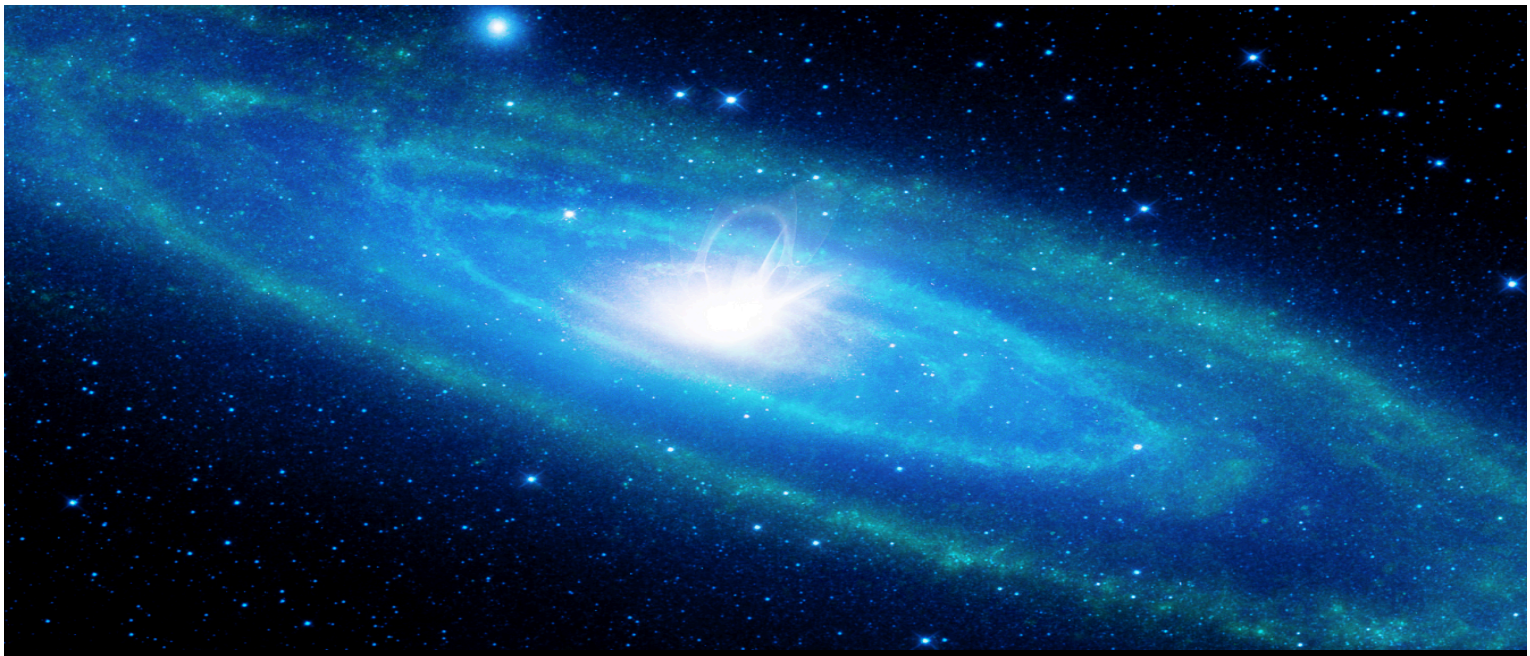


UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

Développement d'outils et d'interfaces utilisateurs pour les services web du CDS

Rapport de stage ST40 - P2015

THOMAS Rolling

Département Génie informatique

Observatoire astronomique de Strasbourg

11 rue de l'Université
67000 Strasbourg

Tuteur en entreprise
ANDRE Schaaff

Suiveur UTBM
CHRISTIAN Fischer

Remerciement

Mes premiers remerciements vont tout naturellement à mon tuteur en entreprise M. André SCHAAFF, ingénieur de recherche au Centre de Données astronomiques de Strasbourg, pour m'avoir encadré pendant ces vingt deux semaines de stage et pour sa grande disponibilité malgré ses responsabilités, pour ses encouragements et précieux conseils qui m'ont permis de mener à bien ce travail ainsi que pour les corrections minutieuses qu'il a portées à ce rapport.

Je tiens à remercier chaleureusement M. Gilles LANDAIS, ingénieur d'étude au Centre de Données astronomiques de Strasbourg, pour avoir également encadré mon stage, pour toutes ses explications et pour m'avoir aidé de nombreuses fois.

Mes remerciements s'adressent également aux membres de l'Observatoire et plus particulièrement, M. André SCHAAFF, M. Gilles LANDAIS, M. Laurent MICHEL, et M. Thomas BOCH pour leur accueil chaleureux et leur réponse à mes diverses interrogations.

J'exprime également ma reconnaissance envers Hervé Wozniak et Françoise Genova, respectivement directeur de l'observatoire et directrice du CDS qui m'ont intégré au sein de l'organisation.

Je remercie aussi mes deux collègues de bureau M. Sébastien DERRIERE et Mme. Anne Camille SIMON pour la très bonne ambiance.

Introduction

Dans le cadre de ma formation d'ingénieur en informatique à l'Université de Technologie de Belfort-Montbéliard (UTBM), j'ai effectué mon stage au sein de l'Observatoire Astronomique de Strasbourg. Au cours d'une période de six mois, j'ai eu l'occasion d'acquérir de nouvelles compétences professionnelles tout en approfondissant celles acquises jusqu'alors.

Mon choix s'est porté sur ce stage pour la qualité du sujet proposé ainsi que la thématique liée à l'espace. Souhaitant rejoindre la filière Ingénierie des Logiciels et de la Connaissance (ILC) je voulais effectuer une première approche professionnelle dans ce domaine. Etant amateur d'astronomie, l'offre de stage m'a tout de suite interpellé.

Dans la première partie de ce rapport, je présenterai l'Observatoire Astronomique de Strasbourg ainsi que sa relation avec le Centre de donnée Astronomique de Strasbourg (CDS. Après cela, j'aborderai le déroulement du stage et l'organisation des projets avant de présenter plus précisément les aspects techniques et les résultats obtenus.

Table des matières

1. Présentation de l'Observatoire astronomique

1.1 Présentation de l'observatoire

1.1.1 Les lieux

Ce haut lieu scientifique prend ses origines dans la seconde moitié du XIXème siècle, à la fin de la guerre franco-prussienne. L'Allemagne victorieuse de ce combat face aux Français mal préparés, elle récupère les territoires de l'Alsace-Lorraine ainsi que la ville de Metz, conquis par les rois de France durant les trois siècles précédents. Ce rattachement à l'Allemagne est mal vécu par les Strasbourgeois encore traumatisé. Au même titre que Metz, Strasbourg se transforme et retrouve la prospérité grâce à la volonté politique du gouvernement. Ce dernier souhaite faire de la ville une vitrine de l'empire et du savoir-faire allemand. C'est dans son vaste plan d'urbanisation de Strasbourg que l'empereur Guillaume Ier décide d'y installer une université avec un jardin botanique et un observatoire astronomique.

Vu de la Grande Coupole.

L'édifice construit entre 1876 et 1880 est inauguré en 1881. Il est composé d'une Grande Coupole, d'un bâtiment des salles méridiennes avec deux plus petites coupoles et un bâtiment accueillant des bureaux et des résidences. Ils sont tous les trois reliés par un couloir couvert en forme de « Y ». La Grande Coupole en fer de plus de 9 mètres de diamètre abrite le Grand Réfracteur, une lunette de 7 mètres de long, la plus grande d'Europe à l'époque, aujourd'hui la troisième plus grande de France. Un siècle plus tard, l'Observatoire se dote d'un planétarium en 1981, un nouveau lieu de loisir, de

diffusion de la connaissance et un outil pédagogiquepuissant qui permet de comprendre l'architecture et l'évolution de l'Univers. C'est également un lieu d'exposition et de découverte des plus beaux instruments du patrimoine de l'Observatoire.

1.1.2 Les activités de recherche

La vocation initiale des activités de recherche concernait l'astronomie de position et l'observation des comètes, de météorites et d'étoiles variables. Par la suite les activités se sont étendues à la photométrie de nébuleuses et à l'observation d'étoiles doubles. Durant les années 1970, l'Observatoire développe l'archivage informatique, qui contribuera à la naissance du centre de données stellaires et qui deviendra en 1972 le Centre de Données astronomiques de Strasbourg (CDS). Actuellement les activités de recherche s'articulent autour de trois grandes équipes scientifiques : « Astrophysique des hautes énergies » étudie la physique des astres compacts en fin d'évolution, accrétion, éjection et phénomènes magnétohydrodynamiques. « Galaxies » se charge des populations stellaires, propriétés chimiques et dynamiques de la Galaxie et des galaxies proches, milieu intergalactique, grandes structures, et dynamique gravitationnelle. Enfin le CDS s'occupe des méthodes de gestion de l'information et de l'exploitation scientifique des grands relevés. Ce dernier est labellisé depuis 2008 "Très Grande Infrastructure de Recherche" (TGIR) par le Ministère de l'Enseignement Supérieur et de la Recherche.

1.1.3 Un rayonnement international

L'Observatoire est également membre du consortium Survey Science Center de la mission XMM-Newton. Cette dernière a pour objectif l'étude des rayons-X afin de mieux comprendre le fonctionnement de l'Univers et des événements violents qui s'y déroulent comme la vie des trous noirs ou les explosions d'étoiles. Mais c'est probablement le CDS qui contribue le plus à la renommée internationale de

l'établissement. En effet il participe activement au développement de l'ASOV et de l'IVOA

, dont nous parlerons ci-dessous, et est à l'initiative de services fortement utilisés par les astronomes et amateurs du monde entier.

1.1.4 Les Observatoires Virtuels

L'Observatoire Virtuel français est une collection d'archives de données interactives et projets similaires d'autres pays comme la Chine, l'Australie, le Canada, le Japon, l'Inde, la Russie et la Corée, se sont associés en 2002 afin de coordonner leurs efforts au sein de l'alliance internationale IVOA avec la mission suivante : « faciliter la coordination et les collaborations internationales nécessaires au développement et au déploiement d'outils, de systèmes et de structures rendant possible l'utilisation des archives astronomiques comme s'il s'agissait d'un Observatoire Virtuel unique ».
L'IVOA regroupe ainsi 17 projets nationaux. Son action principale consiste à favoriser l'établissement de standards à travers des groupes de travaux qui suivent un fonctionnement identique à celui du World Wide Web Consortium (document de travail, proposition de recommandation, et finalement recommandation). Deux conférences par an permettent de coordonner les actions des différents groupes (IVOA Interopconferences).

