visibilité des outils, mais de permettre une recherche dans d'autres catalogues.

Page présentant le résultat de l'objet sélectionné dans Sésame (pour le resolver Simbad). (Figure 6)

La deuxième interface est une interface de type service web qui est interrogeable via d'autres systèmes de type SOAP, c'est ce système que nous utilisons pour récupérer les données lors du développement de notre sujet de stage. Ce dernier permet de récupérer le XML présenté ci-dessous (tout du moins une petite partie) via une URL en lui fournissant certains paramètres dont le nom de l'objet recherché:

Résultat XML de la requête sur l'objet M31. (Figure 7)

2.Les «Resolvers»

Un «resolver» est un service en ligne qui reçoit une demande et qui renvois un résultat donné, selon des paramètres donnés par la personne/le système questionnant ce dernier. Au CDS, et durant le développement de mon sujet de stage, nous en avons utiliser trois différents.

Le premier n'est autre que Simbad et VizieR. Ces deux «resolvers» explicitées précédemment, renvois pour le premier un grand nombre d'informations, VizieR quand à lui renvois peu souvent des informations, étant donné la taille bien inférieur de ce dernier par rapport à Simbad qui contient plusieurs millions d'objets. Ce sont donc des interfaces permettant de faire des requêtes de manière rapide et de centraliser différentes sources d'informations qui peuvent avoir des structures hétérogènes.

Le dernier «resolver» utilisé est américain et se nomme NED («NASA/IPAC Extragalactic Database» - base de données d'objets extragalactiques de la NASA/IPAC). Parmi les informations que NED fournis, rares sont les différences avec son homologue français Simbad. La vraie différence et le nombre considérable d'objets qu'il contient. En tout plus de 170 millions d'objets, ce qui en fait la base d'objet extragalactiques (qui est en-dehors de notre galaxie) la plus importante à ce jour.

Les «resolvers» sont à la base du sujet de stage, ces derniers permettent d'alimenter notre système en informations, via différents moyens d'interrogation que nous allons expliciter par la suite dans le rapport.

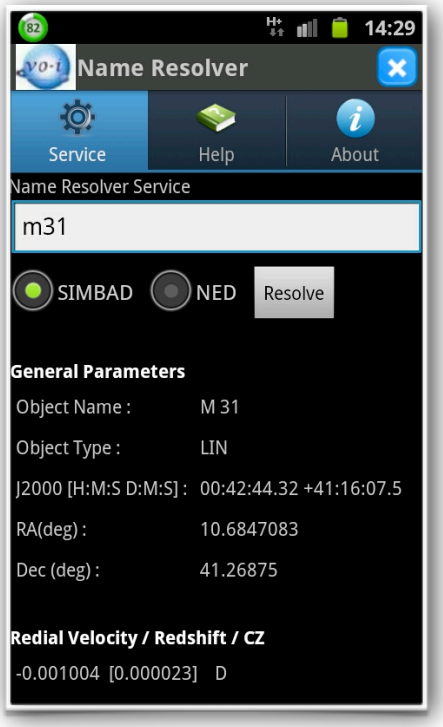
III.Le stage

1.Le sujet du stage

1.1.La demande

L'intitulé du sujet de stage est «développement d'applications mobiles». L'application mobile première qui devait être créée était une application qui devait afficher sur l'appareil de l'utilisateur les informations sur un objet demandé par ce dernier via un formulaire de recherche.

A l'image de ce que font les autres moteurs de recherche d'objets du CDS comme Simbad, l'application doit permettre de récupérer et d'afficher sur une même vue plusieurs types d'informations, issues de différentes sources, pour plus de précision et pour permettre un plus grand choix à l'utilisateur.



Un exemple d'application très simple existant de le domaine est celle développée par le «VO -i» (l'«Observatoire Virtuel» indien) qui est destinée aux plateformes Android, dont voici ci-contre une capture d'écran.

Suite au développement de cette version mobile, une version identique mais avec les technologies du web (HTML5, Javascript, CSS3) sur les points primordiaux devaient être développée. Ce double développement doit permettre par la suite d'effectuer une analyse comparative des deux méthodes de programmation afin d'évaluer l'intérêt d'un développement natif par rapport à un développement web. Nous reviendrons sur ce point plus tard dans ces lignes.

Ce stage s'inscrit directement dans une politique d'avancée technologique et de modernisation des outils, toujours au bénéfice de la communauté astronomique. Le CDS et à travers lui le CNRS, sont des organismes visant la recherche et n'existent pas dans l'immobilité inféconde mais dans le mouvement et l'innovation, en accord avec les technologies modernes qui émergent. Le CDS s'accorde dans son organisation à choisir les pistes technologies prometteuse et portent leurs efforts sur ces dernières pour innover toujours d'avantage.

1.2.Le cahier des charges

A.Plateforme mobile

Dans un premier temps, il est à noter que l'application mobile à développer sera disponible sur deux types de plateformes. Dans mon cas, j'ai du développer cette application pour iOS (système d'exploitation des appareils mobiles d'Apple) et mon autre collègue avec qui nous partageons le bureau, a du développer la version sur Android (système d'exploitation mobile de Google).

Il s'agit donc d'une application mobile sur les plateformes iOS mais qui se doit également de fonctionner sur iPhone, iPad et iPod Touch, c'est donc une app qui sera universelle, pour permettre à l'utilisateur de l'exécuter sur la plateforme de son choix, mais surtout de faciliter le maintient à jour de l'application, étant donné qu'un seul code est généré pour tous les terminaux mobiles d'Apple.

B.Gestion de la consommation de données

L'utilisation d'une plateforme mobile, implique que cette dernière est généralement connectée au réseau internet via un forfait 3G, étant donné que les smartphones iOS sont généralement fournis avec ce genre de contrat opérateur. Ceci signifie que l'application doit faire attention à consommer le moins de données possible, ceci afin de ne pas surcharger la facture de l'utilisateur en connexion trop gourmandes aux services du CDS.

Il faut prendre en considération que ces smartphones vont être à l'avenir de plus en plus connectés à l'internet, mais que pour le moment certains forfaits sont encore onéreux avec des données trop limitées encore, et une connexion WiFi n'est pas toujours disponible.

L'application doit donc permettre de consommer au minimum le forfait de l'utilisateur en limitant au maximum l'accès au réseau. Mais l'application doit pouvoir effectuer certaines tâches qui ne peuvent être réalisées qu'avec une connexion. C'est pourquoi l'utilisateur doit être averti et doit pouvoir choisir par lui-même s'il est d'accord pour que l'application utilise sa connexion internet au réseau mobile.

C.Hors connexion

La gestion de la consommation amène donc à proposer une application qui doit pouvoir fonctionner même si aucune connexion à internet n'est disponible, pour deux raison. Premièrement si l'utilisateur ne souhaite pas autoriser l'application à se connecter à l'internet, et deuxièmement si l'utilisateur, même en ayant autorisé la connexion au web, n'a pas de réseau disponible. L'application doit donc fonctionner en tout indépendance vis-à-vis de l'internet et du choix de l'utilisateur.

D.Langues

L'anglais étant la langue la plus couramment utilisée à travers le globe, l'application doit proposer cette langue. Cette dernière doit également être disponible en français, et le changement de langue doit se faire sans l'action volontaire de l'utilisateur, la langue doit s'auto-sélectionner. C'est ainsi que la langue à utiliser sera celle du téléphone de l'utilisateur. Si la langue de son téléphone n'est pas reconnue dans l'application, alors la langue anglaise sera sélectionnée par défaut. Enfin l'architecture de l'application doit permettre une future évolution simple à mettre en place pour l'ajouts d'autres langues.

E.Version web

Le cahier des charges est identique pour la version web, cette dernière doit pouvoir être utilisable en mode «hors-ligne» ce qui signifie obligatoirement de gérer une base de données sur le poste de l'utilisateur, et ce avec ce que HTML5 propose. L'application n'a pas à reprendre la même qualité aussi bien dans l'ergonomie que dans l'esthétique, elle doit simplement être fonctionnelle car elle fait office de prototype. Elle n'a pas pour but premier d'être utilisable directement comme service web, c'est à dire avec tout les raffinements et toutes les options qu'une telle application pourrait offrir.

2.La découverte

2.1.Découverte des locaux et personnels

Le premier contact avec les locaux et quelques membres du personnel a été le premier jour où je suis venu à l'Observatoire pour mon entretien avec André Schaaff. Et le cadre y est parfait. Le calme et la verdure caractérise ce lieu qui est en pleine «Forêt Noire» de Strasbourg, dans un quartier agréable et bien desservi en terme de transports en communs (tramways, bus) et personnels avec les bornes à vélos à location de la ville de Strasbourg.

Le bureau qui nous a été confié est très lumineux bien qu'un peu bruyant au fil des classes passant devant nos fenêtres pour aller au Planétarium qui se situe juste à côté de nos bureaux, dans les locaux du CDS. Mais le calme règne parfois au coeur des classes, et parfois non, et nous nous transformons en attractions pour ces derniers.

Toujours est-il que même si les classes peuvent être turbulentes, le personnels que j'ai eu l'occasion de croiser et de discuter avec ne l'est absolument pas. Bien au contraire, la sympathie globale règne, même si deux mois ne suffisent pas pour en faire le tour. J'ai pu lors de la première semaine, et avec mes collègues, rencontrer certains membres de l'équipe qui nous ont fait découvrir leurs travaux aux quotidien. Cette découverte du cadre de travail et de ceux qui participent à ce dernier a été fort intéressant.

2.2.Découverte des outils

A.Matériels

Dans mon bureau m'attendait un iMac, outils essentiel pour la programmation sur plateformes iOS. En effet Xcode, l'environnement de développement propre à la plateforme Mac OS, est l'outil obligatoire pour pouvoir exécuter le SDK iOS («Software Development Kit» - Kit de développement utilisé pour gérer les spécificité des plateformes mobiles iOS). La programmation sur iOS utilise le langage de programmation «Objectif-C» il est donc possible de programmer sans Xcode et de compiler le code sur un Mac mais aussi sur un PC.

B.Xcode

Cependant, l'utilisation de Xcode permet l'ajout de fonctionnalités importantes pour le développement sur iOS mais aussi sur Mac OS. Il permet en premier lieu d'utiliser un émulateur d'iPhone mais aussi d'iPad, outils essentiel pour tester rapidement de modifications mineures du code. Le deuxième outil est l'utilisation du terminal intégré à Xcode pour permettre le débogage de l'application, mais aussi l'utilisation d'un outil spécifique qu'est les «NSZombie».

