Rapport de stage :

Développement de nouvelles fonctionnalités pour le portail du CDS

Bertrand Benjamin 2008 / 2009

Centre de données astronomiques de Strasbourg

DUT Informatique
Université Nancy 2 - IUT Nancy Charlemagne

Rapport de stage :

Développement de nouvelles fonctionnalités pour le portail du CDS

Bertrand Benjamin 2008 / 2009

Centre de données astronomiques de Strasbourg
Observatoire Astronomique de Strasbourg
11, rue de l'université
67000 STRASBOURG

Tuteurs entreprise : André Schaaff, Thomas Boch
Tuteur enseignant : Bernard Mangeol

DUT Informatique
Université Nancy 2 - IUT Nancy Charlemagne

Résumé

Ce document a pour but de décrire le travail effectué durant mon stage à l'observatoire astronomique de Strasbourg. J'y présenterai tout d'abord, brièvement, le cadre dans lequel s'est déroulé le stage ainsi que l'existant et les développements en cours. J'essaierai ensuite d'expliquer plus en détail celui-ci et les ajouts que j'y ai apportés.

Ce stage s'est effectué dans le cadre du DUT informatique et a pour but d'ajouter des fonctionnalités au portail du CDS (Centre de Données astronomique de Strasbourg) en cours de développement depuis maintenant un an et demi. Un projet qui est donc assez conséquent et qui m'a permis de travailler sur une structure plus complexe mais néanmoins intéressante.

La découverte d'une technologie fut également un des points forts de ce stage. GWT pour Google Web Toolkit est un framework java utilisé pour développer des applications internet. Outre le fait que GWT utilise un système de Remote Procedure Call pour communiquer entre ses parties client et serveur, il génère les codes HTML et Javascript correspondant.

Remerciements

Je tiens tout d'abord à adresser mes remerciements à M. André Schaaff, ingénieur de recherche qui m'a permis d'obtenir ce stage au centre de données astronomiques et m'a encadré agréablement et à M. Thomas Boch, également ingénieur de recherche, qui m'a suivi tout au long de mon stage et n'a pas hésité à répondre à mes questions lorsque j'en avais besoin. C'est une personne avec laquelle j'ai particulièrement apprécié de travailler sur le portail. J'ai suivis ses conseils tout en lui soumettant mes idées qu'il a écoutées et dont nous avons débattu.

Je leur suis sincèrement reconnaissant de m'avoir permis de participer à cette grande alliance qu'est l'IVOA (International Virtual Observatory Alliance), et à laquelle le CDS participe activement.

Merci à M. Bernard Mangeol, mon responsable enseignant de l'IUT Nancy-Charlemagne, pour les conseils donnés lors de sa visite à l'observatoire.

J'aimerais remercier également tous les astronomes pour qui nous développons, en tant qu'informaticiens, des outils leur permettant de nous faire découvrir l'univers qui nous entoure.

Enfin, je remercie sincèrement toutes les personnes de l'observatoire et du CDS qui m'ont accueilli chaleureusement, et avec qui j'ai partagé de nombreuses conversations et particulièrement Vincent Meslard avec qui j'ai sympathisé, ainsi que Dominika Zabrowska avec qui j'ai partagé mon bureau pendant ces quelques semaines.

Table des matières

Résumé....2

Remerciements....3

I – L'Observatoire....6
1. Présentation générale....6
2. Les domaines de recherche....6
1. Hautes énergies....6
2. Étoiles et systèmes stellaires....7
3. Galaxies....7
4. Le centre de données (CDS)....7
3. Le Centre de Données astronomiques de Strasbourg....7
1. Présentation....7
2. Les services du CDS....8
1. Simbad....9
2. VizieR....10
3. Aladin....11
4. International Virtual Observatory Alliance....12
1. Historique....12
2. Place du Centre de Données de Strasbourg au sein de l'IVOA....12

II - Le Portail du CDS....13
1. Objectif du portail....13
2. GWT (Google Web Toolkit)....13
1. Présentation....13
2. Un framework performant....13
3. Structure générale....15
4. Les développements....16
5. Les modifications que j'ai apportées....17
III – Déroulement du stage....18
1. Découverte de l'environnement de travail....18
1. Stage de Cédric CAPOULUN....18
2. Normes et standards....18
3. Premier projet, convertisseur VOTable => TSV, CSV,....19
1. STIL : parser de VOTable....19
2. Intégration du convertisseur au service de téléchargement....21
3. Envoi de mails en anglais au concepteur de STIL sur des cas non traités....21
2. Le Plotter de données....21
1. Présentation....21

---

1. Objectif du plot....22
2. STILToolSet, des outils pour STIL....23
2. Interface graphique....23
1. Le StackPanel....23
2. Le Curseur....23
3. Les formulaires....24
3. La servlet....26
1. Création de colfits....27
2. Gestion d'un cache....28
3. Création des images....28
4. Fonctionnalités asynchrones....29
1. Schéma....30
2. Récupération des métadonnées....30
3. L'Histogramme....30
4. Perspectives....31
1. L'histogramme....31
2. La carte de densité....32
3. Finalisation....32

IV – Conclusion....33
1. Bilan personnel....33
2. Bilan professionnel....33
3. Conclusion....34

V – Annexes....35
1. Lexique....35
2. Exemples....36
1. VOTable....36
2. Code GWT....38
3. Javascript généré par GWT:....39

I – L'Observatoire

1. Présentation générale

L'observatoire astronomique est une Unité du CNRS et de l'Université de Strasbourg. Il est situé sur le campus de l'Esplanade, non loin du centre ville de Strasbourg. Son personnel se compose d'astronomes, d'informaticiens et de documentalistes. Les enseignements suivants y sont dispensés :

- Master « Analyse et Traitement des Données sur les Milieux Astronomiques »
- Licence en Sciences et autres licences (enseignements d'ouverture)
- Licence de Géosciences
- Préparation au CAPES et à l'Agrégation
- Master Applications des Technologies Spatiales
- Diffusion de la Culture

A proximité des trois bâtiments de l'observatoire, se trouve, le planétarium destiné à l'astronomie grand public, et où les gens peuvent venir s'initier à la découverte de l'univers.

2. Les domaines de recherche

Au même titre que l'espace, les domaines de recherche en astronomie sont vastes. C'est pourquoi on y trouve différentes équipes de recherche spécialisées chacunes dans un domaine.

1. Hautes énergies

L'équipe Astrophysique des Hautes énergies a pour thème l'étude des astres et sites de l'univers émetteurs de photons de haute énergie. Cette thématique générale recouvre des aspects variés, comme l'étude des astres compacts en fin d'évolution, la physique de leur activité, les phénomènes de haute énergie intéressant les étoiles jeunes ou le soleil, ou l'étude de ces phénomènes à l'échelle galactique. Ces recherches se sont largement appuyées sur les données acquises par le satellite ROSAT et s'appuient désormais sur celles des satellites X de nouvelle

génération, tout spécialement XMM1.

2. Étoiles et systèmes stellaires

Les recherches menées par l'équipe « Étoiles et systèmes stellaires » recouvrent un domaine étendu, incluant les étoiles, les milieux interstellaires, la galaxie et les galaxies proches. De l'étoile, objet individuel, l'intérêt s'est porté aux groupes d'étoiles, témoins de l'évolution stellaire mais aussi traceurs des grandes structures de la Voie Lactée.

3. Galaxies

Les activités de l'équipe sont centrées sur les problèmes de la structure du Groupe Local, de ses populations stellaires et sur la dynamique gravitationnelle. De plus, l'équipe possède un savoir-faire sur les outils statistiques d'analyse de données et sur les méthodes inverses non paramétriques. Un des objectifs consiste à combiner les informations d'évolution des populations stellaires et celles de dynamique afin de reconstituer les événements déterminants liés aux processus de formation et d'évolution galactique.

4. Le centre de données (CDS)

L'activité de recherche du CDS s'est concentrée sur l'étude de la dynamique galactique et des populations d'étoiles binaires, sur une participation importante à la mission HIPPARCOS2 de l'Agence Spatiale Européenne, ainsi que sur le développement de méthodologies nouvelles applicables à l'analyse et au traitement de données astronomiques. Il participe activement à la mise en place de l'Observatoire virtuel que nous décrirons plus loin.