Les groupes de travail de l'IVOA couvrent l'ensemble des besoins de standardisation de la discipline. Des données jusqu'aux applications, ils définissent les formatages des données, des métadonnées, les langages d'interrogation, les protocoles d'échange, etc. Le tableau décrivant les neuf groupes de travail, leur champ de compétences respectif ainsi que l'état des standards proposés est présenté dans l'annexe A.
L'IVOA requiert deux mises en œuvre indépendantes des standards avant de les valider. Le Centre de Données de Strasbourg, fortement impliqué dans l'Observatoire Virtuel participe à de nombreux groupes de travail. Le logiciel Aladin a été régulièrement

utilisé pour tester et valider certains de ces standards. De ce fait, Aladin est devenu l'exemple type de «l'outil compatible OV », et un portail reconnu d'accès à l'Observatoire Virtuel.

Il s'agit donc de construire des standards d'échange, des outils d'interrogation, des systèmes d'extraction de l'information, de manière à globaliser les données et plus généralement l'information en astronomie, au niveau international. Dans ce cadre, les principaux centres de données astronomiques internationaux s'efforcent de donner davantage de visibilité à leurs bases de données et développent des descriptions détaillées de leur contenu. Ceci est rendu possible grâce à un immense effort de standardisation aussi bien des données que des méthodes et outils utilisés par les astronomes.

De nombreuses bases de données sont déjà interconnectées dans la communauté astronomique et de multiples applications client/serveur diffusent des données hétérogènes (articles de journaux, catalogues d'observations, images, spectres, etc.). Le Centre de Données astronomiques de Strasbourg (CDS) y prend une part importante au travers de ses services Aladin, Simbad ou VizieR qui sont utilisés par les astronomes du monde entier. Nous les présenterons dans la partie qui suit.

1.1.5 Les services du CDS

1.1.5.1 Vizier

VizieR est une base de données qui regroupe environ 10 000 catalogues d'objets astronomiques. Ces catalogues sont en fait des tables relevées durant des missions d'observation et ajoutées à VizieR par les documentalistes. Ce service permet à un utilisateur d'accéder de manière homogène à des données hétérogènes de catalogues, de les croiser et de les exporter sous différents peuvent porter sur de multiples critères comme la longueur d'onde avec laquelle les objets ont formats. Les interrogations sur la base de catalogues étés observés ou encore le nom de la mission correspondante.

1.1.5.2 Aladin

Aladin est un logiciel d'astronomie développé par le CDS pour la communauté internationale. Véritable atlas interactif du ciel, il permet aux scientifiques de localiser, accéder, comparer et analyser les données images et catalogues issues de la plupart des serveurs astronomiques (images d'archives, relevés du ciel, catalogues, bases bibliographiques). Ses caractéristiques techniques lui ont permis de devenir l'un des outils clés de la discipline aussi bien pour la préparation de missions d'observations (télescopes spatiaux Hubble et James Webb), que pour la visualisation de données des plus importants centres de données astronomiques européens (ESO, ESAC, CDS), américains (MAST/NASA, NED/NASA), canadiens (CADC)...Il est un véritable intégrateur de données hétérogènes issues des sites et des projets répartis sur toute la planète. Le logiciel entièrement écrit en Java est disponible sur sous forme d'applet sur le site du CDS.

1.1.5.3 Simbad

Simbad est la base de données de référence pourla nomenclature et la bibliographie des objetsastronomiques. C'est un service qui permet auxastronomes, à partir du nom d'un objet d'accéderfacilement aux informations comme lescoordonnées, mesures physiques, données bibliographiques, etc. de plus de 7 millions références astronomiques en dehors du système solaire. Il est aussi possible d'interroger la base en utilisant les coordonnées ou les références d'un objet. Simbad dispose également d'un résolveur de noms qui permet de connaître toutes les nomenclatures de l'objet considéré.

1.2 L'équipe du CDS

Le Centre de Données astronomiques de Strasbourg est l'une des principales composantes de l'Observatoire. Il offre un accès à des données astronomiques à forte valeur ajoutée au travers de services en lignes et d'applications grâce au travail de ses astronomes, informaticiens et documentalistes. Ils représentent un effectif d'une trentaine de personnes. Ces trois professions sont absolument complémentaires et sont indispensables au bon fonctionnement du centre. En effet, ils forment une chaîne entraînant le processus de collecte et d'enrichissement des données par les documentalistes, de vérification et d'exploitation des données par les astronomes et de développement d'outils et services pour la sauvegarde et l'accès aux données par les informaticiens.

La mission principale de ces derniers consiste à développer et maintenir les services et outils du CDS mais aussi d'assurer la veille technologique. En effet, la technique évolue très rapidement et le CDS tient à identifier les nouveautés prometteuses et à évaluer leur intérêt pour les services en ligne qu'il propose. Il a donc une activité significative de recherche et de développement.
Pour ce faire, les ingénieurs de recherche et les ingénieurs d'étude disposent de nombreux serveurs, de terminaux sous MacOs, Windows 7 et Linux. Afin d'expérimenter le développement mobile, le CDS s'est également doté de Smartphones et de tablettes Android et Apple ainsi que d'un moniteur tactile.

1.3 Description du sujet

Le sujet de stage proposé par le CDS était orienté développement web, avec deux axes de travail principal.

Le premier concerné un analyseur syntaxique pour fichier VOTable développé en JavaScript. Le second était la refonte de l'interface utilisateur de l'outil en ligne VizieR.

L'ensemble du travail devait être effectué en JavaScript où la première partie du stage (analyseur syntaxique) devait être directement implémenté au sein de la nouvelle interface VizieR. Les deux projets sont donc intimement liés.

La première partie du stage fût intégralement encadrée par M. André Schaaff, tandis que la seconde moitié fût également encadrée par M. Gilles Landais, responsable du projet VizieR.

2 Organisation du stage

2.1 Objectifs

2.1.1 Analyseur syntaxique

L'absence de librairie permettant de travailler aisément sur un fichier VOTable en JavaScript a motivé le développement d'un analyseur syntaxique. L'objectif principal était de posséder un outil standard, fiable et simple d'utilisation à proposer à l'ensemble des développeurs pour faciliter leurs travaillent au sein des projets web.

Les objectifs de l'analyseur syntaxique sont les suivants :

- Simple d'utilisation, garnit d'une multitude de méthode pour travailler aisément avec un fichier VOTable (API).
- Gérer des volumes conséquent de données (~100 000 lignes minimum).
- Performant car la librairie est exécutée côté client où les ressources dédiées à l'analyseur syntaxique varient.
- Gérer des fichiers VOTable à contrainte spécifique (encodage en base 64, préfixe, ...).
- Assurer une compatibilité avec les différents navigateurs web existant ainsi que leurs versions antérieurs.

2.1.2 Interface homme machine

L'interface homme machine de VizieR souffre d'un souci d'ergonomie, son utilisation n'est pas des plus aisé pour les non initiée à l'outil. Il est parfois nécessaire d'effectuer plusieurs clics pour accéder au contenu désiré. De plus la version actuelle dispose de deux interfaces d'accueil, une principale, et une seconde accessible via la première (celle-ci étant plus poussé en terme de fonctionnalités).
Les objectifs du projet sont donc les suivants :

- Améliorer l'ergonomie, le contenu doit être accessible en deux clics maximum.

- Respecter l'esprit graphique des outils du CDS, la sobriété.
- Afficher les données issues d'une recherche dynamiquement sur la même page.

2.1.3 Lien entre les deux projets

Une des caractéristique intéressantes du stage est le faite que les deux projets sont étroitement liées. En effet l'interface homme machine intègre l'analyseur syntaxique développé en début de stage.

Cette façon de travailler avait pour but d'améliorer l'analyseur syntaxique, plus particulièrement :

- Vérifier qu'il n'y a aucun dysfonctionnement (bug).
- Solutionner les problèmes.
- Voir comment réagit l'analyseur syntaxique sur le traitement de différents fichiers.
- Ajouter des méthodes à l'analyseur si celle-ci se révèle utile pour l'interface.
- Vérifier que l'analyseur syntaxique était facilement intégrable au sein d'un projet.

2.2 Planning

Les deux projets devaient être réalisé durant les 22 semaines de stages, dès que l'analyseur syntaxique a été partagé auprès d'autres développeurs le développement de l'interface homme machine a débuté. Si durant la seconde partie du projet des lacunes était repérée dans la librairie, celles-ci devaient être corrigé.

Le projet disposait d'une certaine flexibilité, il n'y avait pas de date butoir, or mis finir les deux projets dans le temps imparti par la durée de mon stage.

2.3 Méthodes de travail

Le protocole de développement été identique pour les deux projets à peu de chose près, à savoir :

1. Recherche de solution technologique existante.
2. Développement.
3. Test unitaire en local au fur et à mesure (par le développeur et le suiveur du projet).
4. Phase de test (mise à disposition de l'outil à des testeurs).
5. Mise en production.

Pour l'analyseur syntaxique une version était envoyé chaque semaine à M. Schaaff qui vérifiait l'avancé ainsi que le bon fonctionnement de celui-ci.
Pour l'interface homme machine, le travail était envoyé sur le serveur de production dans un dossier non accessible au visiteur, permettant ainsi de tester le code dans l'environnement final.

Le développement a nécessité des outils liés à l'écosystème JavaScript pour travailler dans des conditions optimal.

- Bracket : Editeur de code développé en JavaScript par Adobe.
- JSLint : Outil d'analyse de code statique.
- JSBeautifier : Outil qui permet l'offuscation et l'indentation du code.
- Rsync : Synchronisation de fichiers avec un serveur.

2.4 Technologies utilisées

2.4.1 JavaScript

Le JavaScript a été crée en 1995 par Brendan Eich, le langage est orienté objet par prototype. Historiquement il s'agit du premier langage de script pour le web, il permet d'exécuter du code côté client, et ainsi d'apporter des améliorations au langage HTML.

Aujourd'hui le langage s'est démocratisé dans des usages variés, il permet notamment de faire des applications pour Smartphone, tablettes, Windows 8, ...

Le Centre de Donnée Astronomique plébiscite souvent ce langage pour les applications web, principalement pour des économies de coût (absence de serveur) et les compétences possédé en interne.

2.4.2 jQuery

jQuery est une bibliothèque créée pour faciliter le développement de code JavaScript. La première version est lancée en janvier 2006 par John Resig. Sa popularité vient du faite qu'il propose une navigation plus aisée dans l'arbre DOM qu'en JavaScript natif.

La bibliothèque fournit un flot de fonctionnalités, comme par exemple :

- Manipulations de feuilles de style CSS (ajout / suppression / modification d'attributs).
- Parcours et modification de l'arbre DOM via les sélecteurs.
- Effets visuels et animations.
- Gestion des évènements.
- Intégration de l'Ajax.