Les «NSZombie» m'ont été très utiles dans le développement de l'application. En effet, le SDK iOS gère le «garbage collector», sorte de ramasse miette qui supprime les variables inutilisées de la mémoire de manière automatique, permettant ainsi une gestion de la vie des objets simplifiée à l'extrême puisqu'aucune action du développeur n'est nécessaire. Ce système s'appel sur Xcode «ARC» pour «Automatic Reference Counting», il se situe au niveau de la compilation et utilise des compteurs qu'ils décrémente pour s'assurer qu'un objet et encore utilisé ou non. Ceci implique que des objets et variables peuvent être supprimées alors même qu'elle avaient encore une utilité.

«NSZombie» est un paramètre à activer pour les tests uniquement, et qui permet d'éviter que les variables ne soient supprimées lors de l'exécution du code. Permettant ainsi d'une part de s'assurer qu'une variable n'est pas corrompue mais également d'obtenir plus d'informations sur un processus via le terminal. Plus d'informations que ne le permet Xcode par défaut. Bien entendu cette fonction est à utiliser avec précaution afin de ne pas surcharger la mémoire de l'appareil, bien que ces derniers disposent de plus en plus de mémoire vive, elle n'en reste pas moins une fonction de débogage propre à Xcode.

C.Coda

Coda est une application disponible uniquement sur Mac qui est un éditeur avancée pour créer du code avec les technologies du web. Il est également un éditeur de style CSS avancé et simple d'utilisation et un navigateur FTP. J'ai utilisé Coda dans sa version 2 pour une bonne raison. Coda 2 dispose d'une nouvelle fonctionnalité qui

permet de partager du code en temps réel sur un réseau WiFi donné avec une personne utilisant également l'application. Ceci permet de voir les modifications en temps réel sur les écrans des personnes partageant un fichier, et permet également de voir le texte sélectionné sur l'écran de tous les participants avec une couleur différente afin de voir qui a fait quel code.

Cette fonctionnalité a été au coeur de la programmation HTML5 de l'application car mon collègue et moi avons travaillé à deux sur cette dernière, ceci nous a donc permis de travailler à deux sur un même fichier et de s'entre-aider en temps réel sur un même code, sans devoir se déplacer.

2.3.Documentation

Aussi bien pour le développement iOS que HTML5, la documentation est abondante et primordiale. Apple propose, directement intégré au coeur de Xcode, l'accès direct à la documentation. Soit via une combinaison sur une fonction afin d'accéder directement à la documentation de cette fonction et de celles relatives, soit via une interface dédiée avec un moteur de recherche complet.

Pour HTM5, le site du «W3C» a été le plus utile ainsi que celui du «W3C Schools» car ils proposent une description complète des méthodes HTML mais propose également des tableaux de compatibilité avec des exemples concrets afin d'utiliser au mieux les fonctionnalités de HTML5.

Pour l'accès à Sésame, le CDS dispose d'un wiki (système de gestion de site internet permettant de créer de manière collaborative ou restreinte, privé ou publique, de sites web dédiés à l'aide ou encore de créer des encyclopédies en ligne ou sur un réseau local) très complet. Le fonctionnement de Sésame étant simple de par sa nature même, une page complète lui était consacré pour les différents paramètres utilisables pour l'URI d'interrogation du service.

Le CDS disposant de ses propres serveur sur place, dispose également d'espaces privés. Nous avons accès au wiki du CDS et un compte nous avez été réservé. Nous avons donc dans un premier temps pu voir les différents travaux et rapport des stagiaires des années précédentes. Nous avons également eu notre propre page dans ces dernières qui nous a permis d'écrire notre avancement dans le projet et de partager des informations avec notre maître de stage. Au début du stage, André Schaaff (mon maître de stage) avais déjà préparé une page complète avec un descriptif complet du projet, avec l'accès à toute la documentation nécessaire. Nous avons donc échanger des idées préparatoire pour l'application qui devait être développée.

3.Le développement iOS

Cette partie va expliciter tout le développement du travail fait pour la création de l'application qui fait suite à la demande et au cahier des charges. Celle-ci se concentre uniquement sur la première partie du stage, le développement de l'application pour appareils mobiles sous iOS.

3.1.La base de données

A.SQLite

a.Les pré-requis

Dès les bases de ce que l'application doit pouvoir fournir en terme d'accès hors-ligne, la question de la solution de base de données à utiliser s'est posée. Celle-ci doit respecter trois critères importants.

Le système à utiliser se doit d'être gratuit, et ne doit pas nécessiter de licences ni d'autorisations spécifiques pour être implémenter dans notre application. Le CDS n'ayant pas de moyens à verser dans des solutions payantes qui bien souvent son propriétaires et donc peu enclin à évolution et risquent même parfois la disparition du marché et qui bien souvent ne valent pas des solutions gratuites. De plus, l'application finale doit être gratuite à celui voulant l'utiliser, le CNRS n'ayant pas de but lucratif.

Le système se doit également d'être performant en terme d'accès et de structuration de l'information. Etant donné son utilisation sur une plateforme mobile, et étant donné la façon dont il va être utilisé (dont nous parlerons de manière plus ample dans la partie dédiée à l'interface utilisateur) celle-ci se doit d'être rapide et de ne pas nécessiter de trop longs temps d'attente pour retourner l'information demandée.

Finalement, le système de base de données doit être léger. L'intégrer à un système mobile signifie un espace de stockage plus limité qu'un ordinateur classique disposant de plusieurs centaines de giga-octets. Un téléphone par défaut ne dispose généralement que d'une dizaine de giga-octets pour les plus avancés, comme l'iPhone qui dispose au maximum de 64 giga-octets non extensibles. Les appareils sous Android disposent eux, selon les modèles, d'emplacement mémoire pour étendre la capacité de stockage de l'appareil.

b.Présentation de SQLite

SQLite s'est imposé comme la solution qui correspondrait le plus à nos pré-requis et comme étant la plus sérieuse. SQLite est un système de gestion de base de données qui a été crée en 2000 et qui en est aujourd'hui à sa version 3.7, il est de plus

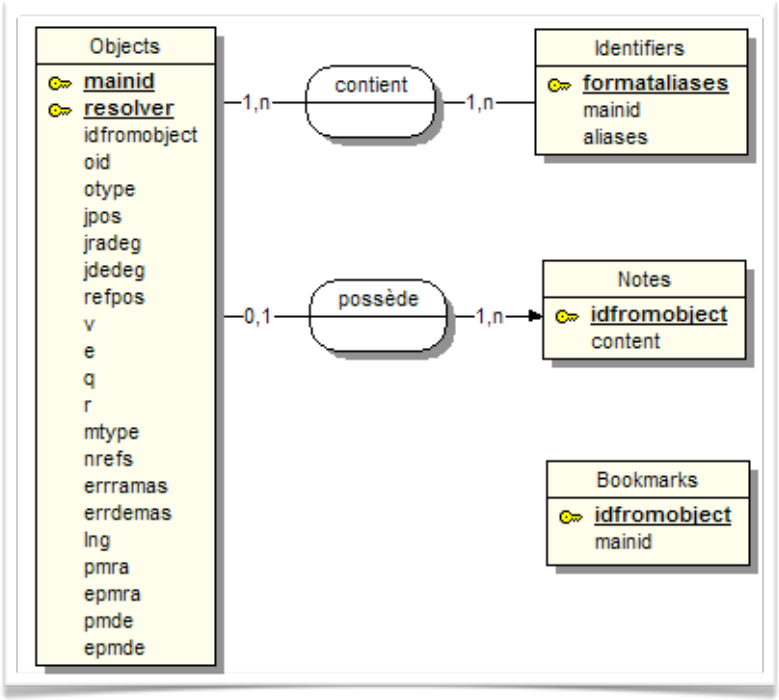
mis à jour très régulièrement, son développement restant très actif. Il est gratuit d'utilisation et sa licence, cette dernière dans le domaine public.

Ce système de gestion de base de données est d'ailleurs le plus réputé dans le domaine. En effet de grandes entreprises multi-nationales l'utilisent telles que Adobe, Airbus ou encore Apple et Microsoft. Sa réputation n'est donc plus à faire.

SQLite est la solution la plus communément utilisée dans les systèmes mobiles pour gérer le stockage des données. La bibliothèque SQLite est d'ailleurs intégrée directement dans la collection de bibliothèques incluse dans Xcode. Ce qui facilite son implémentation dans notre application pour iOS. De plus SQLite utilise le langage SQL classique pour effectuer ses requêtes, langage que nous maîtrisons déjà. Il est de plus très léger, son ajout au code de l'application n'occupe qu'une petite quantité de données.

Il est à noter que Apple propose un système propre à iOS et à la programmation sur Mac OS qui est «CoreData». Celle-ci nous délivre une interface d'accès au système de base de données intégré aux systèmes d'exploitations d'Apple via des méthodes simples. L'inconvénient dans notre situation c'est qu'étant donné que cette application va être portée sur Android également, le système de base de données doit être identique sur les deux plateformes. Ceci afin d'entretenir un seul système à jour, et ne pas à avoir à transposer les données dans deux formats différents.

B.Le contenu de la base de données



Modèle conceptuel allégé des données de notre base de donnés. (Figure 8)

Le modèle conceptuel des données présenté dans la figure 3.1 représente la structure de notre base de données. Celui-ci à néanmoins été simplifiée car dans la table «OBJECTS» à la suite de l'attribut «oid» se trouve 17 autres attributs corresponds à des valeurs propres à l'astronomie, et leurs contenu n'a d'utilité que pour un astronome.

a.La table «Objects»

Cette table contient toutes les informations propres à un objet astronomique donné. Elle contient 18 attributs propres à l'astronomie comme la position de l'objet dans le ciel, ces attributs n'ont d'utilité que pour l'affichage sur la vue de l'objet, pour les astronomes, et ne sont pas utilisés pour le traitement interne de la base de données.

L'attribut «mainid» est fournit pas le fichier XML (dont nous parlerons dans la section suivante) et correspond à un identifiant unique de l'objet. Bien souvent celui-ci est l'identifiant et le nom standard de l'objet, comme «MESSIER 031» pour faire référence à «M31» qui est la galaxie d'Andromède. Ces derniers sont réutilisé tels-quels dans notre base de données et servent à identifier de manière unique un objet, l'attribut étant la clé primaire de cette table.