3. Le Centre de Données astronomiques de Strasbourg

1. Présentation

Le CDS joue un rôle dans d'importantes missions astronomiques spatiales : contribuant aux catalogues d'étoiles guides, aidant à identifier les sources observées, organisant l'accès aux archives, etc. Le CDS contribue au XMM SSC3, sous la responsabilité de l'équipe « Hautes-Énergies » de l'Observatoire de Strasbourg. Le service a également signé des accords d'échanges internationaux avec les organismes suivants :

- NASA

- National Astronomical Observatory (Tokyo, Japon)
- l'Académie des Sciences de Russie
- le réseau PPARC Starlink au Royaume-Uni
- l'Observatoire de Beijing (Chine)
- l'Université de Porto Allegre au Brésil
- l'Université de La Plata en Argentine
- InterUniversity Center for Astronomy and Astrophysics (Inde)

Le CDS est membre de la Fédération des Services d'Analyse de Données Astrophysiques et Géophysiques.

Le CDS coopère aussi avec l'Agence spatiale Européenne (transfert au CDS du service de catalogues du projet ESIS : le projet VizieR) et avec la NASA : en particulier le CDS abrite une copie miroir du Système de Données Astrophysiques (ADS) et ADS abrite une copie miroir de Simbad. Le CDS contribue aussi au projet NASA AstroBrowse. Le CDS abrite aussi les copies miroirs Européennes des journaux de l'American Astronomical Society (AAS).

2. Les services du CDS

Le CDS développe et maintient 3 principaux services qui constituent chacun des références dans le milieu de l'astronomie.

1. Simbad

Simbad est une Base de données d'identificateurs d'objets astronomiques permettant d'associer à un objet tous les identificateurs cités dans la littérature scientifique depuis 1980. Cette base de données contient actuellement 13 000 000 d'identificateurs pour 4 500 000 objets et 230 000 articles référencés. Ce service traite 100 000 requêtes par jour, est aujourd'hui une référence mondiale, et ne connaît pas d'équivalent. J'ajouterai qu'un stagiaire, l'an dernier, a cité dans son rapport 11 000 000 identificateurs, 3 000 000 d'objets et 110 000 articles, ce qui montre une croissance significative.

Le service permet de retrouver un objet astronomique à l'aide de son nom s'il est contenu dans les articles de la base de données. Il fournit des informations de base sur l'objet comme sa position, son type, quelques mesures effectuées ainsi que l'ensemble des articles où il est cité.

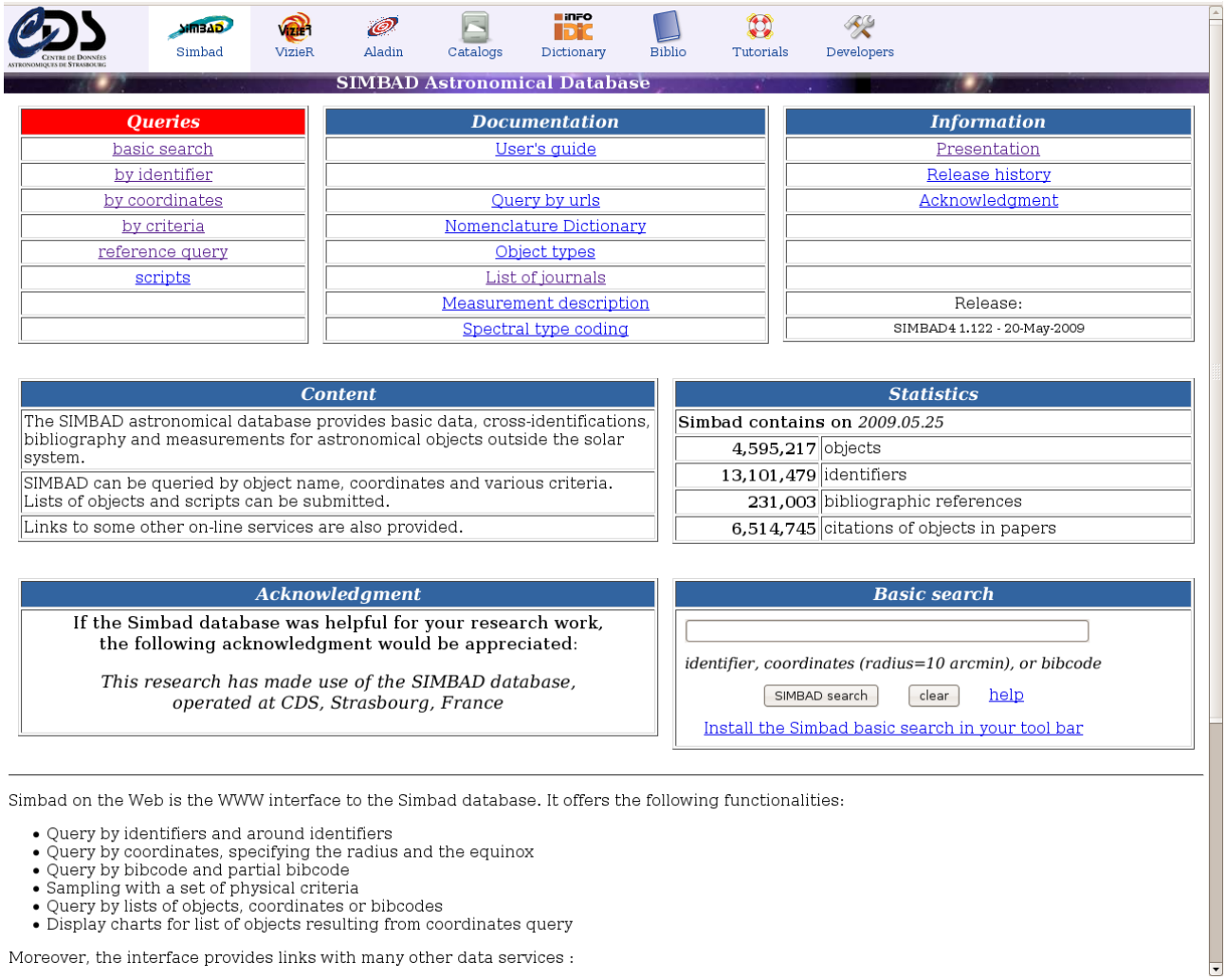


Figure 1 - Page d'accueil de Simbad

2. VizieR

VizieR est une base de données regroupant plusieurs milliers de catalogues d'objets astronomiques et permet d'accéder d'une manière uniforme aux données, de les croiser, de les exporter, etc. Comme on peut le remarquer sur la figure 2, l'exhaustivité de Vizier, sa performance et le nombre de ses outils le rendent peu intuitif. Sur la figure 2 , on peut voir dans la partie colorée verte et orange trois boutons pour valider son formulaire, soit trois pages différentes résultant d'un même formulaire d'origine. Ce qui peut perdre le visiteur même après plusieurs visites successives et lui demande un temps d'adaptation pour utiliser l'application.

VizieR recherche dans plus de 7000 catalogues de différentes tailles, certains de petite envergure mais aussi d'autres très important comme le catalogue 2MASS comportant près de 500 millions d'objets, GSC2 950 millions, USNO un milliard ou NOMAD contenant également un milliard de lignes. Ce service répond à plus de 100 000 requêtes par jours.