2.4.3 Ajax

L'Ajax (Asynchronous JavaScript And XML) est une technologie qui permet de construire des applications et des sites web dynamiques. Ajax est plus une combinaison de technologie, qu'une technologie à proprement parler, il intègre le JavaScript, CSS, XML, JSON, DOM et le XMLHttpRequest. Son atout majeur est qu'il peut modifier dynamiquement une page web en effectuant un échange client / serveur sans rafraichir la page. Là où le JavaScript natif ne peut pas communiquer directement avec un serveur et où un langage serveur comme PHP doit actualiser la page web pour que ces changements soient inclus sur l'interface client.

Imbrication des différentes technologies présentes dans l'Ajax entres elles :

- CSS / DOM / JavaScript : Permet de modifier dynamiquement l'interface web.
- XMLHttpRequest : Communication avec un serveur web.
- JSON / XML : Format standard utilisé pour le transfert de données entre le serveur et le client via l'objet XMLHttpRequest.

2.4.4 Bootstrap

Boostrap est un framework CSS / JavaScript crée par Mark Otto et Jacob Thornton en 2011 fournissant une collection d'outils pour la création d'application web. Bootstrap fournit une feuille de style CSS qui contient des définitions de base pour tous les composants HTML, ce qui permet de disposer d'une apparence uniforme. Le point fort de bootstrap est qu'il adopte la conception de sites web adaptatifs (responsive design) permettant aux application web de s'adapter dynamiquement à différent support (Smartphone, Tablette, ...). Le framework fournit un ensemble d'éléments graphiques au format standardisé (boutons, icônes, barres de progression, ...) accélérant ainsi le

développement d'interface. Il est compatible avec tous les navigateurs récents et fournit un rendu dégradé sur les navigateurs plus anciens.

3 Développement d'un analyseur syntaxique

3.1 Contexte

Beaucoup d'outils développés par le Centre de Données Astronomique utilisent les VOTables comme standard d'échange de données astronomique. Un analyseur syntaxique est nécessaire pour travailler avec ces données structurées en XML.

Auparavant chaque projet développé un analyseur syntaxique propre à ses besoins.

La volonté du Centre de Données Astronomique était de créer un outil unique (une librairie) qui se chargerait d'effectuer cette tâche.

Les objectifs recherchés sont les suivants :

- Accélérer le développement des divers projets en disposant d'un analyseur syntaxique sur lequel s'appuyé.
- Disposer d'un outil éprouvé sur une multitude de fichier XML analysé.
- Avoir un outil unique pour ne pas effectuer un travail de développement redondant.

Les données présentes dans le fichier XML peuvent être encodé en base 64, les décodés en JavaScript n'est pas une tâche des plus aisés. La majorité des services ne supportent pas la base 64, bien qu'intégré comme une fonctionnalité du standard VOTable. Le développement d'un analyseur syntaxique commun était l'occasion de proposer cette fonctionnalité. Ce fût sans doute l'élément déclencheur du projet.

3.2 Etat de l'art

Actuellement il n'existe aucun analyseur syntaxique complet pour VOTable, uniquement des outils propre à chaque projet dont le champ d'action se limite au besoin du projet lui même. Il semble difficile de s'appuyer sur ces outils, en effet ils sont bien souvent directement intégrés au code des projets au gré des besoins d'interactions avec un fichier VOTable.

Une piste de recherche intéressante fût de trouver des bibliothèques qui effectuaient une conversion du format de données, passant du XML (VOTable) à du JSON. Ce dernier est bien plus pratique pour travailler en JavaScript et nous aurions gagné un temps conséquent dans le développement de l'analyseur syntaxique. En effet nous travaillons directement avec un objet qui intègre toutes les données de la VOTable.

Nous avons testé toutes les librairies qui effectuaient ce genre de conversion, à savoir :

- xml2json (par Henrik Ingo).
- xml2json (par Stefan Goessner).
- x2js (par abdmob).
- xmlToJson (par David Walsh).
- etc, ...

Nous avons décidé de nous appuyez sur x2js, en effet c'est la seule a avoir réussi à convertir plusieurs fichiers VOTable sans encombre. A première vu les performances ont l'air honorable.

3.3 Description du standard VOTable

Le format VOTable est une norme XML pour l'échange de données astronomique représenté par un ensemble de table.
L'objectif de ce standard est de rendre les données interopérable et évolutive. VOTable est un format d'échange de données tabulaires conçu pour un stockage flexible, avec une spécialisation pour les tables astronomiques.
L'interopérabilité est encouragée par l'utilisation de nomes (XML). En effet la structure du XML permet aux applications de lire facilement les données.

Le standard VOTable permet de stocker des données sous différent format :

- TABLEDATA : Données au format XML classique.
- FITS (Flexible Image Transport System) : Format de fichier communément utilisé en astronomie.
- BINARY / BINARY2 : Format binaire, utilisé pour compresser les données et ainsi avoir une VOTable plus légère en terme de taille du fichier.

Structure d'un fichier VOTable :

- VOTable : Hiérarchie des métadonnées + les données ordonnées par tables.
- MetaData : Paramètres + informations + descriptions + liens + champs + groupes.
- Table : Liste des champs + les données de la table.
- TableData : Liste de ligne (<TR><TD>data</TD>...</TR>).
- Row : Liste de cellule (<TD>data</TD>).
- Cell : Contient une donnée primitive, ou une liste de primitives, ou un tableau de primitives.
- Primitive : Type de donnée (integer, float, character, short, long, ...).

Exemple de fichier VOTable :

```
<?xml version="1.0"?>
<VOTABLE version="1.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://www.ivoa.net/xml/VOTable/VOTable/v1.1">
  <COOSYS ID="J2000" equinox="J2000." epoch="J2000." system="eq_FK5"/>
  <RESOURCE name="myFavouriteGalaxies">
    <TABLE name="results">
      <DESCRIPTION>Velocities and Distance estimations</DESCRIPTION>
      <PARAM name="Telescope" datatype="float" ucd="phys.size;instr.tel" unit="m" value="3.6"/>
      <FIELD name="RA"   ID="col1" ucd="pos.eq.ra;meta.main" ref="J2000"datatype="float" width="6" precision="2" unit="deg"/>
      <FIELD name="Dec"  ID="col2" "pos.eq.dec;meta.main" ref="J2000" datatype="float" width="6" precision="2" unit="deg"/>
      <FIELD name="Name" ID="col3" ucd="meta.id;meta.main" datatype="char" arraysize="8*"/>
      <FIELD name="RVel" ID="col4" ucd="src.veloc.hc" datatype="int" width="5" unit="km/s"/>
      <FIELD name="e_RVel" ID="col5" ucd="stat.error;src.veloc.hc" datatype="int" width="3" unit="km/s"/>
      <FIELD name="R" ID="col6" ucd="phys.distance" datatype="float" width="4" precision="1" unit="Mpc">
        <DESCRIPTION>Distance of Galaxy, assuming H=75km/s/Mpc</DESCRIPTION>
      </FIELD>
      <DATA>
        <TABLEDATA>
        <TR>
          <TD>010.68</TD><TD>+41.27</TD><TD>N  224</TD><TD>-297</TD><TD>5</TD><TD>0.7</TD>
        </TR>
        <TR>
          <TD>287.43</TD><TD>-63.85</TD><TD>N 6744</TD><TD>839</TD><TD>6</TD><TD>10.4</TD>
        </TR>
        <TR>
          <TD>023.48</TD><TD>+30.66</TD><TD>N  598</TD><TD>-182</TD><TD>3</TD><TD>0.7</TD>
        </TR>
        </TABLEDATA>
      </DATA>
    </TABLE>
  </RESOURCE>
</VOTABLE>
```

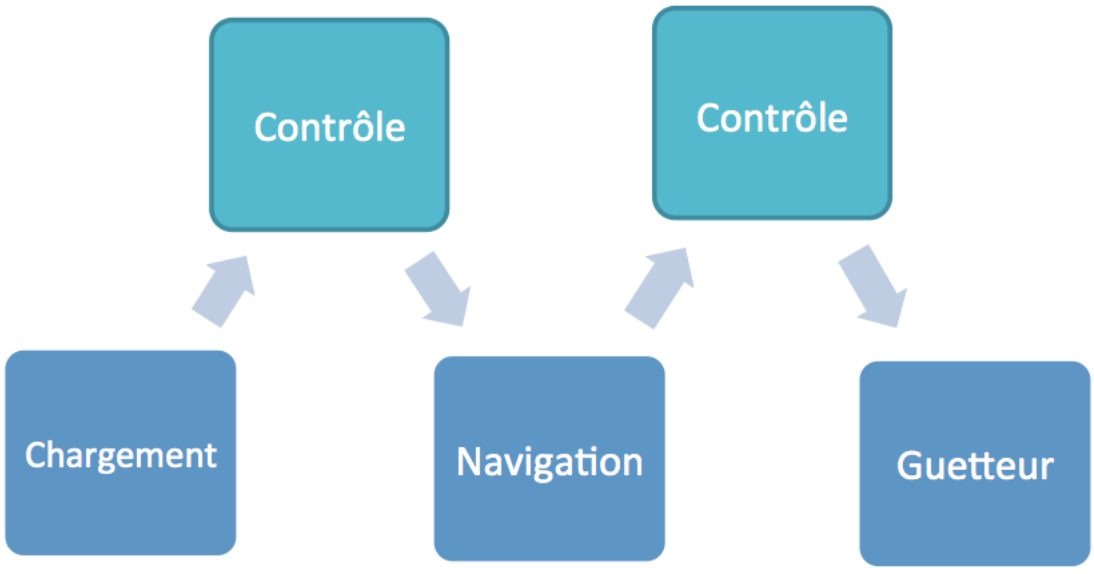
Ce fichier VOTable liste 3 galaxies avec leurs vitesses, distance d'erreur et la distance estimé. Il n'y a qu'une seule ressource et qu'une seule table de donnée associée.

3.4 Solution mise en place

3.4.1 Architecture générale

La librairie s'articule autour de 3 types de méthodes clés :

- Contrôle : Permet au développeur de s'assurer la bonne exécution de son code.
- Navigation : Permet de naviguer dans le fichier VOTable, dans les ressources et dans les catalogues.
- Guetteur : Permet de récupérer des données (nombre de catalogue, données d'une table, métadonnées d'une ressource, ...).



Dans un premier temps le fichier VOTable est chargé, l'idéal est de vérifié si le chargement s'est correctement effectué (phase 1).