La deuxième clé primaire est le «resolver» qui comme indiqué précédemment dans ce document, est le système qui fournit l'information. Dans notre base de données, trois «resolvers» sont utilisés étant donné que ce sont les trois utilisés par Sésame. Cette deuxième clé primaire permet d'être sûr de l'unicité d'un objet, étant donné que dans un même «resolver» les «mainid» sont uniques.

Le dernier attribut utile à notre système interne, qui est «idfromobject» a été créée automatiquement par nos soins. Le problème qui se posait était de pouvoir lier automatiquement les informations des trois «resolvers» sous un seul identifiant. Etant donné que pour un seul objet nous interrogeons trois services, la création d'un identifiant commun aux informations retournées par ces trois service était une solution rapide et efficace. Il est ainsi possible d'exécuter une seule requête pour récupérer les informations sur les trois «resolvers» pour un seul et même objet.

Lors de l'ajout d'un objet dans les favoris, si cet objet n'existe pas dans la base de données locale, c'est à dire par exemple si l'objet provient d'une recherche sur internet, alors ce dernier est intégré dans la base de données locale. Afin d'éviter une collision entre le «idfromobject» des objets existants et les «idfromobjects» de ceux que nous allons générés pour l'ajout de ces objets. C'est ainsi que l'on commence la numérotation à partir de 200 millions, car nous avons estimé que le jour où nous atteindront plus de 200 millions d'objets en local n'est pas encore arrivé.

b.La table «Identifiers»

Cette table est directement liée à la table des objets, celle-ci permet de lier un objet à ses nombreux autres identifiants secondaires. Un objet dispose en moyenne

d'une dizaines d'identifiants secondaires. Cette table contient donc le «mainid» qui fait référence à l'identifiant d'un objet. Pour récupérer par la suite les deux autres «resolvers» de cet objet, il suffit d'effectuer une requête sur les «idfromobject».

Les attributs «aliases» et «formataliases» peuvent être considérés comme des doublons, mais ils existent pour une raison bien pratique. En effet les alias d'un objet peuvent parfois contenir de nombreux espaces. Ces espaces posaient problèmes lorsqu'il fallait faire des recherches dans la base de données avec l'utilisation de termes exactes. Par exemple l'objet «NGC5» existe dans la base de données sous l'identifiant «NGC 5» avec cinq espaces blancs entre «NGC» et le «5». Ce qui rendait les recherches plus longues étant donné qu'il fallait effectuer quasi automatiquement une recherche de type «LIKE» dans la liste de plusieurs milliers d'objets. Car nous sommes partit du principe qu'une personne saisie rarement tous ces espaces lors de la recherche d'un objet, et qu'elle ne les utilise quasiment jamais.

Le système effectue donc une première recherche exacte (de type «SELECT * FROM IDENTIFIERS WHERE formataliases = 'ngc5'») sur le nom de l'objet en effectuant une requête sur «formataliases» avec la chaine que l'utilisateur a saisie. De cette dernière chaine ont été retirés les espaces et toutes majuscules. Ainsi si le résultat ne renvoie aucun résultat, alors le système effectue une recherche de type «LIKE» qui elle sera de ce fait plus longue qu'une recherche avec les termes exacts. Nous avons ainsi pu diminuer de manière notable le temps d'attente lorsque l'utilisateur effectue une recherche.

c.Les tables «Notes» et «Bookmarks»

Ces deux tables sont les seules que l'utilisateur remplit dynamiquement par l'utilisation qu'il peut avoir de l'application. Sans rentrer des les détails, puisque nous en parlerons peu après, l'utilisateur a la possibilité de sauvegarder un objet et d'écrire des notes pour ce dernier.

La table «note» contient un identifiant qui fait référence à l'identifiant d'un objet, et contient dans «content» son contenu. La table «bookmarks» contient elle aussi l'identifiant d'un objet «idfromobject» et le «mainid» de ce dernier. Pour savoir si un objet est défini comme favoris ou non, nous récupérons le «idfromobject» de l'objet consulté, et nous recherchons une occurrence de ce dernier dans cette table. Si le résultat n'est pas nul, alors l'objet est défini comme favoris.

d.Occupation mémoire et iCloud

La base de données, une fois installée sur la mémoire du téléphone occupe environ un espace de 18 méga-octets, ce qui est une occupation mémoire de taille correcte. Il faut ajouter que cette base de données se trouve également dans le «bundle» de l'application, c'est à dire dans l'exécutable qui permet l'installation de l'application sur l'appareil de l'utilisateur. Cependant cette base de données est compressée dans le «bundle» car elle utilise une place importante pour un fichier qui n'est utilisé que très peu et normalement jamais.

Un autre point intéressant en ce qui concerne l'espace de stockage et son utilisation, Apple propose depuis quelques temps «iCloud», un service de stockage en ligne, que tout les utilisateurs d'iPhone, d'iPad ou bien d'iPod Touch peuvent avoir gratuitement (5 giga-octets offerts). Cet espace de stockage ne peut être géré que par les applications qui y accèdent, ces dernières pouvant ajouter, modifier ou supprimer les informations qu'elles ont elles mêmes envoyé sur les serveurs de ce service.

Ce dernier propose également la sauvegarde automatique des applications installées sur l'appareil de l'utilisateur. C'est ainsi qu'Apple demande aux développeurs de placer convenablement les fichiers dans des espaces bien précis dans le système de fichier iOS. Ainsi les fichiers générés par l'utilisateur, et qui sont donc unique, doivent être placés dans le dossier «Documents» alors que les fichiers pouvant être restaurés simplement par le «bundle» par exemple, doivent être placés dans le dossier «Temp». Ceci permet de limiter la masse d'information à stocker sur les serveurs d'Apple et de ne pas surcharger inutilement le compte de l'utilisateur.

Apple contrôlant de manière stricte, selon les cas, les applications avant de les faire paraître sur l'App Store (le magasin en ligne proposant le téléchargement des applications pour iOS) il a fallu faire bien attention à ce détails afin de ne pas se voir refuser l'application, sachant que les éléments techniques permettant de se mettre en conformité avec Apple ne sont pas toujours très clairs.

C.Le XML

Comme indiqué précédemment, Sésame dispose d'une interface d'accès qui permet de retourner un fichier XML. Nous disposons donc d'une URI d'accès qui contient le nom de l'objet que l'on souhaite interroger, et Sésame renvoie un fichier XML avec les informations sur cet objet. Il est possible dans cette URI de choisir les «resolvers» à utiliser pour retourner l'information. Nous utilisons toujours les trois dans notre application.

Le XML est utilisé à trois moments dans le processus de développement de l'application. Il est utilisé au moment de la génération de la base de données, ce que nous verrons dans la partie suivante, il est utilisé en temps réel au cours de l'utilisation de l'application et aussi lors de la mise à jour de la base de données (ce que nous verrons également plus tard) pour ce qui est de l'état actuel d'évolution de l'application.

a.XML via interrogation des services

Le premier cas que nous allons voir dans cette section est celui ou un objet demandé par l'utilisateur ne se trouve pas dans la base de données locale. Dans ce cas, l'application va proposer à l'utilisateur d'effectuer cette recherche sur les serveurs du CDS. Cette recherche s'effectue simplement en fournissant comme paramètre le nom de l'objet demandé par l'utilisateur. Le service en ligne de Sésame étant

intelligent, même une orthographe proche est prise en considération pour la recherche.

Le service renvoie par la suite un fichier XML qui contient trois balises «Resolvers» avec des attributs propres à chacun. C'est ainsi que cet XML est traité dans une classe qui se charge d'en extraire toutes les informations et de créer un tableau structuré contenant toutes les informations sur l'objet. Le traitement se fait en tâche de fond afin de ne pas figer l'écran et permettre à l'utilisateur d'annuler sa recherche.

b.NSXMLParser

Pour effectuer le traitement du XML, Apple propose une libraire directement intégrée à tous les projets Xcode, avec une classe nommée «NSXMLParser» qui comme toutes les classes et fonctions développées par Apple, bien que très verbeuses parfois, dispose d'un nom plus qu'explicite. Cette dernière est un «parser» (analyse syntaxique pour mettre en évidence une structure) de type «SAX».

«SAX» signifie «Simple API for XML» ou en français «Interface de programmation simple pour XML». Cette dernière est semblable au type le plus connus qu'est le «DOM» qui a pour inconvénient une certaine lenteur car celui-ci doit parcourir tout le document avant de pouvoir effectuer des traitement sur ce dernier. «SAX» quand à lui permet d'effectuer des action au cours de la lecture du fichier, car il est asynchrone. Ainsi à la fin de la lecture de chaque objet défini par les balises «Resolvers», la fonction développée par mes soins appelle une méthode qui se charge en arrière plan d'insérer l'information dans la base de données.

L'utilisation de «SAX» permet donc un traitement du document beaucoup plus rapide, et permet de diminuer l'empreinte mémoire de son processus, évitant ainsi au maximum un dépassement de mémoire qui aurait inévitablement entrainé un arrêt brutal de l'application. Voir même un risque de corruption de la base de données. Car à la différence de «DOM», la taille d'un arbre «SAX» n'est pas proportionnelle à la taille du fichier traité, fichier qui peut souvent être énorme.

La classe «NSXMLParser» est donc le meilleur compromis entre simplicité d'intégration au système, rapidité d'exécution mais aussi pour sa documentation foisonnante étant donné qu'Apple la documente et que nombre de sites en ligne reprennent l'exemple avec de nombreuses aides et tutoriaux.

D.Génération de la base de données

L'application étant développée sur deux plateformes, l'utilisation d'une même structure de base de données a été demandée. C'est ainsi qu'un petit logiciel a été développé afin de générer la base de donnée qui sert de socle pour les deux plateformes. Cette application a été développée en Java et permet à partir du XML de créer une base de donnée contenant tous les objets voulus. Pour ce faire un fichier en XML a été fournit par le CDS. Ce fichier contient plus de 7000 objets parmi les plus demandés au CDS, ce dernier disposant de tout les outils pour extraire les statistiques

de requête de leurs services et ainsi mettre en évidence les plus demandés au cours des 2 dernières années.

Un fichier XML pesant environ 18 méga-octets avec plus de 500 milles lignes d'informations est donc traité par l'application qui utilise elle le type «DOM» ce qui demande donc plusieurs heures afin de créer le fichier final. Car le fichier SQLite créer est contenu dans un seul et unique fichier, l'application doit donc constamment avoir un accès en écriture sur ce dernier afin de le compléter.