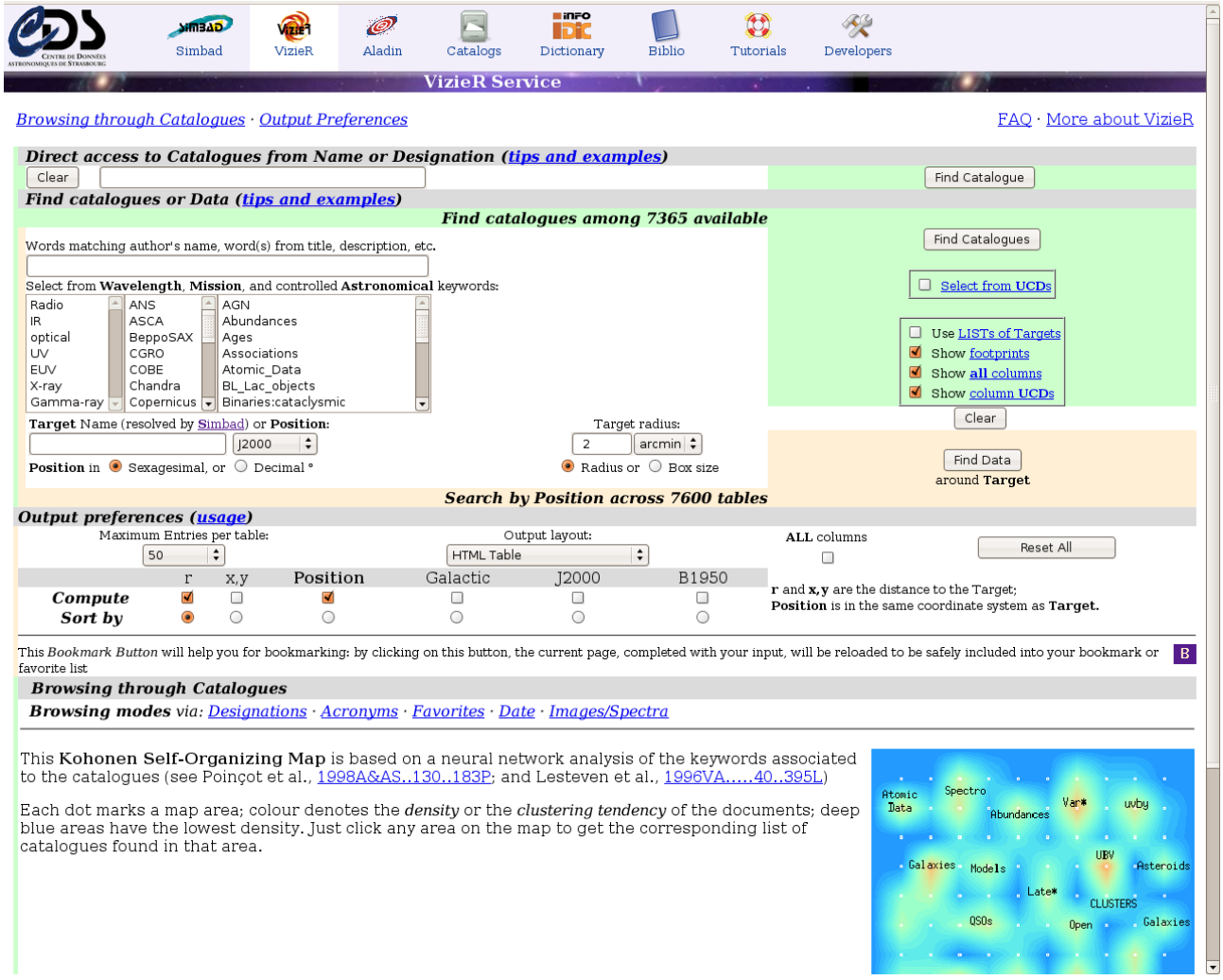


Figure 2 - Page d'accueil de Vizier

3. Aladin

Aladin est un atlas interactif du ciel ainsi qu'un logiciel de visualisation et de traitement d'images astronomiques. Il permet entre autres de générer des images en couleur à base d'images du ciel prises à différentes longueurs d'ondes en appliquant un filtre rouge, vert ou bleu pour chacune d'elles. Il permet également de superposer sur l'image obtenue, des catalogues de données à partir de VizieR ou Simbad, qu'il peut interroger par le biais d'une interface simplifiée. Aladin est un outil également complet et performant qui fait actuellement plus de 170 000 lignes de codes en Java.

On notera également qu'entre 6 et 10 000 requêtes par jours sont effectuée sur d'Aladin de la part de 400 à 500 utilisateur différents.

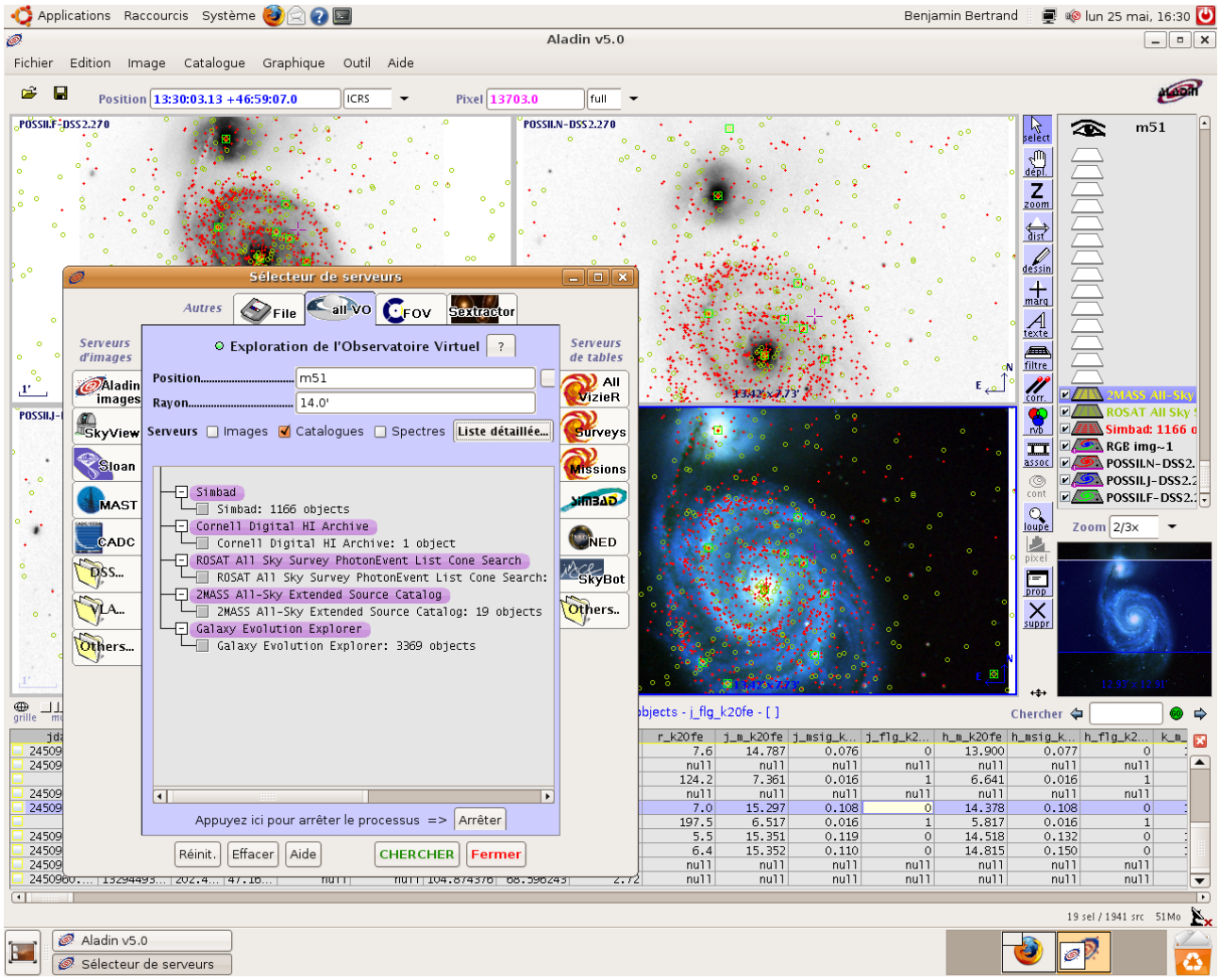


Figure 3 - Comparaison d'image et de données avec Aladin

4. International Virtual Observatory Alliance

1. Historique

L'IVOA est une organisation internationale ayant pour but « de faciliter les échanges internationaux et la collaboration nécessaire à l'élaboration et au déploiement d'outils, de systèmes et de structures organisés nécessaires pour permettre une utilisation internationale des archives astronomiques dans un unique observatoire virtuel exploitable ». Telle est la mission confiée à l'IVOA lors de sa création en juin 2002.