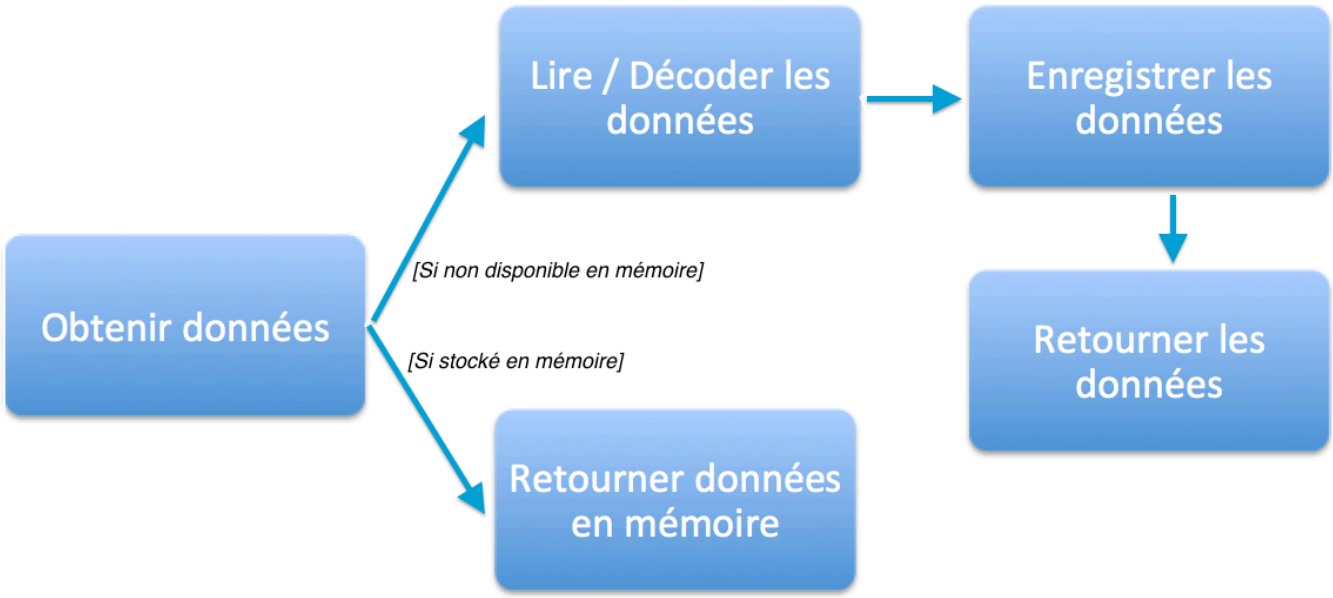
Dans un second temps nous naviguons dans le fichier VOTable pour aller là où nous souhaitons récupérer des données. Une fois arrivé à destination nous vérifions que nous sommes bien là où nous voulions aller (phase 2).
Pour terminer nous récupérons les données que nous souhaitons via les différents "Guetteurs" (phase 3).
Les phases 2 et 3 peuvent être répété à souhait au sein d'un fichier VOTable.

3.4.2 Gestion de l'encodage

L'analyseur syntaxique gère un encodage classique (UTF-8) mais aussi un encodage en base 64, utilisé pour compresser les données. Au niveau du développeur cette gestion est transparente, cependant il peut récupérer le type d'encodage présent dans le fichier VOTable.

3.4.3 Optimisation et système de mémoire.

Une des problématiques du projet est de pouvoir gérer des fichiers VOTables conséquent (plusieurs centaines de milliers de lignes de données). C'est pourquoi un système de mémoire a été mise en place pour enregistrer les traitements déjà effectués.

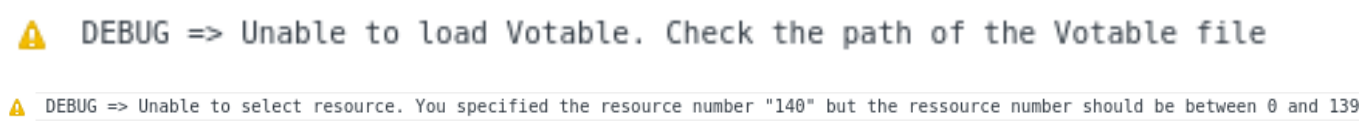


Ce système de mémoire peut être considéré comme un cache. Il est effectif uniquement pour les opérations lourde (lecture de flux de données et décodage base 64).

Les données sont sauvegardées dans un tableau à deux dimensions. La première dimension allant de 0 à n ressources et la seconde allant de 0 à n catalogues.

3.4.3 Mode Développeur

Pour faciliter l'intégration et le maintien de l'analyseur syntaxique, deux modes sont prévus, un mode développeur et l'autre pour la production. Le mode développeur génère des avertissements dans la console JavaScript du navigateur lorsque l'analyseur syntaxique détecte des cas d'utilisation irrégulier.



Exemple de message d'avertissement présent dans la console JavaScript.

La configuration du mode est disponible via la méthode "DisplayError(boolean)". Par défaut l'analyseur syntaxique est en mode développeur et affiche les messages d'avertissement.

3.4.4 Méthodes disponibles

3.4.4.1 Contrôles

DisplayError()	Affiche les erreurs générées par l'analyseur syntaxique.
WhoAmI()	Retourne la VOTable, la ressource et le catalogue courant.
IsLoaded()	Retourne "TRUE" si VOTable chargé avec succès, sinon "FALSE".
IsEncodedB64()	Retourne un "TRUE" si encodé en base 64, sinon "FALSE".

3.4.4.2 Navigation

SelectCatalog(integer)	Sélectionne le catalogue souhaité.
SelectResource(integer)	Sélectionne la ressource souhaitée.

3.4.4.3 Getter

GetField()	Retourne un objet avec l'ensemble des champs.
GetData()	Retourne un objet avec l'ensemble des données.
GetThisData(integer, integer)	Retourne un objet d'une plage de donnée défini en paramètre par un minimum et un maximum.
GetNbCatalogVotable()	Retourne le nombre de catalogue présent dans une VOTable.
GetNbCatalogResource()	Retourne le nombre de catalogue présent dans une ressource.
GetNbResource()	Retourne le nombre de ressource présent dans une VOTable.
GetEncodage()	Retourne l'encodage du fichier VOTable.
GetHtmlField()	Retourne l'ensemble des champs formatés en HTML.
GetHtmlData(integer, integer)	Retourne l'ensemble des données formatées en HTML.
GetVotableMeta()	Retourne les métadonnées présentes dans la VOTable.
GetResourceMeta()	Retourne les métadonnées présentes dans une ressource.
GetCatalogMeta()	Retourne les métadonnées présentes dans un catalogue.
GetCatalogInfo()	Retourne les informations liées à un catalogue.
GetResourceInfo()	Retourne les informations liées à une ressource.

3.5 Interface de démonstration

Pour faire une démonstration de la librairie, ainsi que pour effectuer des tests avec différent fichier VOTable, une interface de démonstration a été développée. L'objectif étant d'avoir une interface convivial et simple d'utilisation pour présenter le travail effectué et effectuer des benchmarks.
L'interface permet d'interagir avec la librairie et faire appel à des méthodes, l'outil s'articule autour de 4 étapes :

- Etape 1 : Charger un fichier VOTable, en spécifiant le chemin.
- Etape 2 : Sélection d'une ressource via un menu déroulant.
- Etape 3 : Sélection d'un catalogue via un menu déroulant.
- Etape 4 : Travailler avec la VOTable en utilisant des méthodes de la librairie.

En ce qui concerne l'étape 4, voici l'ensemble des méthodes accessible :

- GetField()
- GetData()
- GetVotableMeta()
- GetResourceMeta()
- GetCatalogMeta()
- GetThisData()
- GetHtmlData()

La majorité des méthodes renvoient un objet JavaScript consultable dans la console du navigateur web. Chaque action est historisé en haut à droite de l'interface pour que l'utilisateur sache précisément où il en est (quel fichier VOTable a été chargé, dans quelle ressource / catalogue il se trouve, etc ...).

Voici un aperçu de l'interface de démonstration :

3.6 Difficultés rencontrées et solutions

La plus grande difficulté du projet est sans aucun doute le décodage des données en base 64. A première vu la tâche semble aisé, on applique les règles de conversion base 64 / base 10 et on récupère les données, mieux encore, nous nous servons d'une fonction JavaScript dédié (fonction atob) et le tour est joué.

Cependant voici les difficultés liées à cette tâche :

- La partie codée en base 64 représente des données binaires, une simple conversion en base 10 rend les données inexploitables.
- Il est très difficile de travailler en binaire en JavaScript car c'est un langage haut niveau, il n'existe pratiquement aucune méthode intégré nativement.
- Toutes les données à récupérer sont à la suite les unes des autres, il n'y a aucun caractère pour délimiter 2 données de nature différentes.
- Au moindre bit mal interprété toutes les données sont corrompues.

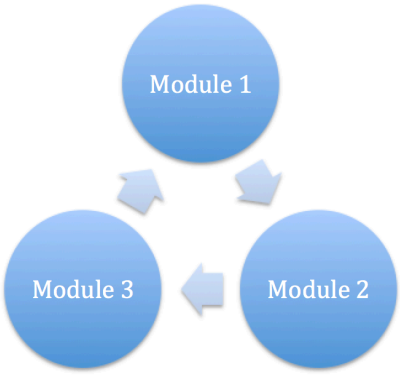
- Il faut effectuer des conversions selon plusieurs types différent (float 32 bits, double 64 bits, short 16 bits, byte 8 bits, string, etc ...).
- En général avant de décoder chaque type de données nous savons combien de bit il faut récupérer (renseigner autre part dans le fichier VOTable) cependant pour les données de type string, cette information est renseigné directement à l'intérieur de la partie en base 64 sous forme binaire. Ce qui complexifie encore le développement.
- En UTF-8 certaines données sont directement arrondies pour une lecture plus aisée, Cependant en base 64 les données sont brutes, il faut donc les arrondir selon la précision souhaitée.
- Nous ne connaissons pas à l'avance la taille total des données à décoder, il faut donc savoir quand s'arrêter sinon le dernier groupe de données décodé sera corrompu.
- Etc, ...

Technique utilisée pour solutionner le problème

Il est assez complexe d'expliquer à l'écrit comment nous avons résolu un problème aussi technique et pointu, plus encore si le lecteur n'a pas directement travaillé sur le projet. Nous allons donc nous contenter d'une explication sommaire en faisant abstraction des cas particuliers et de nombreux détails.

Le mécanisme s'articule autour de 3 modules :

- Module 1 : Il s'agit du module principal, il lit l'ensemble du flux de données, et fait appel au module 2 et 3 selon le nombre de bit à décoder et selon le type de données.
- Module 2 : Il reçoit les données en base 64 et retourne un tableau avec les données au format binaire. Il est appelé par le module numéro 1.
- Module 3 : Il s'agit d'un ensemble de fonction de conversion de type de donnée (binaire vers float, binaire vers string, binaire vers integer, ...). Il est appelé par le module numéro 2.