E.Réinitialisation de l'application

Une des fonctionnalités de l'application est de permettre à l'utilisateur de réinitialiser l'application s'il le souhaite. Cette action est simple à mettre en place étant donné que le «bundle» contient encore la base de données complète. Celle-ci est néanmoins compressée afin d'obtenir un gain important d'espace étant donné que cette base ne sert qu'à l'installation de l'application par iOS et également à la restauration de l'application.

Si l'utilisateur le souhaite, la base de données présente dans le système de fichier de l'application, est effacée dans sa totalité (1 fichier SQLite à supprimer) et celle présente dans ce «bundle» est donc recopiée à sa place. Ceci permet de très rapidement remettre les paramètres par défaut en cas de dysfonctionnement, sans faire attendre l'utilisateur. Dans la seconde l'application est utilisable comme à son premier lancement.

a.La configuration par défaut

Pour restaurer les paramètres de l'utilisateur par défaut, comme nous le verrons plus tard dans ce document, un fichier «.plist» est créer et celui-ci contient un certain nombre d'informations qui sont celles par défaut.

cdsServerUrlRequestXml	String	http://cdsweb.u-strasbg.fr/cgi-bin/nph-sesame/-ox/~SNVA?
databaseName	String	cds.sqlite
cdsServerUrlRequestUpdatesXml	String	http:// /sesame
cdsServerUrlRequestInitialiseXml	String	https:// Xml.xml
automaticDatabaseUpdates	String	FALSE
showAlertInternetUnreachableXmlResults	String	TRUE
applicationGlobalName	String	SkyObjects
applicationGlobalVersionNumber	String	0.8
showImageOfTheDay	String	FALSE
urlImageOfTheDay	String	http://alasky.u-strasbg.fr/cgi/simbad-thumbnails/get-thumbr
reportErrors	String	FALSE
lastIdfromObjectFromInternet	String	200000000

Copie d'écran d'une partie du fichier de configuration. (Figure 9)

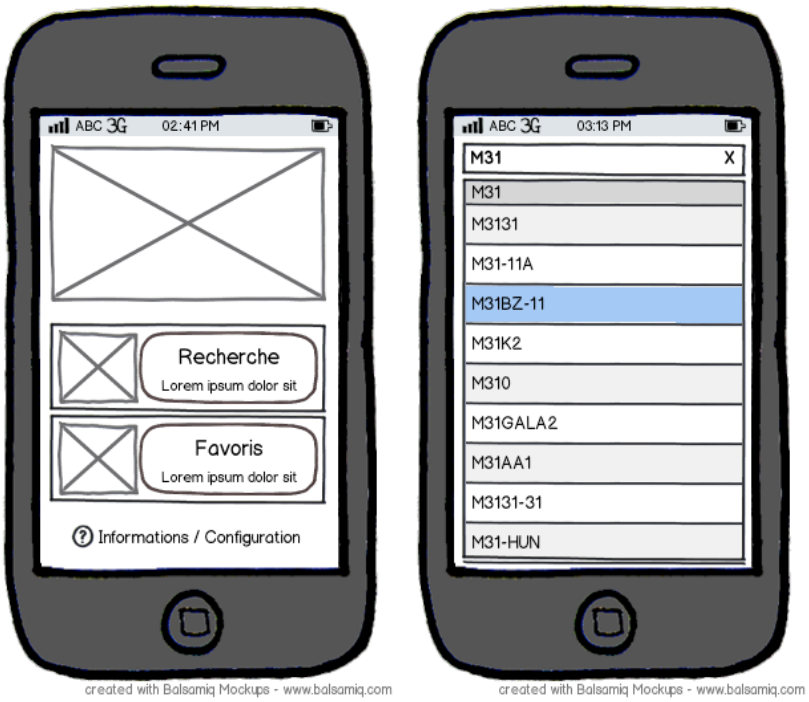
Ce fichier n'est pas modifiable en état, car il sert uniquement à la restauration des paramètres par défaut. Au premier lancement de l'application, une classe se charge de récupérer les valeurs de ce fichier et d'en créer des «NSUserDefaults» ce

qui permet de sauvegarder ses variables et ainsi de pouvoir les réutiliser à n'importe quel moment dans le code de l'application. Ceci permet de gérer un seul fichier de paramètres centralisé et de pouvoir changer son contenu sans modifier le code source de l'application. Car les «NSUserDefaults» sont des variables, lorsque l'utilisateur définit un nouveau paramètre, une seule ligne de code permet de modifier son contenu. Il est ainsi inutile d'alourdir le code avec des fonctions tierces pour écrire dans un fichier les configurations actuelles de l'utilisateur.

3.2.L'interface utilisateur

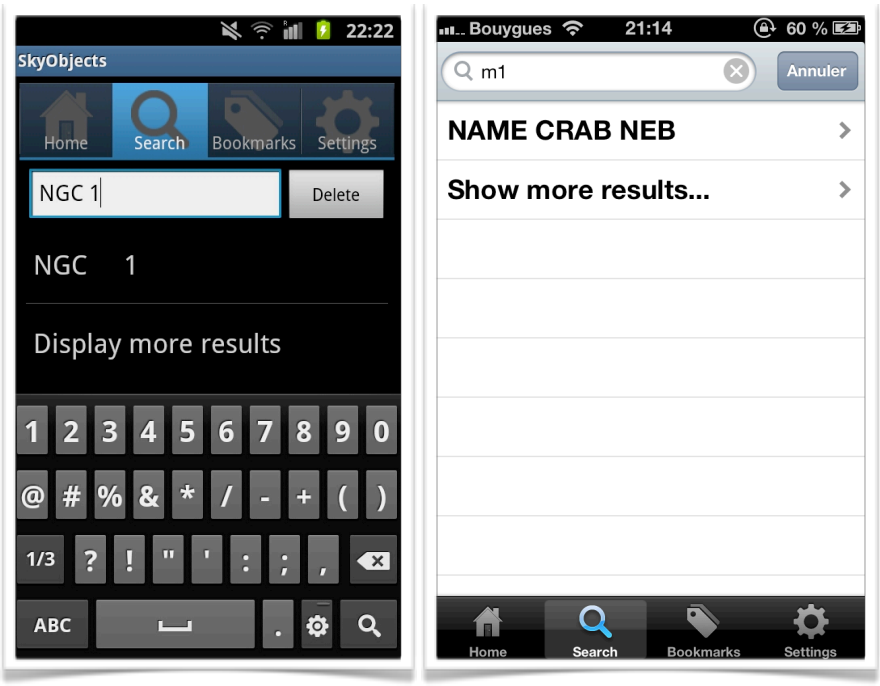
A.Concertation

Une fois la base de données définie, nous avons commencé avec André et surtout mon collègue qui a travaillé sur la version Android à se questionner sur l'interface utilisateur à mettre en place. Nous avons donc fait quelques maquettes d'exemple d'interface utilisateur avant de les présenter à André pour en discuter. Voici ci-dessous deux fenêtre issues d'une maquette, mais celles-ci n'ont pas été appliquées car n'utilisent pas totalement les cannons d'interface d'iOS et d'Android. Car la question était de pouvoir définir une interface la plus proche possible l'une de l'autre sur les deux plateformes ceci afin de garder une certaine unité dans le design pour garder l'identité de l'application, tout en utilisant les canons esthétiques des deux plateformes.



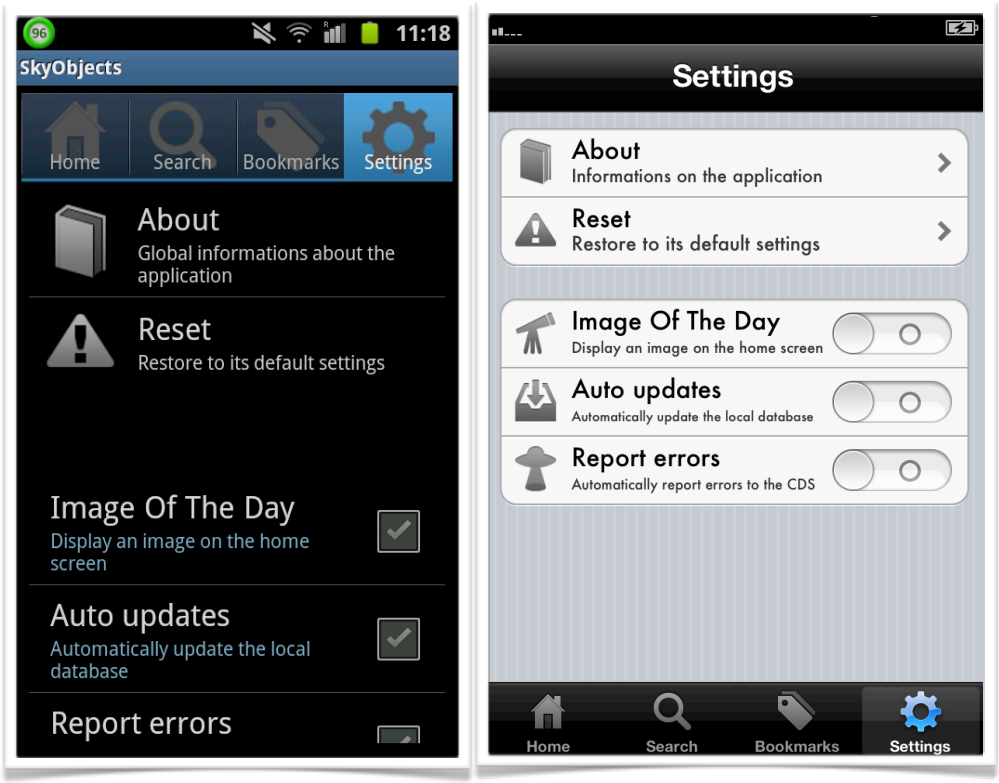
Deux exemples d'interfaces utilisateurs. (Figure 10)

De manière générale, il a été possible de conserver la même structure pour l'interface utilisateur, comme par exemple ci-dessous avec la page de recherche d'un objet.



Page d'affichage du résultat de recherche d'un objet sur iOS (gauche) et Android (droite). (Figure 11)

A l'inverse, certains aspects de l'interface d'iOS non pas trouvé d'équivalent sur Android, ainsi la page de configuration contient des cellules qui disposent toutes du même style, à l'inverse de celles d'Android qui ne peut appliquer des images à tout les types de cellules.



Page de configuration de l'application sur Android (gauche) et iOS (droite). (Figure 12)

Dans les deux applications, nous avons décidé d'utiliser des onglets en bas de page (pour iOS car c'est la façons dont ils sont crées) et en haut pour Android. Ces onglets sont au nombre de 4: la page d'accueil, la recherche, les favoris et la configuration. Dans les sections ci-dessous, nous allons expliciter leurs contenus.