La participation à l'IVOA est libre mais à l'heure actuelle seize projets sont en cours à l'initiative de l'Arménie, l'Australie, le Canada, la Chine, l'Europe, la France, l'Allemagne, la Hongrie, l'Inde, l'Italie, la Corée, le Japon, la Russie, l'Espagne, le Royaume-Unis et les Etats-Unis.

L'ESA (Agence Spatiale Européenne) et l'ESO (Organisation Européenne pour la recherche en Astronomie) participent activement à cette internationalisation et standardisation des échanges.

2. Place du Centre de Données de Strasbourg au sein de l'IVOA

En 2009, le centre de données est reconnu comme une « très grande infrastructure de Recherche » par l'état. Il joue un rôle, depuis longtemps, dans les collaborations internationales. Il est notamment reconnu par la communauté des astronomes du monde entier et on peut dire qu'il a un rôle très important au sein de l'IVOA. Il participe activement à ses travaux.

L'organisation au sein de l'IVOA se fait par groupe de travail, chacun constitué de membres de plusieurs observatoires différents dont le but est de discuter des standards à mettre en place. Le CDS joue un rôle primordial car il participe à la plupart de ces groupes de travail et en dirige certains :

- Sebastien Derriere pour le groupe de travail sémantique qui s'occupe notamment de l'élaboration et la mise en place des UCD.
- Mireille Louys pour le groupe « Modèle de Données » qui se concentre sur les relations logiques entre metadonnées et qui modélise comment les astronomes retrouvent, analysent et interprètent les données astronomiques.
- Francois Ochsenbein pour le groupe de travail VOTable qui comme son nom l'indique travail sur le standard VOTable que j'ai utilisé tout au long du stage.

II - Le Portail du CDS

Le portail est une interface Web pour les services du CDS en cours de développement depuis un an et demi environ. Dans un contexte de développement grandissant des Applications Internet Riches, il est développé en GWT (Google Web Toolkit) qui est une technologie Google que nous détaillons plus loin.

1. Objectif du portail

Le but premier du portail est d'offrir une interface visuellement propre et homogène à l'utilisateur, pour lui permettre d'accéder rapidement aux informations souhaitées. Sa fonction n'est donc pas de remplacer des services déjà en place, mais bien pour les exploiter tout en modernisant leur apparence et leur fonctionnement. La première étape qui, par ailleurs, est la trame de mon travail et m'est restée en tête tout au long du stage, est de ne pas surcharger une page d'informations si cela n'est pas nécessaire. Comme on le voit avec Simbad et Vizier, la quantité importante de fonctionnalités nous permet très facilement de tomber dans l'excès. Vient ensuite l'homogénéité, qui est autant visuelle que fonctionnelle puisqu'une seule recherche sur le portail permet d'interroger à la fois Simbad, Vizier et Aladin. Élément très important lorsque l'on sait qu'une partie des utilisateurs ont du mal à différencier les deux.

Enfin, le portail permettra peut-être même à un utilisateur néophyte d'utiliser des services qui sont actuellement réservés aux initiés.

2. GWT (Google Web Toolkit)

1. Présentation

Google Web Toolkit est un framework créé par Google qui permet de construire des applications Web en Java (similaire à Swing) tout en ayant un résultat en HTML et Javascript. Le fonctionnement, d'une apparente simplicité, est très complexe, car il convertit le Java que l'on code soi-même en Javascript. Ceci permet entre autres d'éviter de programmer en Javascript qui est assez difficile à maintenir, et possède de nombreuses interprétations différentes dans les navigateurs Internet Explorer, Mozilla Firefox, Opera...

2. Un framework performant

La force de GWT réside dans sa capacité à générer du Javascript quel que soit le navigateur et sa version. De manière générale, GWT est purement et simplement de l'AJAX. Ce n'est pas une technologie à proprement parler, c'est un ensemble de technologies. Il est composée d'XML et JavaScript pour répondre aux standards actuellement utilisées par les navigateurs Web et utilise Java comme langage de programmation pour sa simplicité et pour toucher l'ensemble de la communauté des développeurs.

Cependant, GWT ajoute des notions de performance, en retirant toute fonction inutile du code puis en réduisant sensiblement la taille du Javascript généré par compression de celui ci. Son mécanisme de RPC permet de transmettre les objets réduisant ainsi le volume des échanges d'informations.

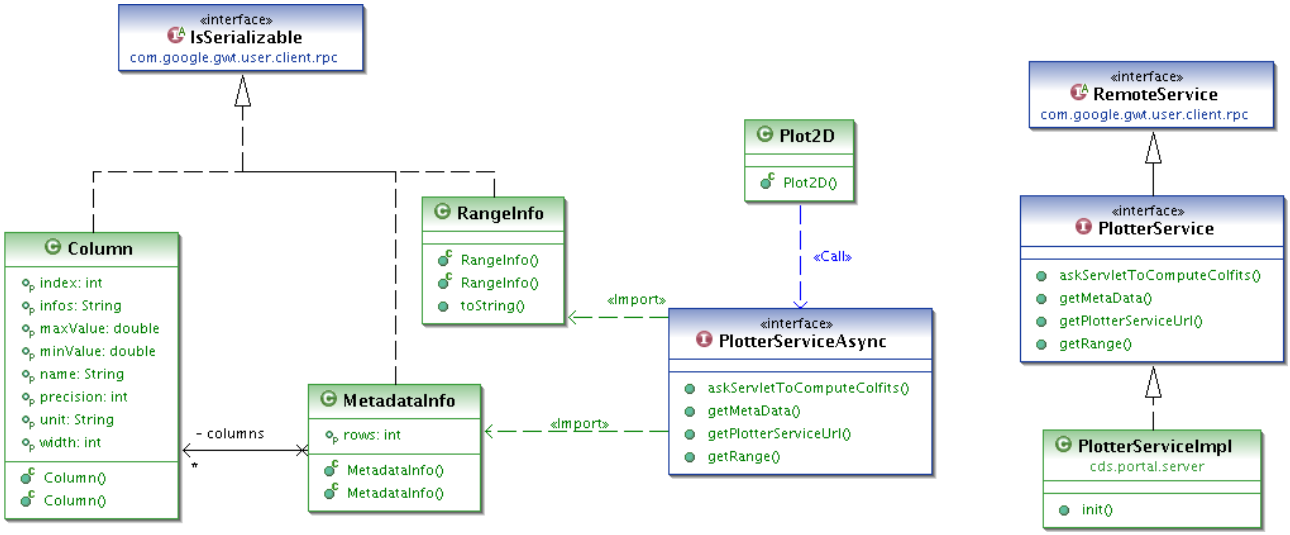


Figure 4 - Diagramme de classe du Mécanisme RPC de GWT

Le système de RPC de GWT requiert que tout les objets transférés héritent de l'interface **IsSerializable** comme il est visible sur la figure 4. La classe **Plot2D** dans notre exemple créera un objet implémentant **PlotterServiceAsync** pour communiquer avec un objet dont le nom se compose comme suit : nom_du_service**Impl.** Celui-ci implémente nécessairement une interface héritant de **RemoteService** nommée nom_du_service.

La simplification des échanges asynchrones est un atout important. Tout en utilisant les navigateurs actuels, cela permet de faire aisément des applications client-serveur. Ce qui aurait été bien plus difficile en PHP + JavaScript

Mais l'un des atouts majeurs de GWT est d'être libre et populaire, ce qui permet de trouver aisément de nombreux outils fonctionnels ou purement graphiques pour développer ses propres applications, ce que nous ne manquons pas de faire pour le portail.

Enfin, lorsque l'on demande à Thomas Boch pourquoi le choix s'est porté sur GWT plutôt

que sur un autre framework, il répond que premièrement la compatibilité est essentielle vers tous les navigateurs, deuxièmement que maintenir du Javascript est bien trop lourd pour être géré à la main, et enfin que la technologie a besoin d'avoir un avenir pour maintenir et développer une application aussi importante que le portail, ce qui semble être le cas à en juger par :

- sa popularité,
- les nombreuses API,
- les mise à jours du framework,
- La simplicité d'utilisation : programmation, test et déboguage en Java puis déploiement à la compilation.