3.7 Utilisation de l'outil

L'analyseur syntaxique a été diffusé à l'ensemble des développeurs du CDS ainsi qu'à d'autres centres de données, dont celui de Paris. Il y a eu des retours sur le travail effectué, dont des corrections à effectuer, en général assez minime.

L'analyseur syntaxique a été utilisé pour la seconde partie du stage, mettant en avant certains bugs qui n'avaient pas été repéré au préalable, comme par exemple une incompatibilité sous Internet explorer et Safari pour récupérer les données entre balise XML ou encore la possibilité de sélectionner un catalogue qui n'existe pas (exemple : S'il y a 5 catalogues présent, il était possible de sélectionner le catalogue numéro 8).

3.8 Performance

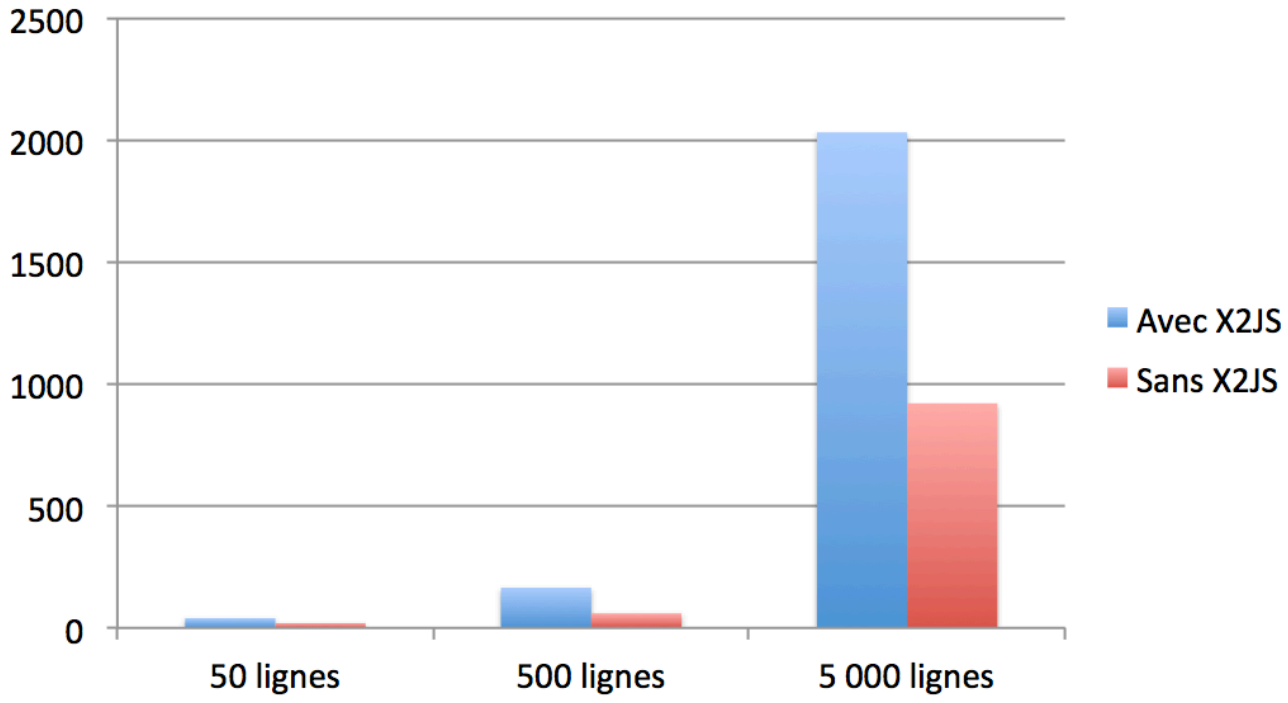
3.8.1 Méthodologie utilisée

Pour effectuer les mesures de performances nous nous servons de l'objet JavaScript "getTime". Celui-ci est lancé une première fois avant les traitements, puis une seconde fois après. Il suffit de soustraire la seconde mesure à la première pour avoir une idée du temps écoulé en milliseconde entre les deux mesures.

3.8.2 Avec la librairie X2JS / Sans la librairie X2JS

Il a été mentionné au début du projet que pour faciliter le développement il pouvait être judicieux de se servir d'une librairie existante qui se chargera de transformer le XML en JSON. Nous travaillons ainsi directement avec un objet JavaScript recensant toutes les informations du fichier VOTable.

Malheureusement, comme le montre le graphique ci-dessous les performances n'étaient pas au rendez-vous. En effet la librairie X2JS effectue un nombre conséquent de traitement et de test pour former la hiérarchie de l'objet JSON, à contrario notre librairie n'effectue pas ces traitements car nous connaissons la structure générale du fichier VOTable.
Cependant la plus grosse lacune de la librairie X2JS est d'effectué une lecture complète du fichier XML, chaque ligne est lu, de la première à la dernière. Notre librairie quand à elle effectue "des bonds" dans le fichier XML pour récolter directement les informations nécessaire, évitant ainsi d'effectuer des lectures inutiles.



Abscisse : Nombre de ligne du fichier VOTable.

Ordonné : Temps de traitement en milliseconde, plus le temps est faible, meilleur sont les performances.

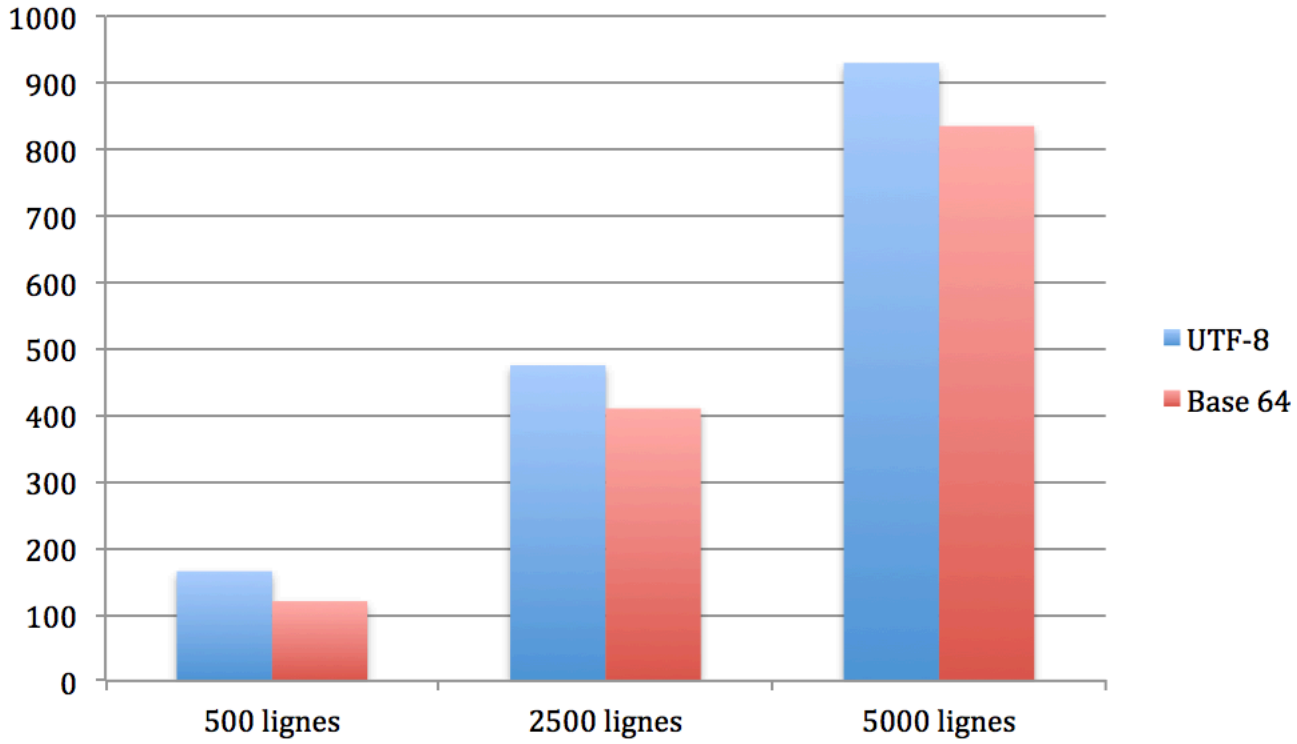
3.8.3 Base 64 / UTF-8

Les performances en base 64 sont légèrement meilleures, de l'ordre de 10%. Le gain de performance s'explique à cause du poids du fichier, en effet en base 64 les données sont représentées directement en binaire, le fichier étant plus léger, le navigateur le charge beaucoup plus rapidement.

Voici un tableau avec les poids des fichiers pour se rendre compte de la différence entre base 64 et UTF-8.

	UTF-8	Base 64
500 lignes	128 Ko	68 Ko
2 500 lignes	840 Ko	395 Ko
5 000 lignes	1 700 Ko	824 Ko

Les tests ont étés effectué en local, les fichiers VOTable était donc déjà présent sur le disque dur de l'ordinateur, or en utilisation normal ces fichiers doivent être téléchargé depuis un serveur du CDS. La différence devrait donc encore se creuser en faveur de la base 64 car les fichiers à télécharger ont une taille plus faible que leurs équivalent en UTF-8.



Abscisse : Nombre de ligne du fichier VOTable.

Ordonné : Temps de traitement en milliseconde, plus le temps est faible, meilleur sont les performances.

4. Développement d'interface homme machine

4.1 Contexte

La dernière interface utilisateur pour le service VizieR a été développée en 2008. Cependant au fil du temps des lacunes se sont greffés :

- Des fonctionnalités ont été ajoutées au fur et à mesure, cependant il est difficile de les intégrer à l'interface car celle-ci n'a pas été pensée de cette manière lors de la phase de conception.
- Il existe deux pages d'accueil, l'une d'elle est davantage plébiscité par les utilisateurs car elle est plus complète en terme de fonctionnalité.
- L'interface manque d'ergonomie pour un utilisateur inexpérimenté au service, il n'est pas rare de devoir effectuer plusieurs "clic" avant de trouver l'information désirée.

Pour supprimer ces problèmes il a été décidé de créer une interface utilisateur à partir de zéro. La nouvelle interface homme machine se devait plus dynamique, ergonomique et sobre.

4.2 Etat de l'art

Nous avons recherché des outils pour simplifier le développement de l'interface homme machine, tant sur le point visuel (CSS / HTML) que dynamique (JavaScript).

Nous avons sélectionné le framework **Bootstrap** pour la partie graphique, en effet il a l'avantage de fournir une feuille de style CSS qui contient les définitions de base pour tous les composant HTML. Un autre atout majeur fût sa sobriété, en effet le CDS développe des outils pour des chercheurs où l'efficience prime sur le graphisme.