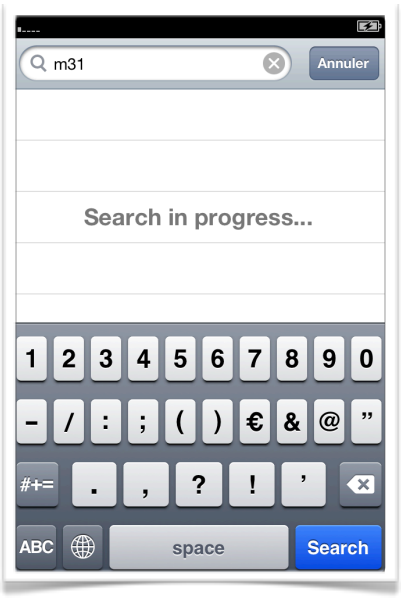
B.La page d'accueil

La page d'accueil et la vue sur laquelle l'utilisateur arrive au tout premier lancement de l'application, c'est la page d'accueil de l'application. Nous avons appliqué à cette page une image de fond qui reprends le thème de l'espace avec cette nébuleuse (image issue de la NASA) et le logo du CDS. Cette page était néanmoins vide, et son existence également. C'est la raison pour laquelle une image du jour viens s'ajouter, en option paramétrable, sur cette dernière.

Tous les jours, au lancement de l'application, le système va piocher un identifiant dans sa base de données interne, et va ensuite interroger le serveur d'image «Aladin» pour récupérer l'image correspondant à l'objet demandé. Les images sont donc choisie par hasard, et non sur des critères esthétiques. Il arrive donc parfois d'avoir une image floue à l'intérêt artistique limité, comme il arrive parfois d'avoir de très belles images de galaxies. Car il s'agit avant tout de rendre vivante l'application, l'image n'intéressant pas forcement l'utilisateur professionnel, mais est plus destinés aux utilisateurs amateurs.

C.La recherche

La page de recherche est la page la plus utilisée et est le coeur de l'application. Cette dernière dispose de plusieurs états, selon la recherche. Dans un premier temps l'utilisateur arrive sur la page de recherche et clique sur la barre de recherche pour y saisir l'objet recherché. Sur la page s'affiche, au milieu des cellules, un texte qui indique à l'utilisateur d'attendre le temps que sa recherche est en cours d'exécution.



Page de recherche d'un objet - première saisie des termes de recherche. (Figure 13)

Pour limiter le temps de traitement d'une requête, le système limite l'affichage de 1 à 5 valeurs au maximum. Cette valeur est facilement modifiable, mais permet d'accélérer l'affichage. Afin d'accélérer encore plus l'affichage, la requête renvoie un tableau contenant uniquement les identifiants à afficher, et non toutes les informations sur tous les objets.

C'est pourquoi dans la dernière cellule s'affiche un champ «Afficher plus résultats» qui une fois sélectionné bloque l'affichage de l'utilisateur et va requêter la suite des identifiants à afficher. Ce blocage sert à empêcher l'utilisateur de changer les termes de la requête pendant le chargement. Car cela entrainerait certains ralentissements, étant donné qu'un processus se charge d'accéder en tâche de fond à la base de données, modifier les termes de la recherche enverrait un nouveau signal de recherche.

Page d'attente permettant de charges de nouveaux objets suite à une recherche. (Figure 14)

Comme indiqué précédemment, un objet n'existant pas dans la base de données locale peut être consulté en interrogeant directement le serveur du CDS. Une cellule contenant le texte «Effectuer la recherche sur les serveurs du CDS» s'affiche. C'est à l'utilisateur de valider le fait d'effectuer cette recherche sur internet, le système ne le fait pas automatiquement, pour comme expliqué précédemment, laisser libre choix de l'utilisation du réseau à l'utilisateur. La structure de l'affichage des résultats est encore en réflexion.

Affichage de la demande de recherche sur internet (gauche) - Résultat de la recherche (droite). (Figure 15)

D.Les favoris

Le troisième onglet disponible est celui réservé aux favoris. Les favoris permettent à l'utilisateur d'accéder directement aux objets qu'il veux utiliser le plus. Ainsi sur cette page s'affichent tout les objets favoris, dans des cellules. Un moteur de recherche de favoris est intégrés pour retrouver rapidement un objet dans la liste qui peut être conséquente. La page des favoris reprends la même interface que celle de la recherche, en utilisant des cellules dans une vue extensible.

Dans la base de donnée, tous les objets désignés comme favoris, vont remplir la table «bookmarks» avec les «idfromobject» de ces derniers. Un objet qui provient d'une recherche en ligne et qui sera ajouté comme favoris, recevra un «idfromobject» supérieur à 200 millions afin de ne pas entrer en conflit avec les objets déjà existants ou allant être ajoutés.

Il aurait pu être inutile de créer une table des favoris étant donné qu'il suffit d'ajouter un champ à un objet permettant de savoir si ce dernier est oui ou non un favoris. Néanmoins créer une table dédiée n'occupe pas plus d'espace, et permet surtout de créer la liste des favoris de manière quasi-instantané car il suffit de sélectionner tout le contenu de cette table pour créer la liste. C'est donc pour une question d'efficacité que cette solution a été retenue.

E.Détails de l'objet

Cette partie va expliciter les différentes parties présentes dans l'onglet de recherche, lorsqu'une personne sélectionne un objet et désire voir afficher les informations sur ce dernier.

Cette vue se décompose en différentes parties, symbolisées par les points de couleur blanche dans le bas de la page. Ces points montrent à l'utilisateur qu'il y a plusieurs pages de disponible, ce système est propre à iOS et est simple à ajouter car c'est un élément de l'interface classique sur cette plateforme. Cette présentation se trouve être en fait un élément de type «ScrollView*» dont on a décomposé la taille en plusieurs pages délimitées par leurs coordonnées de départ et de fin. Créant ainsi de véritables pages que l'utilisateur peut parcourir à sa guise, avec des effets de blocage de la page si l'utilisateur parcours les pages trop rapidement ou bien s'il va trop loin.

Ce système est créé de manière dynamique, il s'adapte donc automatiquement à la taille de l'écran. Il correspond donc parfaitement aux écrans de l'iPhone et à celui de l'iPad. Dans ce système, le nombre de page peut différé selon que l'objet soit définit comme favoris ou non.

E.1.Les resolvers

Les trois première pages de la vue reprennent les informations fournies par les trois «resolver» que sot «Simbad», «VizieR» et l'américain «NED». Les champs n'ayant pas d'information son grisés, afin de montrer à l'utilisateur que l'application fonctionne mais qu'une information n'est pas disponible pour l'objet demandé. Les champs disponibles sont eux mis en évidence en blanc.

Première page présentant les informations d'un objet («resolver» Simbad). (Figure 16)

E.2.Les alias

Les alias sont important en ce qui concerne la recherche pour les astronomes. Car comme indiqué dans la première partie, un objet peut avoir plusieurs identifiants voir plusieurs dizaines pour certains. Ces identifiants étant utilisés dans de nombreux documents, il faut pouvoir d'un seul regard, voir tout les identifiants liés à un objet. L'utilisateur peut d'ores et déjà retrouver un objet par un de ses identifiants, et il peut à présent dans cette vue voir les identifiants associés à ce dernier.

L'affichage des identifiants se fait de manière asynchrone car la recherche de ces derniers peut être plus longue que le simple affichage des informations sur l'objet qui, elle, s'exécute en une seule et unique requête. Pour récupérer les identifiants secondaires, le système doit aller chercher dans la table «Identifiers» et pour chaque «idfromobject» doit récupérer les valeurs liées. C'est ainsi qu'il doit parcourir toute la base de données, qui bien évidement est indexée pour ce champ. L'affichage des identifiants se fait donc une fois ceux-ci totalement récupérés, en attendant un message informatif indiquant à l'utilisateur que sa requête est en attente de traitement.

Ceci a posé un autre problème qui sera résolu par la suite, c'est celui de l'accès concurrent à la base de données. Le soucis était que lors de la recherche des alias, si la personne indiquait retirer un objet de sa liste de favoris via le bouton «switch» dans la barre de navigation, alors sa demande n'est pas prise en considération. Une solution temporaire est de n'afficher le «switch» que lorsque l'accès à la base de données est libre. Une solution existe pour SQLite permettant l'accès concurrent à la base de données mais cette dernière ne semble pas la solution la plus optimale, du moins pour la version actuelle de SQLite.

Les identifiants secondaires de l'objet «M13» chargés de manière asynchrone. (Figure 17)

E.3.Les notes

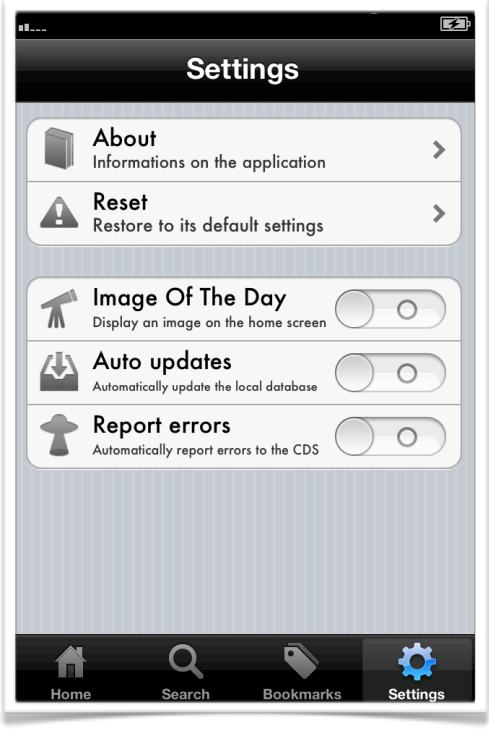
Si l'objet est dans les favoris, la page correspondant à l'ajout d'une note s'affiche, sinon cette page est absente. Ceci permet d'éviter de créer des notes, d'occuper de la mémoire donc, qui risqueraient d'être perdue par la suite par l'utilisateur s'il n'a pas ajouté cet objet en favoris. Car nous sommes parti du principe qu'un utilisateur saisissant une note sur un objet, veux pouvoir y accéder à postériori.

La note se constitue d'un champ de texte classique, une fois sélectionné le clavier apparait, si la personne change de vue en glissant son doit de droite à gauche ou inversement, le clavier se cache. Dès que la vue de la note disparait, le contenu de la note est automatiquement sauvegardé dans la base de données, seulement si le contenu de cette dernière change par rapport au contenu original présent lors du chargement de la note.