3. Structure générale

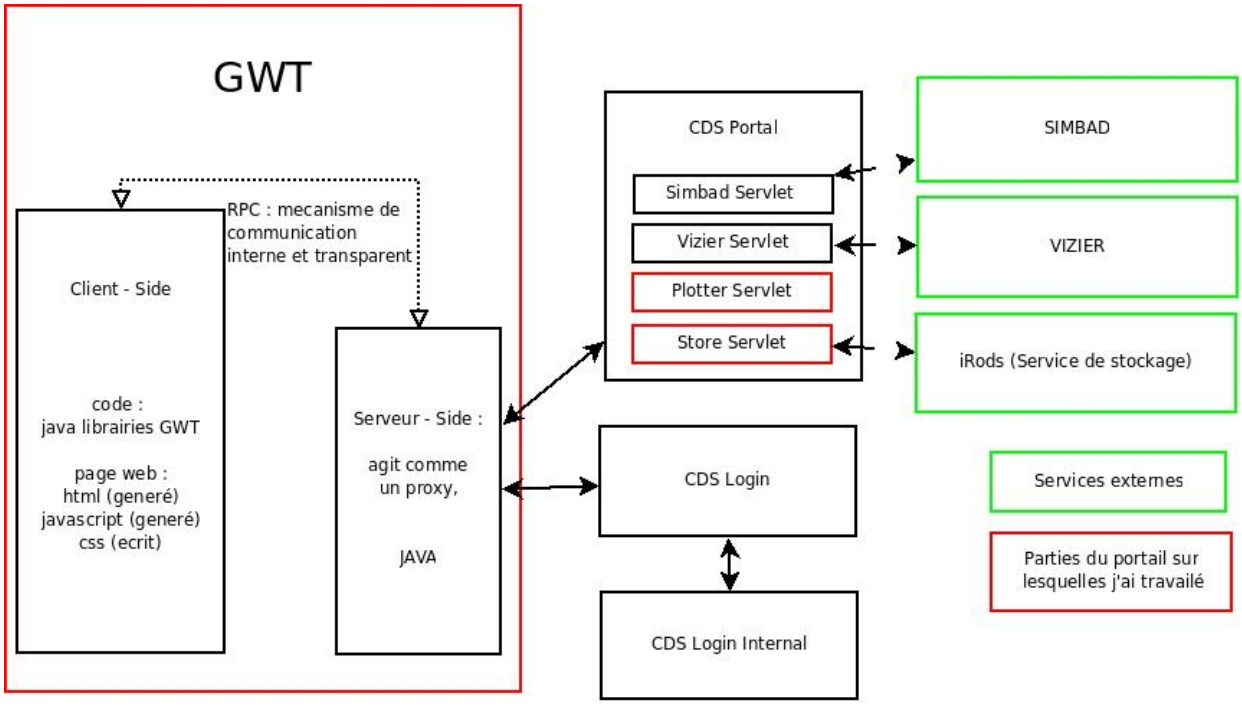


Figure 5 - Diagramme des éléments constituant le portail

Le portail est constitué de différents éléments indépendants et interopérants qui utilisent différentes technologies, ce qui rend son architecture assez complexe. La plupart des communications client-serveur sont internes à GWT alors que les traitements sont effectués par CDS Login, CDS LoginInternal et CDS Portal (cf. Figure 5). Ces services traitent l'information des services externes tels que Simbad ou VizieR et les transmettent à la partie GWT du portail. La partie serveur de GWT fait donc office de proxy pour des raisons d'accessibilité aux services par d'autres moyens que GWT si les choix technologiques futurs venaient à changer. Le stockage des données est géré par iRODS (integrated Rule Oriented Data System), qui est une virtualisation du systeme de stockage ce qui le rend flexible pour écrire et accéder aux données. C'est un système

récent qui permet de stocker des données hétérogènes tout en conservant de bonnes performances et une simplicité d'utilisation.

4. Les développements

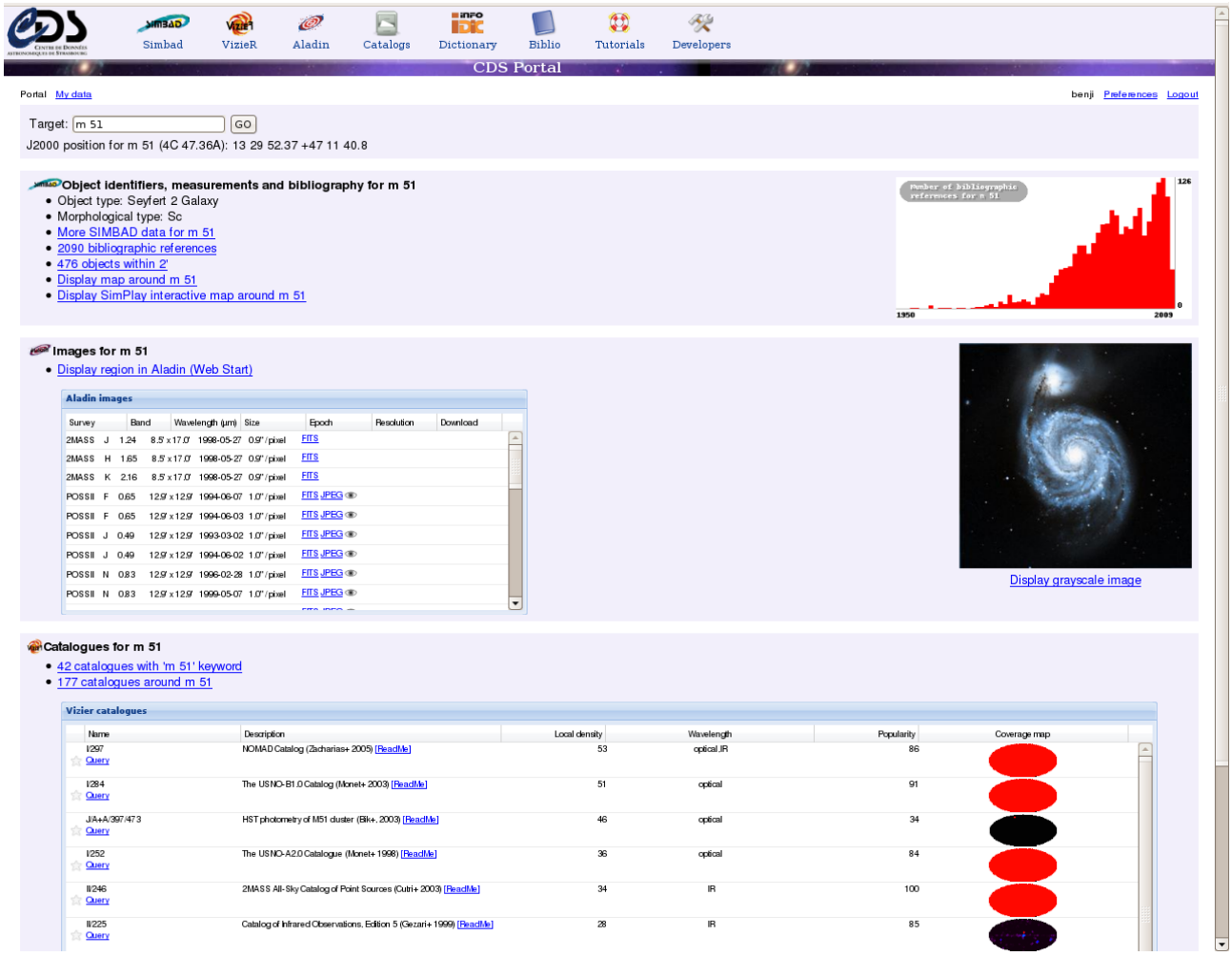


Figure 6 - Recherche classique sur la page d'accueil du portail

Le portail est actuellement fonctionnel, il permet d'interroger Simbad, VizieR et Aladin simultanément, d'obtenir des données statistiques sur les références de l'objet dans Simbad, d'avoir un aperçu des images disponibles avec Aladin et même de générer une image couleur lorsque cela est possible. Enfin, il fournit la liste des catalogues de Vizier possédant des informations sur l'objet recherché ainsi que des statistiques sur le catalogue qui informent l'utilisateur sur la popularité, la densité d'objets pour la zone demandée, etc...