Pour la partie dynamique nous avons sélectionné la librairie JavaScript **DataTable**. Elle permet de créer des tableaux dynamiques incluant des fonctionnalités intéressantes pour gérer les données (tri, pagination et recherche). La **librairie JavaScript de Bootstrap** nous a également intéressé, notamment pour bénéficier du composant "modal" (lors d'un clic sur une ancre un pop-up recouvre la page principal), l'idée étant de réduire le nombre de page visité par un utilisateur avant de trouver le contenu convoité.

Nous souhaitions également inclure le Framework JavaScript **jQuery** pour ajouter et supprimer du contenu aisément sur notre interface.

4.3 Description de VizieR

Une description succincte de VizieR est déjà disponible dans la partie 1.1.5.1 de ce rapport. Etant donné que la moitié du stage était dédié à développer une interface pour cette outil nous avons décidé d'étoffer un peu cette description dans les deux sous partie suivantes.

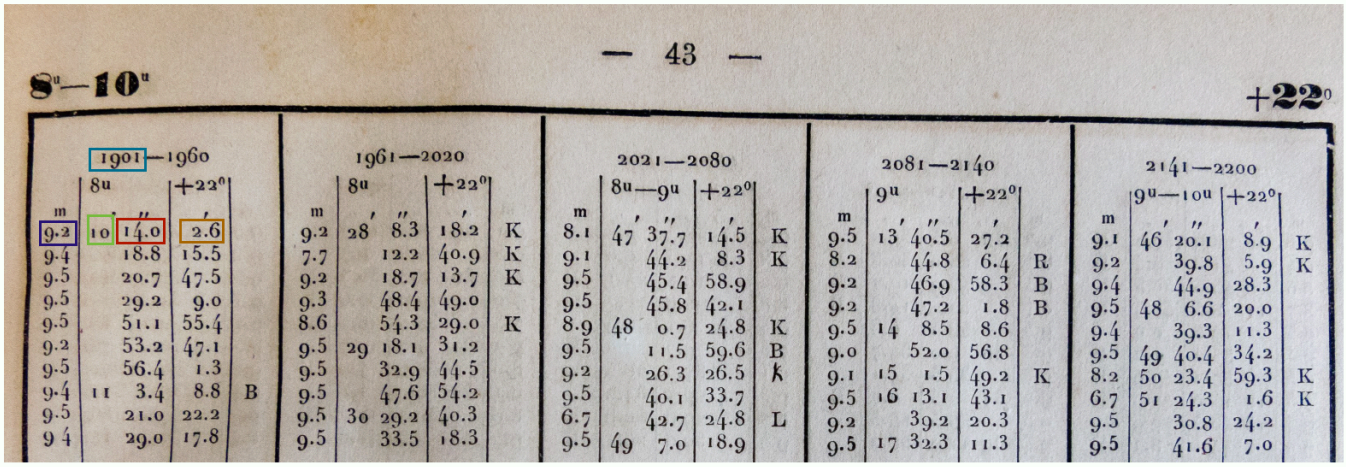
4.3.1 Les technologies utilisées avant VizieR

VizieR a été crée en 1996 pour stocker et diffuser des collections de données issu des catalogues. C'est actuellement l'outil plébiscité pour cette tâche, nous allons découvrir comment les astronomes accédait à ces données auparavant.

Avant 1972 les données issues de catalogue étaient répertoriées dans des livres classiques, les astronomes de l'observatoire astronomique se rendaient donc fréquemment à la bibliothèque (intégré dans l'observatoire) où ils demandaient à une documentaliste le livre où se trouvait les

données qu'ils recherchaient. Il n'y avait aucune automatisation et la gestion des données était avant tout un travail d'archivage.

Ci-dessous une comparaison du catalogue Bonner Durchmusterung (donne la position et l'éclat de 324 000 étoiles) entre un le livre original (publié en 1852 - 1859) et l'équivalent disponible sous VizieR.



— 43 —

8u—10u +22°

1901—1960				1961—2020				2021—2080				2081—2140				2141—2200			
8u			+22°	8u			+22°	8u—9u			+22°	9u			+22°	9u—10u			+22°
m	′	″	′	m	′	″	′	m	′	″	′	m	′	″	′	m	′	″	′
9.2	10	14.0	2.6	9.2	28	8.3	18.2 K	8.1	47	37.7	14.5 K	9.5	13	40.5	27.2	9.1	46	20.1	8.9 K
9.4		18.8	15.5	7.7		12.2	40.9 K	9.1		44.2	8.3 K	8.2		44.8	6.4 R	9.2		39.8	5.9 K
9.5		20.7	47.5	9.2		18.7	13.7 K	9.5		45.4	58.9	9.2		46.9	58.3 B	9.4		44.9	28.3
9.5		29.2	9.0	9.3		48.4	49.0	9.5		45.8	42.1	9.2		47.2	1.8 B	9.5	48	6.6	20.0
9.5		51.1	55.4	8.6		54.3	29.0 K	8.9	48	0.7	24.8 K	9.5	14	8.5	8.6	9.4		39.3	11.3
9.2		53.2	47.1	9.5	29	18.1	31.2	9.5		11.5	59.6 B	9.0		52.0	56.8	9.5	49	40.4	34.2
9.5		56.4	1.3	9.5		32.9	44.5	9.2		26.3	26.5 K	9.1	15	1.5	49.2 K	8.2	50	23.4	59.3 K
9.4	11	3.4	8.8 B	9.5		47.6	54.2	9.5		40.1	33.7	9.5	16	13.1	43.1	6.7	51	24.3	1.6 K
9.5		21.0	22.2	9.5	30	29.2	40.3	6.7		42.7	24.8 L	9.2		39.2	20.3	9.5		30.8	24.2
9.4		29.0	17.8	9.5		33.5	18.3	9.5	49	7.0	18.9	9.5	17	32.3	11.3	9.5		41.6	7.0

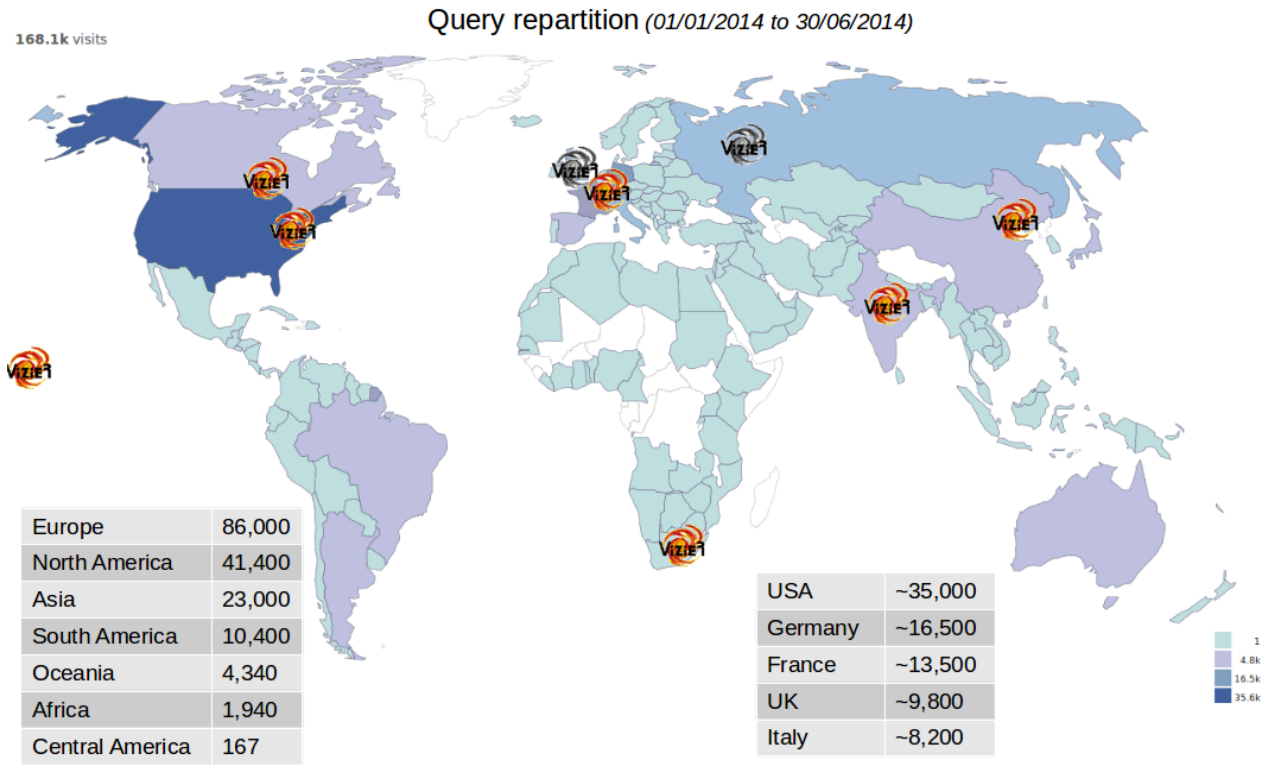
Full	RAJ2000 "h:m:s"	DEJ2000 "d:m:s"	zonesign	zone deg	num	suppl	mag mag	RA1855 "h:m:s"	DE1855 "d:m:s"	RA.icrs "h:m:s"	DE.icrs "d:m:s"
1	08 18 44.2	+21 35 46	+	22	1901		9.2	08 10 14.0	+22 02.6	08 18 44.2	+21 35 46
2	08 18 49.7	+21 48 40	+	22	1902		9.4	08 10 18.8	+22 15.5	08 18 49.7	+21 48 40
3	08 18 53.4	+22 20 39	+	22	1903		9.5	08 10 20.7	+22 47.5	08 18 53.4	+22 20 39
4	08 18 59.7	+21 42 08	+	22	1904		9.5	08 10 29.2	+22 09.0	08 18 59.7	+21 42 08
5	08 19 24.1	+22 28 28	+	22	1905		9.5	08 10 51.1	+22 55.4	08 19 24.1	+22 28 28
6	08 19 25.7	+22 20 09	+	22	1906		9.2	08 10 53.2	+22 47.1	08 19 25.7	+22 20 09

Les carrés de différentes couleurs représentent l'équivalence des données.

De 1972 à 1996 il est décidé de faciliter l'accès aux catalogues astronomique via l'utilisation de bande magnétique et de microfiche. L'objectif étant de réduire l'encombrement pris par les livres astronomique (la bibliothèque n'est pas extensible) mais aussi de simplifier le travail des astronomes, en effet il n'est pas rare que les astronomes travaille sur plusieurs catalogue en parallèle.

4.3.2 Statistiques du service

En moyenne le service VizieR reçoit 1 000 visites journalières, variant au grès de l'apparition de nouveaux catalogues. Les statistiques mettent en évidence l'internationalisation de la communauté scientifique, en effet bien que développé à Strasbourg, l'outil reçoit davantage de visiteur provenant des Etats-Unis que de France.