Note avec son champ de saisie pour un objet. (Figure 18)

F.La configuration

Ceci est le dernier onglet de l'application, il permet toute la configuration de l'application. Pour le moment l'application comprends trois types de configurations.



L'utilisateur peut tout d'abord demander l'affichage de l'image du jour sur la page d'accueil. Cette option peut intéresser les astronomes amateurs, et propose parfois des images de qualités, issues des serveurs du CDS.

Il est ensuite possible d'activer les mises à jour automatiques de la base de données, nécessitant une connexion à l'internet. Cette partie sera abordée dans la section suivante de ce rapport.

Enfin la personne peut décider de faire remonter les erreurs d'exécution de l'application. Autrement dit à chaque fois que l'application se fige ou stoppe brutalement, à son redémarrage, si l'option a été activée, alors un rapport d'erreur sera envoyé aux serveurs du CDS pour une prochaine correction logicielle si nécessaire.

Sur cette même page l'utilisateur peut consulter la page d'informations de l'application, qui décrit tous les aspects légaux de cette dernière. Enfin la personne peut décider de réinitialiser l'application. Ainsi la base de données et les paramètres seront ré-initialisés comme au premier lancement de l'application.

G.Version universelle

Un des points importants soulevé lors des discutions préparatoire était à propos du support de plusieurs terminaux mobiles, et notamment le support d'une même version pour l'iPhone et pour l'iPad. L'application développée est donc totalement universelle, c'est à dire que pour un seul fichier logiciel, deux versions seront supportées.

En effet la gestion de la taille des éléments de l'interface utilisateur et la façons dont ils sont centrés sur l'écran dépends de la taille de ce dernier, et non de sa résolution, ce qui permet même pour un iPad 3 qui dispose d'une résolution quatre fois plus importante qu'un iPad 2, d'avoir le même aspect visuel.

L'avantage d'une version universelle, est également dans le fait que celle-ci est contenu dans un seul code source, avec une vue pour l'iPhone et une vue pour l'iPad. Ce qui signifie un maintient du code plus simple à gérer. Et d'un aspect commercial, proposer au téléchargement une application qui fonctionne a la fois sur iPhone et iPad est un grand plus, à l'heure ou les tablettes d'Apple cohabitent de plus en plus avec les appareils mobiles d'Apple au sein des foyers du monde entier.

3.3.Le moteur de mise à jour

A.Fonctionnement

Le moteur de mise à jour est un élément important de l'application, au même titre que le moteur de recherche. Car étant donné que l'application embarque une base de données locale, il est important de la maintenir à jour afin de garantir les informations les plus à jour pour l'utilisateur.

La mise à jour automatique de la base de données est soumise à activation de cette fonctionnalité par l'utilisateur. C'est lui qui dans la page de configuration active ou désactive l'option. Ceci afin de préserver sa connexion internet, surtout s'il utilise le plus souvent son forfait internet mobile (EDGE/3G). Car la mise à jour de la base de données peut selon la taille de cette dernière, être plutôt longue et peut parfois être lourde en terme de ressources système.

Le système pour se faire va, dans le cas de l'activation de cette option, requêter les serveurs du CDS via un service développé par André Schaaff. Ce service est accessible via une URL à laquelle l'application fournie une date, le service en ligne renvois pour cette date les informations différentielles de cette journée. Ces

informations sont retournées à l'application dans un fichier XML. Ce fichier XML est traité et les informations contenus sont appliquées à la base de données.

Afin d'éviter que la base de données ne soit corrompus dans le cas où l'utilisateur stoppe l'application ou si un soucis réseaux le coupe de l'internet, le système de mise à jour effectue cette activité dans un «thread» (unité de calcul) secondaire afin de ne pas bloquer l'interface utilisateur. Mais également en tâche de fond via les API fournies par Apple dans iOS afin de ne pas corrompre la base de données en cours de modification.

B.Améliorations envisageables

Afin d'accélérer les mises à jour, et de faciliter l'application de la mise à jour, il est possible, au lieu d'utiliser des fichiers XML, d'utiliser des fichiers «SQLite» directement. Car actuellement le système traite le XML, en extrait les informations, et les appliques à la base de donnée SQLite via des requêtes.

Le but serait d'utiliser une fonctionnalité de SQLite qui permet de fusionner deux bases de données. Ainsi l'application de cette mise à jour serait accélérée de part le fait que ce soit SQLite qui se charge en natif de cette fusion. Mais cette dernière serait également plus simple étant donné qu'il suffirait de déléguer cette action à une classe SQLite déjà développée, et déjà optimisée au maximum.

Le fait de fusionner deux bases de données SQLite reviendrais à un simple traitement de fichier par ce dernier, action qui serait effectuée de manière quasiment instantanée étant donné l'utilisation de mémoire de type «flash» dans les terminaux mobiles d'Apple. Ceci réduirait également les risques de corruption en cas de coupure du réseau ou de l'application elle-même, puisque la base de données ne serait fusionnée avec la base de donnée existante qu'une fois celle-ci téléchargée complètement sur l'appareil de l'utilisateur. Il serait donc possible de la récupérer via le cache de l'application, afin qu'en cas de plantage, ne pas à avoir à la re-télécharger complètement cette dernière.

4.L'«IVOA Interop»

4.1.Présentation

L'«IVOA Interop» fait références aux réunions internationales organisées par les Observatoires Virtuels mondiaux, dans le cadre de l'«IVOA» qui est, rappelons-le, l'«International Virtual Observatory Alliance» autrement dit «L'Alliance Internationale des Observatoires Virtuels». Le terme «Interop» signifie «Interopability», en français «Interopérabilité».

Ce sont des réunions qui ont lieu deux fois par année, généralement en début et en fin d'année. Ces réunions réunissent chaque fois des centaines d'astronomes et informaticiens qui viennent présenter le travail qu'ils ont effectués dans le cadre du VO («Virtual Observatory»). Ces réunions sont organisées à chaque fois dans des pays différents, les équipes chargées des observatoires virtuels locaux se chargeant d'organiser ses rencontres.

C'est au cours de la dernière rencontre, qui a eu lieu à Urbana Champaign, une ville américaine près de Chicago, que notre application iOS et Android a été présentée à la communauté internationale présente. Ce fut l'occasion, durant les quelques jours précédant la présentation de cette dernière par André Schaaff, d'améliorer et de corriger certains points de nos applications afin de présenter une version la plus correcte possible.

4.2.Corrections et améliorations logicielles

Durant ces jours, quelques mises à jour logicielles de correction de bogues ont été apportées à l'application. La plus importante concerne l'amélioration de l'affichage des résultats d'une requête.

En effet, une fois que l'utilisateur a sélectionné un objet dont il souhaite obtenir les informations, bien que l'affichage des résultats fut instantané, l'affichage du résultat en lui-même fut long. En effet, le requêtage et l'affichage de la liste des identifiants secondaires de l'objet demandais beaucoup de temps.

Ce dernier devait effectuer un grand nombre de requêtes dans la base de données interne pour récupérer les noms des identifiants secondaires. Pour chacun d'eux, il devait récupérer dans la table des objets, le nom associé à un identifiant principal pour chacun des «resolvers». Ce qui provoquait de très forts ralentissement s étant donné les milliers d'identifiants secondaires qu'elle contient.

Pour remédier à ce soucis, la recherche dans la base de données en elle-même n'a pas été modifiée, mais cette dernière s'exécute dans un «thread» secondaire et ne bloque donc plus l'affichage de l'utilisateur. Pendant la recherche, dans la liste qui doit contenir les identifiants secondaires, s'affiche un message indiquant à l'utilisateur

d'attendre la fin de la recherche. Une fois les objets récupérés, la page contenant les identifiants secondaires et mise à jour dynamiquement afin de ne pas interpeller outre mesure l'utilisateur.

4.3.Améliorations graphiques

Durant cette période, André Schaaff avait besoin de générer des captures d'écrans à intégrer à ses diapositives. Bien que mon collègue travaillant sur la version Android et moi-même ayons travaillé en collaboration et s'étant accordés sur l'interface utilisateur, certains soucis esthétiques étaient encore à déplorer.

Les deux versions, iOS et son Android, ont donc été revus légèrement dans la mise en place des éléments sur les différentes vues. Ceci afin de présenter deux applications les plus proches possibles en terme d'expérience utilisateur, tout en conservant les aspects propres aux deux plateformes.

5.Le développement HTML5

1.Introduction à HTML5

A la suite de développement de l'application en code natif pour la plateforme iOS d'Apple, nous avons développé une application web en collaboration tous les deux. Cette application est une application développée en utilisant les techniques de programmation web actuelles, et des techniques plus récentes non encore standardisées.

L'application en elle-même est développée en utilisant HTML5. HTML5 est la cinquième version du langage de programmation web HTML («HyperText Markup Language»). Cette version n'est pas encore standardisée, elle le sera courant 2014, la clôture de l'ajout des fonctionnalités de cette dernière a été définie à mai 2012. Malgré le fait que le langage ne soit pas encore standardisé, tout les navigateurs modernes le prennent déjà en considération dans leurs moteurs de rendus, avec plus ou moins de succès.

Car HTML5 est une avancée majeure pour l'internet, le HTML n'ayant pas évolué de manière significative depuis le début des années 2000, le web a lui par contre évolué de

manière profonde. Les contenus multimédias, les sites dynamiques et les services en ligne de plus en plus complet ont pris un essor considérable sur la scène mondiale. Les utilisations du web ont beaucoup évoluées, et il était temps de proposer un langage qui prennent en compte ses changements importants de fond.

C'est ainsi que le HTML5 va pouvoir proposer aux développeurs des fonctions avancées avec la plus grande simplicité, là où auparavant il n'y avait que des librairies et des technologies tierces. Par exemple en HTML5 il est possible via une simple balise «<video>...</video>» d'intégrer une video avec son interface et le tout en haute définition. Ceci est permis sans nécessiter des modules externes, et se fait de manière transparente pour l'utilisateur, les capacités de l'ordinateur sont correctement utilisées pour visualiser la vidéo, à la différence de certains modules actuels.

Exemple d'une animation en HTML5 et CSS3. (Figure 19)

Le HTML5 est donc l'avenir dans ce domaine, avec l'utilisation de CSS3 («Cascading Style Sheet» - mise en page des pages web) celui-ci permet de créer de manière native, sans aucune bibliothèque ni modules externes, de créer des animations en trois dimensions de grande qualité, voir même de faire des jeux vidéos complets, en toute fluidité.