La seconde partie importante du portail est le compte utilisateur, qui permet à l'utilisateur de travailler sur ces données directement en ligne. Un compte fournit 500 Mo d'espace disque,

cependant si l'utilisateur ne possède pas de compte il peut néanmoins stocker jusqu'à 100 Mo de données et sera identifié par un cookie ce qui ne lui permettra pas d'utiliser ces données depuis un autre poste de travail.

Actuellement, l'utilisateur peut télécharger ou envoyer des tables depuis / vers son compte au format VOTable. Il peut aussi interroger Simbad et Vizier avec les données présentes dans les tables sélectionnées ou bien encore supprimer une ou plusieurs tables.

5. Les modifications que j'ai apportées

Les modifications effectuées pendant mon stage sont accessibles depuis cette page et concernent :

- La possibilité de télécharger la table dans un autre format que VOTable
- L'accès à des outils statistiques graphiques (le lien « Plot ! » du tableau de la Figure 7)

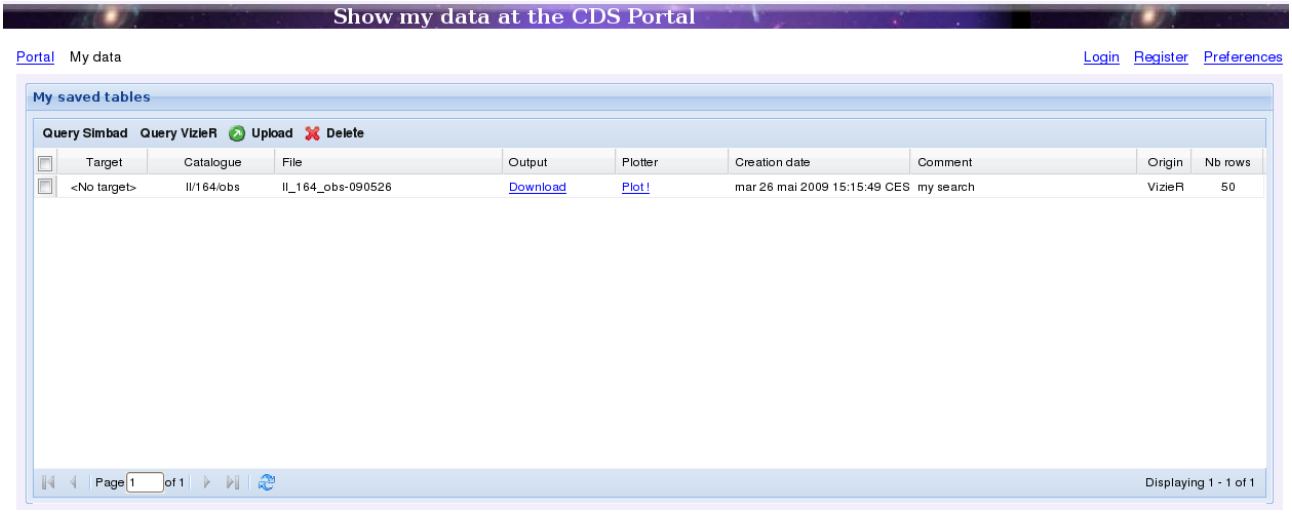


Figure 7 - Liste des tables de l'utilisateur

Je présenterai en premier le convertisseur de VOTable que j'ai réalisé au début de mon stage. Il s'agit d'un petit projet qui m'a permis de me familiariser avec l'environnement de travail mais qui comporte néanmoins quelques difficultés intéressantes à résoudre.

Je présenterai par la suite la partie la plus longue de mon stage qui fut le « Plotter ». Cette partie fut compliquée sur différents points à cause de l'architecture complexe des projets et des contraintes à respecter (cf. diagramme Figure 5). Il s'agit de fournir à un outil déjà existant une interface graphique dans une architecture client-serveur.

Avant de conclure, je présenterai brièvement les travaux qui suivront mon stage.

III – Déroulement du stage

1. Découverte de l'environnement de travail

1. Stage de Cédric CAPOULUN

Au début du stage j'ai eu l'occasion de voir le travail effectué par un ancien stagiaire de l'IUT, Cédric Capoulun (Licence pro CISII 2007/2008) qui a fait un état de l'art très intéressant sur les RIA et sur les possibilités offertes par différentes technologies autour du Web 2.0. C'est dans la continuité de son stage que le portail a été conçu et donc à partir de l'étude des technologies qu'il a effectué.

Face à la découverte d'un langage, il est toujours agréable de savoir que celui-ci a été jugé positivement par une personne ayant les mêmes connaissances que soi. C'est donc avec le sentiment de continuer dans une voie ouverte par un autre que j'ai commencé mon stage tout en découvrant l'environnement de travail de l'observatoire.

2. Normes et standards

Pendant les premières semaines, j'ai également dû m'habituer aux normes et aux standards utilisés par le CDS pour la conception et le développement des projets sur lesquels j'ai travaillé. L'IVOA (International Virtual Observatory Alliance) définit les standards tel que VOTable qui est un format XML pour décrire les données tabulaires en astronomie. Dans ces tables, les métadonnées (données qui décrivent des données) sont décrites en début de table et sont suivies par les données. C'est un format complexe comparé aux autres mais qui est largement utilisé malgré son jeune âge. Un standard de l'observatoire virtuel aussi très important est l'UCD, Unified Content Descriptors, qui permet de décrire des quantités astronomiques. L'UCD est en réalité plus une balise sémantique qu'un outil, mais il a pour but de le devenir. On le retrouve notamment dans les métadonnées d'un fichier VOTable.

En dehors de l'IVOA, le TSV (Tab Separated Values) et le CSV (Comma Separated Values) qui présentent les données respectivement par des tabulations et des virgules sont des formas que j'ai également utilisés.

3. Premier projet, convertisseur VOTable => TSV, CSV, ...

Mon premier projet consistait à permettre aux utilisateurs du portail de télécharger des tables dans d'autres formats que VOTable, par exemple CSV ou TSV. Cette étape intervient donc entre le moment où l'utilisateur demande le téléchargement de la table depuis le service de stockage iRODS et la réception de celle-ci. Il faut donc parser le fichier, le convertir puis intégrer cette option au service de téléchargement du portail.