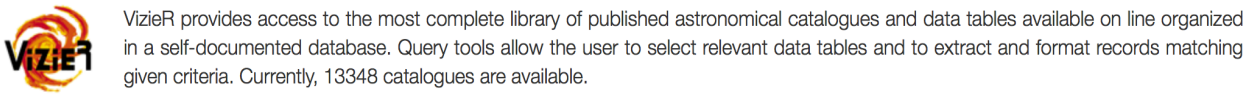
Les logos VizieR indique sur la carte où se situe un miroir du service.

4.4 Solution mise en place

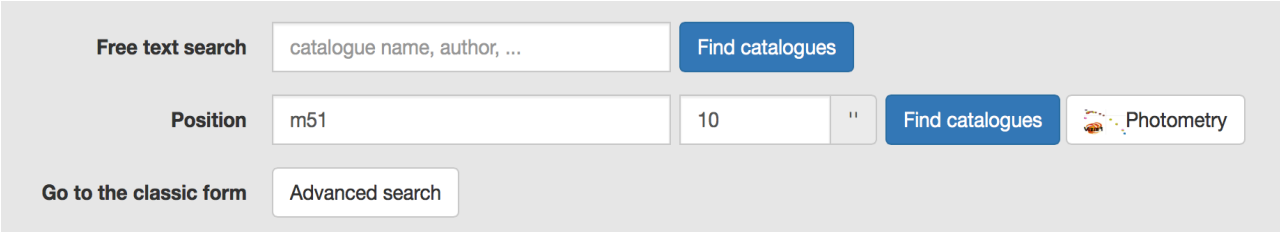
4.4.1 Ensemble des modules présents

L'interface homme machine est composé de 8 modules dont voici leurs descriptions classé selon leurs positions à la verticales sur l'interface :

- **Module 1** : Description du service VizieR, incluant le nombre de catalogue disponible sur l'application web, ce nombre est mis à jours régulièrement via une crontab.

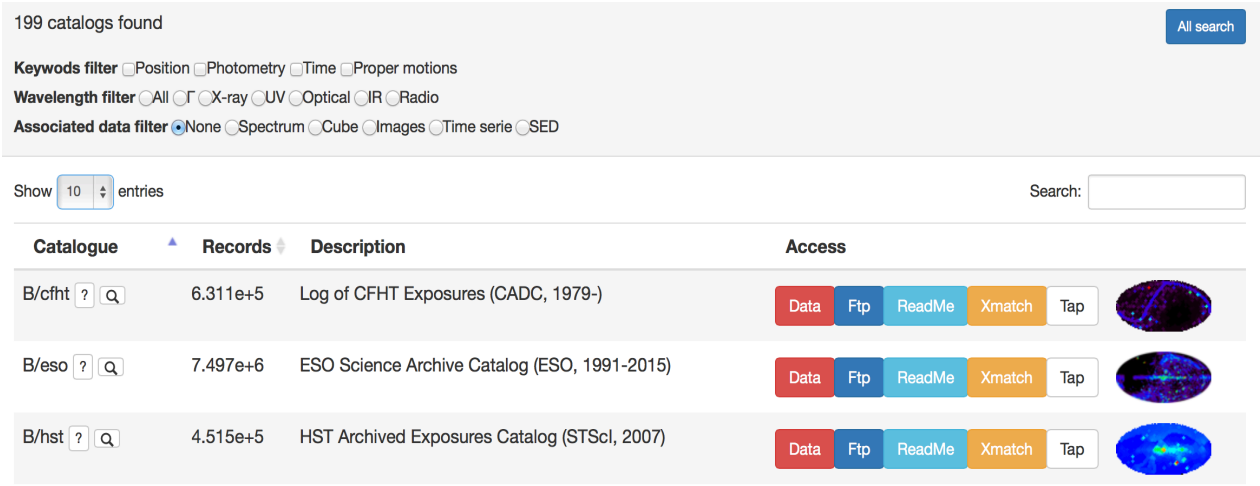


- **Module 2** : Zone de recherche, permet de rechercher un catalogue par mots clés ou via une position (dans ce cas il est possible de spécifier le rayon de recherche dans l'espace en radius, et de visionner la photometry lié à la recherche). Le bouton "Advanced search" permet de se rendre sur une page avec plus de fonctionnalité en terme de recherche.

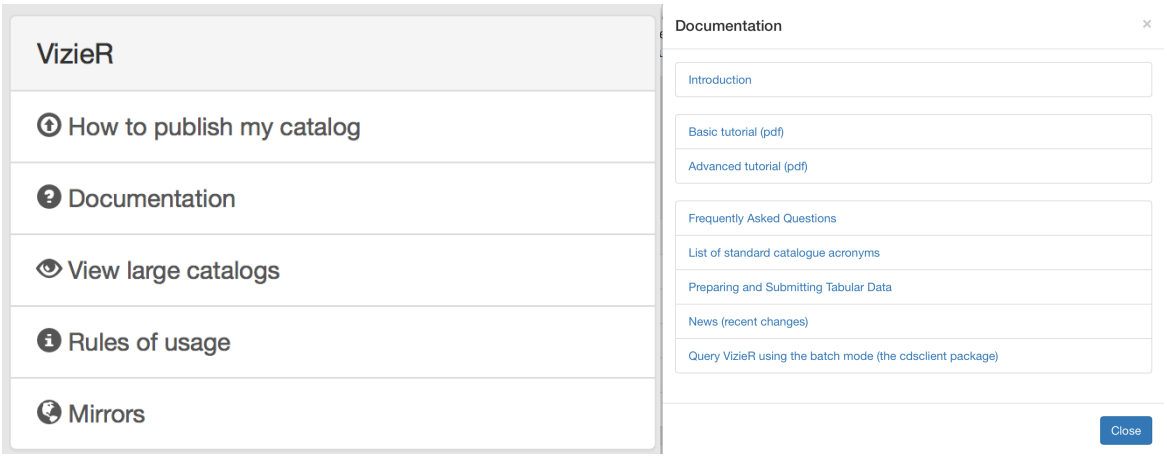


- **Module 3** : Résultat de la recherche (visible uniquement après une recherche). Ce module est le plus complet de tous en terme de fonctionnalité, en voici la liste :
- Affiche le nombre de résultat.
- Permet de trier les résultats grâce à des filtres.
- Affiche les mots clés par catalogue via une tooltip.
- Système de pagination dynamique, pour naviguer entre les pages web sans rechargement de la part du navigateur.

- Permet d'afficher 10, 25, 50 ou 100 résultats sur la page des résultats.
- Système de recherche pour trouver un catalogue en particulier rapidement.
- "Bouton All search" pour afficher des informations supplémentaire sur un ensemble de catalogue.

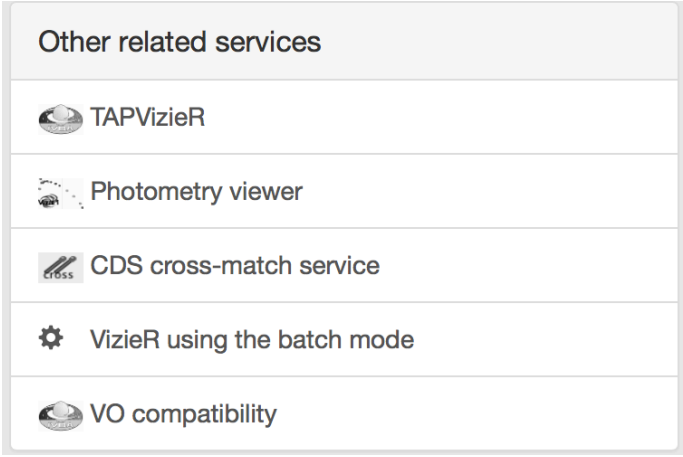


- **Module 4** : Ce module répertorie des liens vers le fonctionnement de VizieR, sa documentation (sous forme de PDF), les conditions d'utilisation, la liste des miroirs du service, etc ... pour améliorer l'ergonomie de l'interface certain lien ouvre un modal (page qui recouvre la page existante) avec le contenu à afficher. Le but est de limiter le nombre de page que l'utilisateur doit visiter pour accéder au contenu désiré.

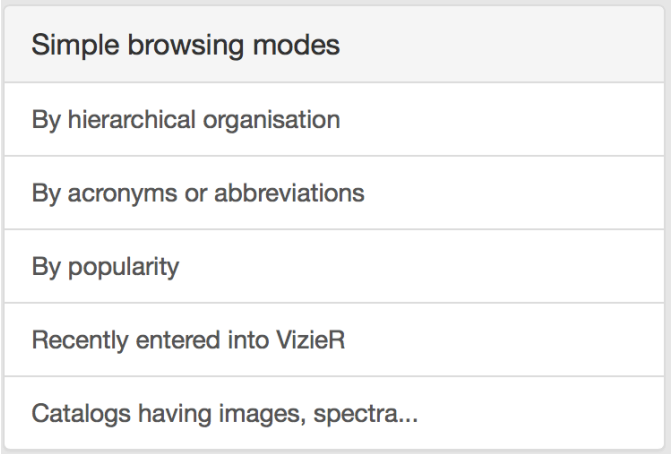


- **Module 5** : Ce module est une simple liste de liens vers des services complémentaire à VizieR. Comme par exemple le service de cross-match qui

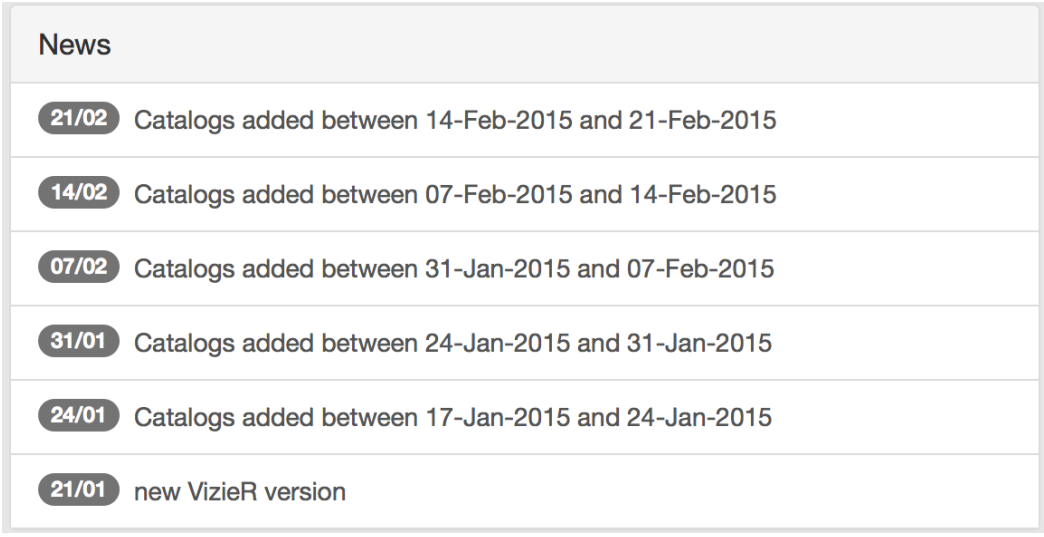
permet de croiser des sources identique entre deux catalogues, ou encore le service de photometry qui permet une visualisation de point de photométrie extraites autour d'une position ou d'un nom d'objet donné à partir de catalogues présent dans VizieR.