2.Présentation générale du sujet

Le développement de cette application avec les technologies actuelles et futures du web reprends les même caractéristiques que l'application développée nativement. L'application développée sera un service en ligne qui permettra de requêter une base de données d'objets astronomiques via un moteur de recherche. L'application permettra de visualiser le résultat d'une recherche dans une page qui contiendra les trois «resolvers», qui doit pouvoir permettre de créer des notes personnalisées et d'afficher les identifiants secondaires d'un objet. Elle permettra également d'ajouter des objets en favoris.

3.Base de données locale

La base de données sera en local sur le poste de l'utilisateur, mais permettra, comme l'application iOS, d'effectuer des recherches en ligne sur les serveurs du CDS en cas de réponse négative à une requête dans la base interne.

Cette base de données pourra par la suite être également mise à jour si l'utilisateur le souhaite, toujours sur le même schéma, c'est à dire en interrogeant un service web du CDS qui renverra un fichier XML avec les mises à jours différentielles à appliquer à la base. Celle-ci utilise la technologie nommée «WebSQL» qui a été introduite la première fois par le W3C début 2010. «WebSQL» permet d'utiliser SQLite pour créer sur le poste de l'utilisateur, dans le cache, une base de données qui se constitue d'un seul fichier SQLite. Notre base de données ainsi créée occupe sur le poste utilisateur une place d'environ 16 méga-octets.

Il existe cependant une limitation à l'accès au cache du navigateur. Les navigateurs gèrent à leurs manières le cache, mais dans la majorité des cas le navigateur limite la taille du cache, mais demande clairement à l'utilisateur de l'autoriser à stocker plus d'informations. Ceci n'est pas un problème et ne bloquera en rien l'utilisation de l'application en tant que service en ligne disposant d'un mode hors-ligne étant donné que l'utilisateur est informé de ce qui se passe sur son navigateur, et il sait que l'application demande l'accès à plus d'espace de stockage.

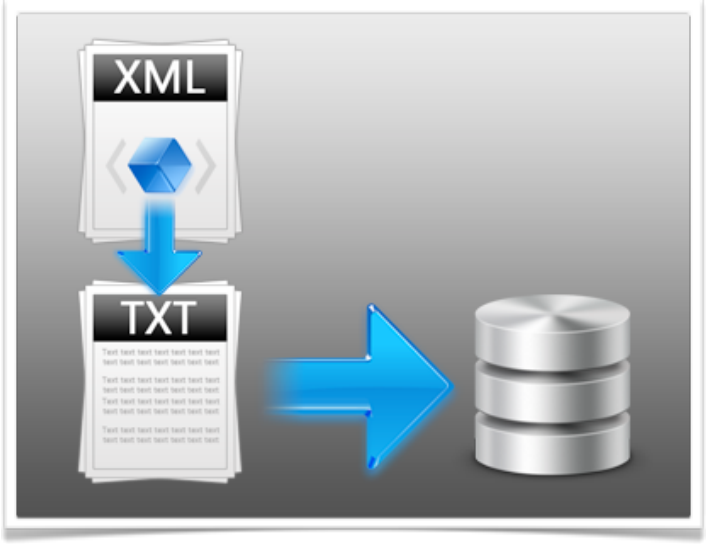


Schéma du fonctionnement de la création de la base de donnée locale. (Figure 20)

La création de la base s'exécute en deux étapes. Tout d'abord un fichier texte contenant toutes les données du fichier XML d'initialisation (celui qui contient tous les objets) est créé et est stocké sur les serveurs du CDS. Ce dernier contient pour chaque ligne, des requêtes SQL permettant de créer et d'insérer les valeurs dans une base de données. A la fin de ce processus, qui s'effectue via des fonctions Javascript, une base de données est créée dans le dossier cache de l'utilisateur. Ainsi même sans accès au réseaux internet, l'application peux continuer à fonctionner étant donné que tout les éléments sont stockés en cache (HTML/CSS3/Javascript/SQLite).

4.Portage de l'application

A.Présentation de PhoneGap

PhoneGap est un «framework» autrement dit un kit de composants logiciels qui sert à créer des fondations d'une ou de toute parties d'une application. Celui-ci permet à partir du code HTML5 de créer des applications natives pour différentes plateformes. Les plateformes les plus populaires prises en charges à l'heure actuelle sont iOS, Android, Windows Phone 7 et BlackBerry.

PhoneGap peut s'exécuter sur tout les systèmes d'exploitations (Mac, Linux et PC) et s'installe simplement comme une bibliothèque dans les principaux outils de développements. Dans mon cas ce fut donc Xcode, pour Android ce fut Eclipse. L'installation de ce «framework» se fait très simplement, et à la création d'une nouvelle application, il suffit d'indiquer vouloir créer une application de type «PhoneGap» et de créer un dossier «www» dans ce dernier et d'y déposer son code.

B.Portage vers PhoneGap

PhoneGap n'est pas un système qui permet de générer du code natif en tant que tel, c'est à dire que le code qu'il génère n'est pas modifiable ni même visible. Ce dernier exécute en faite le code HTML5 qu'il compile à la volée en code natif. Dans son fonctionnement, il se rapproche plus d'une «WebView» (vue qui permet de récupérer et d'afficher du contenu provenant d'une page internet) que d'un compilateur à part entière.

L'intérêt d'utiliser une telle solution viens du fait que cela permet d'être présent sur toutes les plateformes populaires, via les marchés d'applications qui peuvent exister aujourd'hui («App Store» et «Google Play» par exemple). Etre présent en ne créant qu'un seul et unique code, un code de type HTML5 qui en plus de fonctionner en ligne via les navigateurs modernes, est transposable sur les plateformes mobiles. Le portage vers «PhoneGap» se fait sans aucun encombre, l'application se lance normalement, et son utilisation est très fluide.

5.Etude comparative

Tableau comparatif des deux méthodes de programmation. (Figure 21)

A.Temps et finances

Parmi les points les plus avantageux pour le développement d'une application web puis son portage sur différentes plateformes via «PhoneGap», on peux notamment citer un temps de développement et donc un coût financier plus réduit qu'un développement natif. Toujours en prenant compte l'étude et le travail préliminaire nécessaire à tout développement.

B.Esthétique

En terme d'esthétique, un développement web permet de créer une application sur différentes plateformes en ayant une strictement la même interface. Etant donné que l'application est développée en utilisant «CSS3» il est possible de faire des interfaces avancées et de rendre l'expérience utilisateur identique sur toutes les plateformes.

Néanmoins l'application ne dispose plus que d'une seule identité, unique et calquée sur les plateformes. Le développement natif d'une application en utilisant les outils disponibles par défaut permet de créer une application dont le thème est en corrélation avec le thème de la plateforme qui exécute le logiciel. Il ne faut pas

négliger l'identité graphique d'une plateforme, sans pour autant créer des expériences utilisateurs totalement différentes. Il est possible, et le développement de l'application iOS et Android l'a démontré, de créer la même application avec la même structure mais avec deux expériences utilisateurs différentes, tout en conservant l'esprit visuel propre aux deux plateformes.

C.Compatibilité

Développer une application web permet de facto de pouvoir toucher le plus grand nombre d'utilisateurs possible. L'application HTML5 étant à terme compatible avec tout les navigateurs, qui intègrent déjà HTML5 à leurs moteurs. D'ici 2014 tout les navigateurs gèreront la dernière version de HTML.

Le point négatif de l'utilisation de «PhoneGap» réside dans l'utilisation même de «WebSQL» pour la partie gestion de base de données. En effet «WebSQL» est requis par «PhoneGap» et est inscrit dans ses API à utiliser pour le stockage local mais cette technologie n'est plus supportée par le W3C depuis novembre 2010. La dernière version de «PhoneGap» date de mai 2012. Autrement dit, nous utilisons une technologie qui à terme va disparaître car elle n'est plus supportée par le W3C.

Cela se traduit aujourd'hui par une application qui fonctionne parfaitement sur les terminaux via «PhoneGap» mais qui ne fonctionne que sur les navigateurs utilisant comme moteur de rendu «WebKit» comme Apple Safari et Google Chrome. Ce qui a pour effet de rendre caduque la version web de ce service car étant donné que «WebSQL» n'est plus supporté, on peut se douter que par la suite les navigateurs vont cesser de le prendre en charge.

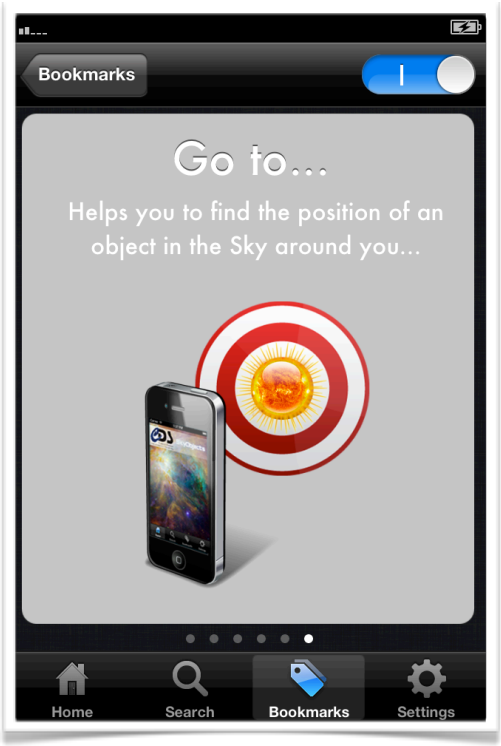
D.Puissance

Le dernier point noir du développement via l'utilisation de «PhoneGap» est l'accès aux composants du téléphone. «PhoneGap» propose une dizaine d'«API» qui permettent d'accéder au gyroscope, GPS, système de fichier ou encore caméra photo. Ces «API» sont développées en Javascript, ce qui implique l'ajout d'une couche (Javascript) avant que le «framework» n'accède finalement au composant via du code qui lui est natif. Cela va donc ralentir l'utilisation des composants, et nous rends dépendant de «PhoneGap» en ce qui concerne l'évolution et le support des futures versions de ce dernier.

Quand il s'agira d'ajouter à l'application le pointage, qui requiert l'accès rapide à divers composants du téléphone (voir section suivante), l'utilisation de ces API va ralentir l'application au meilleur des cas, et va ne pas pouvoir être implémenté au pire des cas. En effet il va donc falloir développer une bibliothèque complète en Javascript pour fonctionner avec «PhoneGap» pour effectuer les calculs de positionnement via le flux de données transmis par les API de «PhoneGap» qui lui même accède aux composants via sa sur-couche logiciel en code natif.