1. STIL : parser de VOTable

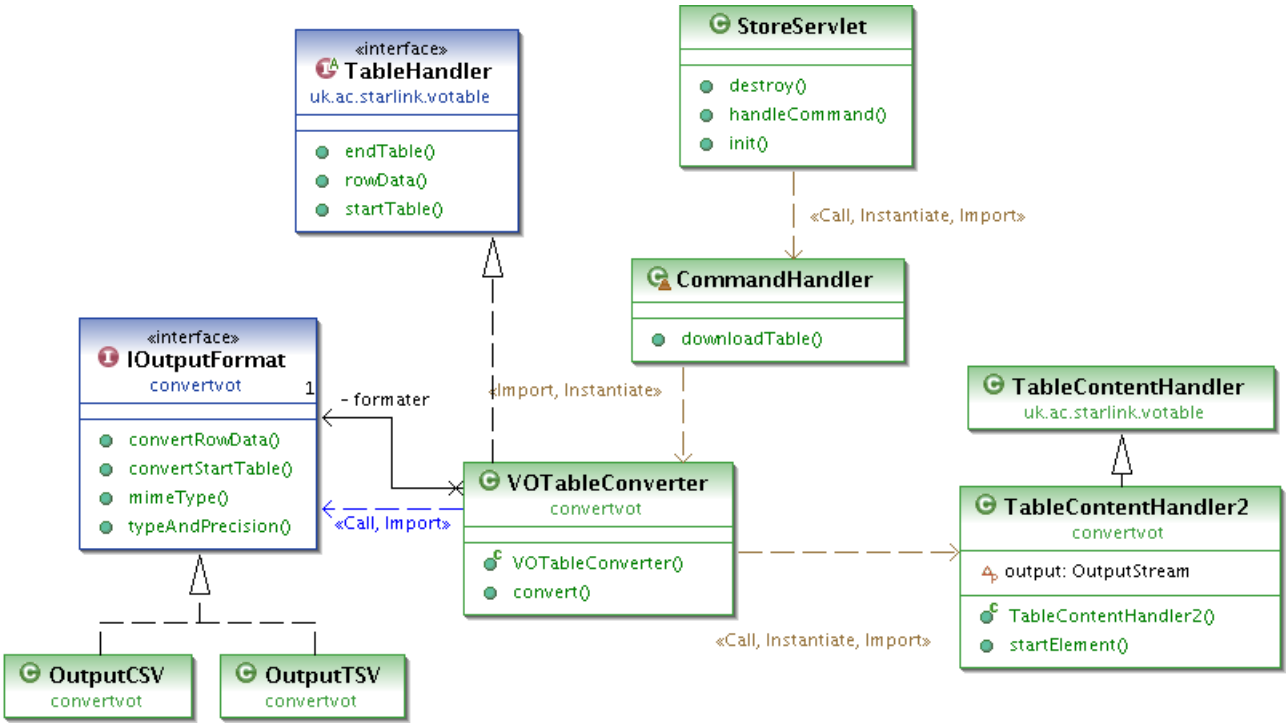


Figure 8 - Schéma de classes du convertisseur de VOTable

Tout d'abord, pour convertir le fichier, nous le parsons à l'aide d'une librairie mise au point par Mark Taylor, travaillant à l'université de Bristol en Angleterre. Cette librairie s'appelle STIL pour Starlink Tables Infrastructure Library, est libre et possède entre autres cette fonctionnalité. Nous utilisons les classes et interfaces référencées sur le schéma par leur appartenance au package **uk.ac.starlink.votable.** Le StoreServlet, déjà existant, utilise VOTableConverter par le biais de CommandHandler son gestionnaire de commande.

VOTable étant un formalisme XML, deux méthodes sont possibles pour parser les fichiers. La première, le DOM (Document Object Model), consiste à lire l'intégralité de la table pour pouvoir la manipuler sous la forme d'un objet java. Ceci s'avère intéressant pour des tables de petite taille

mais dans notre cas, la table peut atteindre plusieurs centaines de milliers de lignes, ce qui poserait de sérieux problèmes de mémoire que nous ne pouvons pas nous permettre. Nous avons donc opté pour la méthode SAX qui consiste à analyser le fichier séquentiellement. Cette méthode permet également de traiter entièrement le fichier sous la forme d'un flux. Nous n'avons donc pas besoin de stockage temporaire mais juste d'un buffer en mémoire pour permettre de traiter le fichier.

Sur le graphique Figure 9 et d'après les équations des régressions linéaires de nos mesures, on remarque que parser de gros fichier en DOM (◆) demande 2,5 fois plus d'espace mémoire que le poids du fichier lui même (■). Ceci n'est pas acceptable au vue de la taille que peuvent atteindre les fichiers traités. La méthode SAX (▽) quant à elle traite le fichier comme un flux et n'utilisera que très peu de mémoire. Même dans des cas extrêmes, on pourra donc parser des fichiers de plusieurs centaines de milliers de lignes dans des temps similaires sans risquer de problèmes de mémoires.

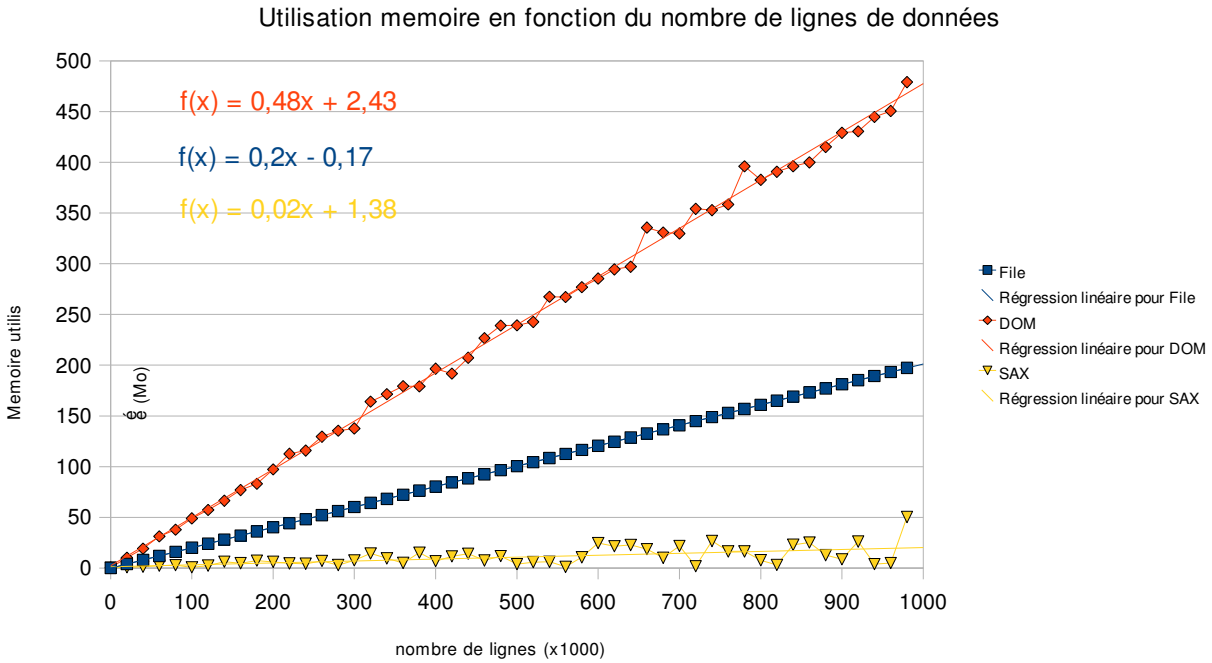
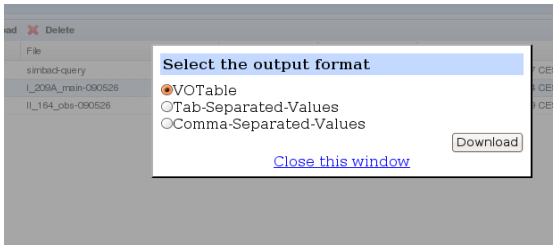


Figure 9 - Statistiques justifiant l'utilisation de la méthode SAX pour parser les VOTable

STIL fonctionne de manière événementielle, c'est à dire qu'à la lecture de certains éléments (Objet **VOTableConverter**) , il renvoie un événement nous permettant de traduire la ligne dans le format voulu. Les événements reçus sont de trois types, **startTable** qui renvoie l'ensemble des métadonnées de la table, **rowData** qui renvoie un tableau des valeurs à chaque ligne et enfin **endTable** qui indique que nous arrivons en fin de table mais que nous ignorons ici car aucun traitement ne lui est lié ici. Afin de laisser la possibilité d'ajouter de nouveaux formats de sortie, le parsing est fait de manière générique et un objet implémentant l'interface **IOutputFormat** s'occupe de réécrire les données de manière à respecter les standards souhaités.

2. Intégration du convertisseur au service de téléchargement



Vient ensuite l'intégration au service de téléchargement qui s'effectue par l'ajout d'une colonne dans la liste des tables de l'espace personnel de l'utilisateur, avec le lien « download » ouvrant une petite fenêtre pour choisir le format et retirant à l'utilisateur la possibilité de cliquer dans la fenêtre principale. Il faut ensuite indiquer au service le format désiré en ajoutant un paramètre à l'URL.

3. Envoi de mails en anglais au concepteur de STIL sur des cas non traités

Nous avons, afin d'éviter de parser l'intégralité du document, utilisé STIL qui s'avère être un très bon choix mais nous sommes par conséquent dépendants de son fonctionnement... De nombreux champs des métadonnées, entre autres, ne sont pas récupérables directement par cette méthode pour des raisons que nous ignorons**.** J'ai donc envoyé un mail à Mark Taylor lui demandant s'il existait un moyen de récupérer ces données. Lequel m'a répondu aussitôt que la classe utilisée ne permettait pas, en effet, de récupérer ces données telles-quelles mais que la librairie étant basée sur SAX, en ré-implémentant des classes sous-jacentes, cela devait être possible. Ce que j'ai fait aussitôt pour récupérer les informations voulues. Sur le diagramme de classe Figure 8, l'objet **TableContentHandler2** héritant de **TableContentHandler** nous permet de récupérer les champs de métadonnées souhaités. Cependant, un deuxième problème s'est posé car l'un des champs habituellement numérique peut parfois contenir une astérisque (ex : « 15* »), indiquant qu'il s'agit alors que le champs peut contenir jusqu'à 15 caractères. Un deuxième mail s'est alors imposé mais qui, malgré la réponse, n’a pas pu donner suite à une solution, considérant ce cas négligeable face à la quantité de classes à réecrire pour récupérer la valeur littérale du champs.