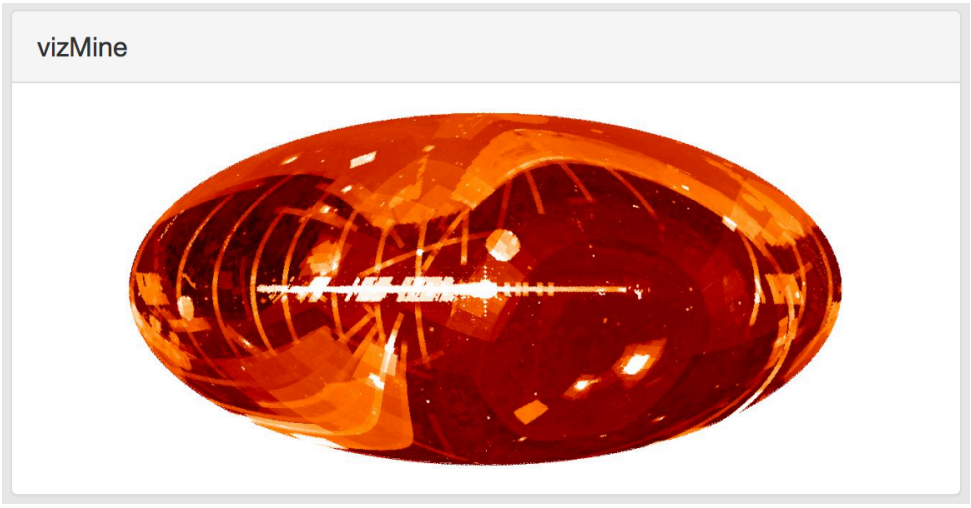
- **Module 6** : Ce module représente des classements et des statistiques liées aux catalogues stockés par VizieR. Par exemple un classement des catalogues les plus populaires, les derniers catalogues ajoutés aux services, les catalogues possédant une image ou un spectre. Pour afficher les classements nous utilisons la même librairie JavaScript que pour le module numéro 3.



- **Module 7** : Liste des dernières actualités publiées sur le site internet du CDS. La liste est chargée dynamiquement au lancement de la page par un appel Ajax, nous récupérons un flux XML qui est parsé. Nous affichons les résultats par ordre chronologique décroissant uniquement pour les 6 dernières actualités.



- **Module 8** : Il s'agit d'une image du ciel qui est une représentation de la densité des objets accessibles dans VizieR, en coordonnées galactiques. Plus l'image est blanche à des endroits, plus nous avons de données correspondant à ces points.



4.4.2 Pages additionnels

Un certain nombre de page additionnel ont étés développés, notamment pour montrer des classements et des statistiques liées aux catalogues présent sur VizieR.

L'ensemble de ces pages utilise la librairie "DataTables" pour afficher les données, notamment pour profiter des fonctionnalités de la librairie, à savoir :

- Recherche à partir de mot clés.

- Pagination des pages de recherche.
- Tri par colonne (croissant et décroissant).
- Affichage de 25, 50 ou tous les résultats.

Ces fonctionnalités rendent l'expérience utilisateur plus agréable car il peut manipuler facilement les données s'il le souhaite.

Voici l'ensemble des pages additionnels crées :

- Nombre de catalogue disponible par catégorie.
- Correspondance entre acronyme et catalogue.
- Classement des catalogues par popularité.
- Liste des derniers catalogues ajoutés.
- Liste des catalogues possédant une image, un spectre, ...

L'ensemble des données est récupéré au chargement de la page via un appel Ajax sur une page où sont stockés les données au format JSON. Les données sont automatiquement mises à jour via des scripts maison appelée fréquemment via des crontab. La gestion des données ne rentre pas dans le cadre du projet, il s'agit d'une existant dont nous nous sommes servit.

4.5 Difficultés rencontrées et solutions

Moins de difficultés ont étés rencontrés lors du développement de l'interface homme machine car ce développement été moins technique que celui de l'analyseur syntaxique.

Cependant quelques points on quand même ralentie l'avancé du projet de manière significative.

Tout d'abord des soucis de compatibilité entre les différents navigateurs, en effet des bugs ont étés constatés au sein de l'analyseur syntaxique sous internet explorer et

Safari, il s'agissait de la méthode "innerHTML" qui n'était pas utilisable dans un document XML. Pour parer le problème nous avons utilisé une expression relationnelle (regExp) pour récupérer le contenu entre la balise souhaité.

Nous avons également perdu un temps significatif pour comprendre et appréhender le fonctionnement des différentes librairies utilisés, par exemple pour le tableau de données géré par la librairie "DataTables" nous ajoutions d'abord manuellement le contenu dans l'arbre DOM de la page web via la méthode "append" disponible en jQuery. Cependant il existe une méthode spécifique à la librairie "DataTables" pour exécuter ce genre d'opération, à savoir "row.add". Cette méthode évite des conflits avec la librairie (contenu ajouté dans la page, mais la librairie pense que la table de donnée est vide car le contenu n'a pas été ajouté via la méthode adéquat).

Le dernier point qui a sans doute ralenti le développement de manière significative est sans doute l'intégration des modifications et améliorations souhaités au fur et à mesure par l'ensemble des personnes de la communauté scientifique, en effet il a fallu sans cesse adapté le travail effectué au préalable avec les modifications souhaité. La lisibilité du code en a souffert, il a donc fallu l'éclater en module pour retrouver une clarté à l'intérieur de celui-ci. De plus il fallait à chaque fois réfléchir à où placer les nouvelles fonctionnalités car celle-ci n'avait pas été imaginée dans l'interface de départ.

4.6 Protocole de test

L'outil s'adresse exclusivement à des astronomes avertis, c'est pourquoi il est nécessaire de travailler en collaboration avec eux pour être sûr que la nouvelle interface homme machine réponde intégralement à leurs besoins.

C'est pourquoi des réunions ont étés organisés avec un astronome chargé de représenter l'ensemble de la communauté scientifique pour ce projet. Ces remarques ont étés prise

en compte pour corriger l'évolution du projet et intégrer des améliorations pour rendre l'interface plus fonctionnelle.

Dès qu'une version de l'interface posséder un minimum de fonctionnalités viable il a été décidé de le soumettre en accès libre à l'ensemble de la communauté astronomique. Pour ce faire une version béta a été intégrée sur le site internet existant. Cette version a été crée dans un dossier séparé du reste du site, disponible à l'adresse suivante : http://vizier.u-strasbg.fr/beta/index.gml
Une actualité a été mise en place sur le site internet du CDS pour diffuser l'adresse de la béta test et ainsi récolter les avis des utilisateurs du service. Ainsi nous avons pu améliorer l'interface en intégrant les modifications souhaités par les utilisateurs ou encore ajouter des fonctionnalités qu'ils désirés, comme par exemple le bouton "Advanced search" dans les résultats issu d'une recherche.

5 Conclusion

Ce stage a été l'occasion pour moi d'apprendre de nouvelle chose, tant sur le point du contact humain, que scientifique.

Il m'a été permis de découvrir de nouvelles technologies via le développement d'une interface homme machine et d'un analyseur syntaxique. J'ai été très bien accueilli par l'équipe du Centre de Donnée Astronomique de Strasbourg et j'ai souvent pu compter sur le soutien du personnel pour le développement de mes deux projets.

L'environnement et les moyens mis à disposition furent propices à un travail sérieux et appliqué. Le stage fût découpé en deux projets : Premièrement le développement d'un analyseur syntaxique pour traiter des données astronomique formaté en XML. Deuxièmement une interface homme machine pour l'un des services web du CDS. Les deux projets sont liés car l'analyseur XML est nécessaire au bon fonctionnement de l'interface utilisateur.

6 Glossaire

VOTable : Format XML utilisé pour l'échange de données astronomiques, standardisé par l'IVOA en 2003.

CDS : Centre de données Astronomie de Strasbourg, lieu du stage.

IVOA : International Virtual Observatory Alliance.

ASSOV : Action Spécifique Observatoire Virtuel France.

Framework : Ensemble cohérant de composant et de module assurant une structure logiciel de base à une application.

Crontab : Programme sous Unix qui permet de lancer des actions à des dates défini.

7 Bibliographie

Site officiel des services du Centre de données Astronomique de Strasbourg

CDS : http://cdsweb.u-strasbg.fr/index-fr.gml

VizieR : http://vizier.u-strasbg.fr

Simbad : http://simbad.u-strasbg.fr/simbad/

Aladin : http://aladin.u-strasbg.fr/aladin.gml

Technologies utilisés

jQuery : http://jquery.com

Bootstrap : http://getbootstrap.com

Ajax : http://fr.wikipedia.org/wiki/Ajax_(informatique)

Librairie X2JS : https://code.google.com/p/x2js/

Librairie DataTables : https://datatables.net

Site utile durant le développement des projets

StackOverFlow : http://stackoverflow.com

Ressource JavaScript : http://www.w3schools.com/js/

Indentation de code en ligne : http://jsbeautifier.org

Ressource utile au rapport

Historique du CDS : http://simbad.u-strasbg.fr/Files/CDS-informatique-40ans.pdf

Mots clefs

Fonction publique - Recherche - bases de données - IHM - logiciel d'analyse de données

THOMAS Rolling **Rapport de stage ST40 - P2015**

Résumé

Ce rapport résume 6 mois passés au sein du Centre de Données astronomique de Strasbourg (CDS), qui a pour mission de rassembler des informations utiles concernant les objets astronomiques et de les distribuer à la communauté astronomique internationale.

En collaboration étroite avec mon tuteur de stage et le responsable du service VizieR, j'ai notamment participé au développement d'un analyseur syntaxique pour le format (XML) VOTable et d'une nouvelle interface homme machine pour le service VizieR.

Le rapport tente donc de décrire de manière exhaustive les méthodes de travail que j'ai utilisées au sein de l'institut de recherche, les difficultés liées au développement et les contraintes techniques auquel j'ai du faire face. Je présente aussi une synthèse des résultats obtenus ainsi que les bénéfices à la fois techniques et humains que j'ai pu tirer grâce à ce stage.

Observatoire astronomique de Strasbourg
11 rue de l'Université
67000 Strasbourg