IV.Perspectives

1.Le pointage d'objets



Le pointage d'objet est une des fonctionnalités qui va faire de «SkyObjects» une application plus grande public qu'elle ne l'est actuellement. Ceci va permettre d'ajouter une plus grande interactivité entre l'application, son contenu et l'environnement de l'utilisateur.

Le pointage trouve sa place au sein de l'interface dans la vue réservée à l'affichage des informations d'un objet, une fois ce dernier sélectionné à partir de l'affichage des résultats suite à une recherche d'objet. Cette vue affichant les informations comporte donc plusieurs pages symbolisées par des points en bas de page.

Pour le moment le pointage n'est donc pas actif car non encore développé au moment du stage. Un tel dispositif ne pouvant pas être développé en si peu de temps, en plus de la conception, des phases de réflexion et de correction de l'application première («SkyObjects») qui elle n'existait pas auparavant. Mais le pointage dispose de sa propre vue sur laquelle s'affiche, pour le moment, une image et un texte explicitant ce que cette vue proposera à l'avenir.

Un peu à l'image de ce que font d'autres applications de ce type comme par exemple «SkyView», notre application permettra d'afficher sur l'écran de l'utilisateur une indication, par une flèche par exemple, de la zone qu'il doit pointer avec son téléphone pour y trouver l'objet demandé.

Ceci fait appel à différents composants de l'appareil tel que l'accéléromètre, le gyroscope et le GPS. Tout ces outils sont primordiaux afin d'afficher en réalité augmentée sur l'écran, la position vers laquelle pointer pour voir l'objet. Le pointage implique l'utilisation de nombre de fonctions présentes dans application qui a été développée en Java par le CDS pour les terminaux Android, mais qui n'existe pas encore en Objective-C pour les terminaux sous iOS.

2.L’ajout de catalogues

Actuellement, la base de donnée locale contient environs 7000 objets tous issus d’un catalogue nommé «NGC». Ce catalogue comprends des objets astronomique parmi les plus utilisés et les plus demandés. C’est la raison pour laquelle ce catalogue est inclus par défaut en local.

Néanmoins, il est possible que des utilisateurs plus avancés ou des astronomes, aient besoins d’autres catalogues plus spécifiques. C’est pourquoi par la suite, l’application pourrait proposer ceci dans la page de configuration. Il pourrait y avoir un menu particulier qui contiendrais un certain nombre de catalogues, l'utilisateur n’aurais qu’à les sélectionner pour les ajouter à sa base de données locale.

L’ombre dans ce tableau reste la quantité de données à transférer et à organiser dans la base de données interne de l’appareil. En effet la base de données «NGC» actuelle pèse plus de 17 méga-octets. Cette base de donnée à été générée à partir d’un fichier XML, génération qui a durée près de deux heures, entre le traitement du fichier XML et l’ajout des informations recueillies dans la base de données SQLite.

Proposer à l'utilisateur l’ajout de nouveaux catalogues revient à lui demander de télécharger une base de données de plusieurs dizaines de méga-octets via WiFi mais également via réseaux mobile (EDGE/3G). Autant le téléchargement en WiFi de cette base de données peut être rapide, autant sur un réseaux mobile ce téléchargement peut être long et couteux sur le forfait de l'utilisateur. Sans compter que l’application, une fois cette base téléchargée, doit pouvoir la fusionner à la base existante. Il serait néanmoins possible de conserver plusieurs base de données mais le désavantage serait une extrême lenteur dans la recherche d’objet. Il faudrait requêter dans toutes les bases, c’est à dire l’ouvrir et la fermer plusieurs fois en très peu de temps.

3.Le mode éducation

L’application actuellement est plutôt réservée à un public d’initiés et d’astronomes. Son contenu multi-médias et interactif étant très limité, cette application ne peut pas intéresser tout les utilisateur classiques, qu’ils soient novices en astronomie, ou juste curieux.

En effet les champs de résultats d’un objet correspondent aux libellés dans champs dans la base de données, créant des champs nommé «R», «oid» ou bien encore «V» dont nous ne pouvons connaître la signification par soi-même, cette dernière n’étant pas du tout explicite.

Le mode éducatif pallierait donc au manque d'accessibilité de l’application pour un public de novices. Il permettrait également plus de contenus multi-médias dans l’application, que ce soit des images et pourquoi pas des vidéos. Ceci nécessitant

cependant une connexion internet, le but premier de créer une application utilisable en mode hors-ligne, s'éloigne. Mais s'il s'éloigne, c'est pour faire profiter de son contenu à un public bien plus large.

Un ajout supplémentaire à ce mode grand public, serait la possibilité de rechercher des objets astronomiques connus en français. Car actuellement, il est possible d'effectuer une recherche par identifiant, et par nom d'objet également. Cependant, ces noms sont en anglais. Ainsi une recherche «Andromeda» donnera le résultat correspondant à la galaxie d'Andromède, alors qu'une recherche «Andromède» avec l'accent ne donnera aucun résultat. Il serait alors possible de limiter la taille de la base de données aux objets ayant été définis comme méritant un intérêt de la part de l'utilisateur novice.

Il est à noter que la base de données actuelle comprend déjà un champs destiné à traduire le nom de l'objet. Néanmoins ces derniers sont vides car ils doivent être traduits un par un.

V.Diagramme de Gantt

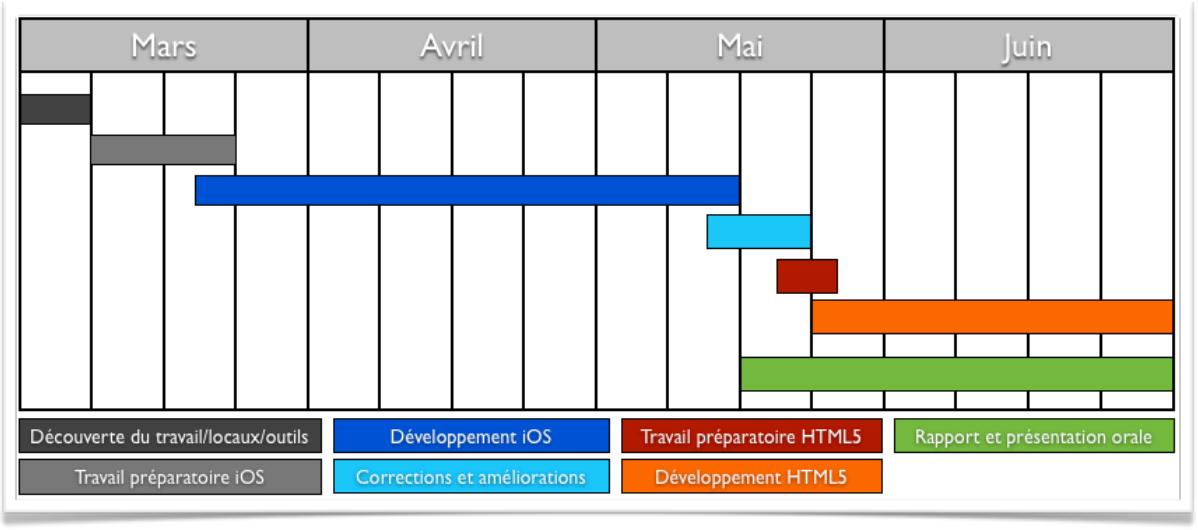


Diagramme de Gantt du développement du stage. (Figure 22)

Conclusion

L'application «SkyObject» développée est une application native iOS mais également une application web qui mérite entièrement sa place au sein des outils développés par le CDS. Cette application s'inscrit complètement dans le cadre du travail du CDS pour fournir des outils de plus en plus proche des utilisateurs et de plus en plus modernes. Il s'inscrit également dans la politique du CDS pour la création d'outils pour la communauté internationale astronomique via l'Observatoire Virtuel.

L'application, toute récente, a devant elle un avenir intéréssant. Notamment via l'implémentation du pointage qui permettra à l'utilisateur de pointer dans le ciel un objet via l'ajout des différents catalogues à la base de données interne pour plus de résultats, même sans aucune connexion internet.

Mais l'application s'ouvrira également à un plus grand nombre de personnes via l'ajout d'un mode éducation/grand public. Ce mode permettra plus d'interactivité entre l'application et l'utilisateur pour lui fournir plus de contenus multi-médias. Ce mode permettra une plus grande accessibilité dans les termes de recherche en permettant une recherche d'objet en français.

Cette expérience fut très enrichissante, aussi bien sur le plan des connaissances, que le plans culturel et que sur le plan humain. Le développement de la version mobile sur iOS m'a permis d'approfondir encore plus mes connaissances du développement sur les plateformes mobiles d'Apple. Ce développement m'a également confirmé que ce domaine de l'informatique est celui qui m'attire le plus et qui présente un grand intérêt et de grandes opportunités actuelles et futures.

J'ai pu découvrir un monde qui m'était encore très peu connus qu'est celui de l'astronomie, en travaillant au sein d'une équipe dynamique et qui ne craint pas de faire face à des challenges pour faire avancer les choses et aller de l'avant. Une équipe qui prends très au sérieux son action au sein de la communauté internationale pour proposer toujours plus d'outils, au plus grand nombre. Pour proposer des outils avec une accessibilité toujours plus fine pour élargir son public et partager la connaissance, sans oublier les professionnels du domaine dont le travail du CDS et de l'Observatoire est considéré comme extrêmement important et utile.

Ce stage a également été une grande opportunité pour moi de travailler pour soutenir l'effort fournit par l'Observatoire (mondialement reconnu dans la communauté astronomique) et le CDS, au sein d'une unité mixte du CNRS et de l'Université de Strasbourg, auquel est rattaché l'Observatoire.

Bibliographie

StackOverflow: *http://stackoverflow.com*
iPhoneDevDesk: *http://www.iphonedevsdk.com*
RayWenderlich: *http://www.raywenderlich.com*
CocoaWithLove: *http://cocoawithlove.com*
W3C: *http://dev.w3.org/html5/html-author/*
W3C School: *http://www.w3schools.com/html5/html5_intro.asp*
InstaCSS: *http://www.instacss.com*
AlsaCreation: *http://www.alsacreations.com*
Présentation d'André Schaaff à l'«IVOA» à Urbana Champaign:
http://www.ivoa.net/internal/IVOA/InterOpMay2012Applications/IVOA2012-Urbana-Application-CDS-AS.pdf

Figures