2. Le Plotter de données

1. Présentation

Je présente ici la partie principale de mon stage. Elle m’a demandé plus de recherche au niveau de GWT et de nombreux retours en arrière dans le développement pour des choix qui à première vue paraissaient judicieux mais ne l'étaient pas toujours.

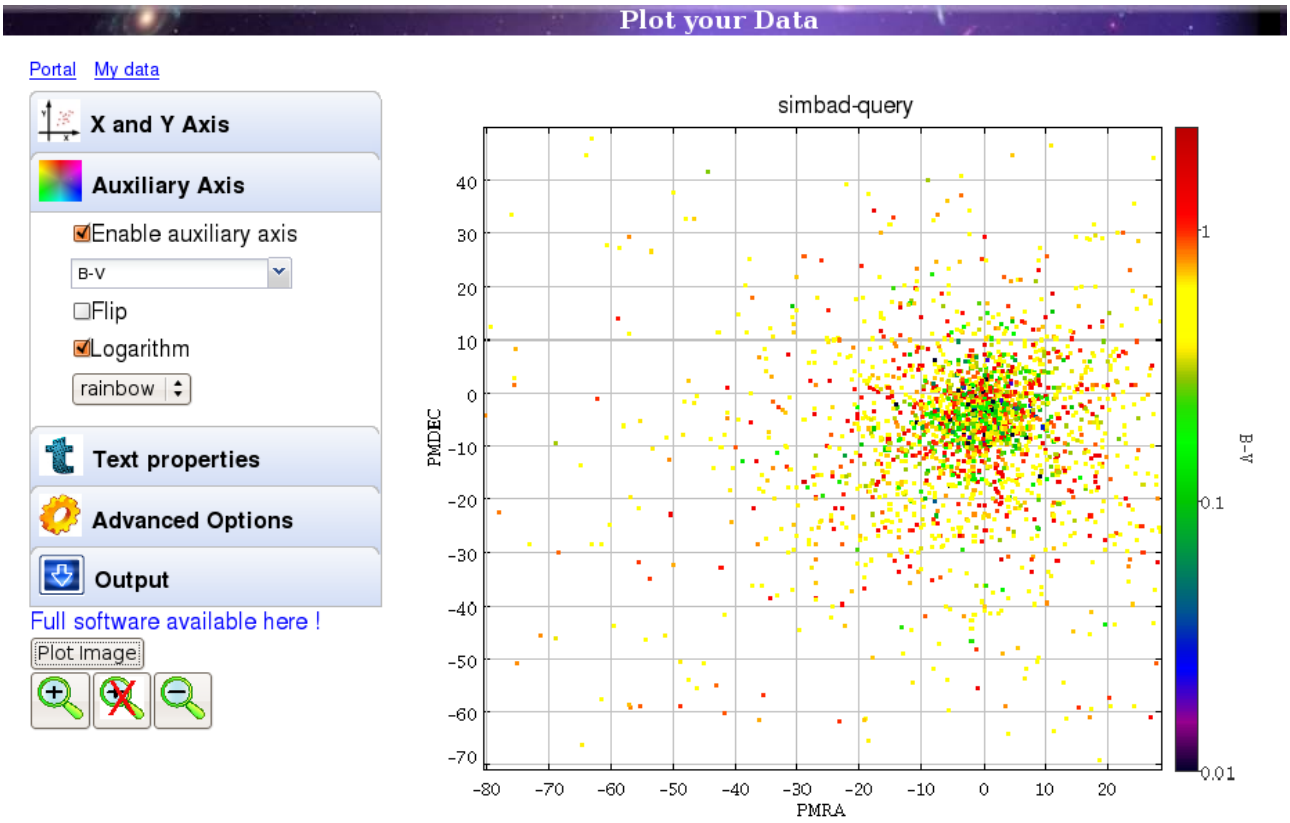


Figure 10 - Aperçu du « plotter » de données

1. Objectif du plot

Le plotter est un outil graphique permettant, en dessinant des points sur différents axes, d'obtenir par exemple l'allure d'une courbe statistique. Habituellement on « plot » des mesures pour les comparer à des courbes théoriques. A l'observatoire, j'ai eu l'occasion de discuter avec des astronomes et d'assister à des présentations sur les recherches qu'ils mènent. Le « plot » fait partie des techniques de base de l'astronome pour observer et comprendre. Partant de ce fait il n'est pas inutile d'intégrer ce genre d'outil à des services qui contiennent les données pour travailler avec sans manipuler les fichiers.

Dans notre cas, l'objectif visé par le « plot » est d'abord de pouvoir visualiser les données stockées par le portail directement en ligne, sans avoir à télécharger la table. Le second objectif est d'offrir un aperçu du programme qui est lui aussi basé sur STIL en lui donner une interface graphique par le biais de GWT. Enfin de fournir, dans l'idée de l'observatoire virtuel, des outils accessibles.

2. STILToolSet, des outils pour STIL

Le logiciel utilisé pour le plot est STILTS, pour STIL ToolSet qui comme son nom l'indique est un ensemble d'outils pour STIL. Le programme s'utilise en ligne de commande, nous effectuons donc le « plot » coté serveur grâce à la servlet qui renvoie ensuite l'image.

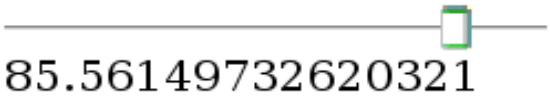
2. Interface graphique

1. Le StackPanel

N'ayant pas de véritable contrainte graphique à respecter, j'ai pu choisir une apparence appropriée pour les différentes options du formulaire. C'est à dire ni trop chargé ni trop dépouillé, ce qui n'est pas toujours chose facile. Ceci m'a amené à voir les différentes librairies annexes de GWT pour finalement choisir un design appartenant aux sources natives de GWT à savoir le StackPanel. J'ai donc implémenté ce panel qui est une imbrication de panels les uns dans les autres pour obtenir plusieurs panels accessibles en cliquant sur leurs entêtes. L'imbrication reste pourtant un problème pour récupérer les valeurs du formulaire que nous verrons par la suite. Ceci nous permet de cacher de nombreux champs de formulaire qui ne sont pas nécessaires tout en permettant d'alléger considérablement la page.

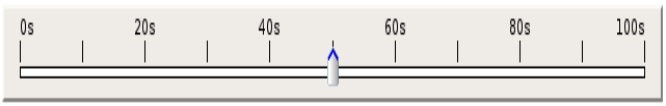
2. Le Curseur

Un des éléments les plus longs à mettre en place fut la barre permettant de sélectionner le taux de transparence des points du plot. Cette fonction étant essentielle, chaque problème a du être solutionné malgré les choix que cela impliquait. Premièrement, l'API du framework ne possède pas, à mon grand étonnement de curseur, il a donc fallu en trouver un qui soit correct :



Ce premier curseur, trouvé sur EXT-JS est seulement composé de Javascript « natif » donc pas réellement GWT.

Ce second, développé par net-Sphene, est graphiquement intéressant car les couleurs de la barre suivent le curseur.



Celui-ci, d'une librairie officielle mais externe de Google, apparaît un peu trop classique et un peut trop présent car le bouton sur lequel il est dessine ne s'enlève pas aisément.

Le choix s'est donc porté sur le second qui possède de nombreuses possibilités en plus d'être visuellement propre. Pendant l'implémentation, nous nous sommes aperçus que STILTS calculait la transparence de sorte que la valeur n corresponde au nombre de points qu'il faut superposer pour obtenir un point opaque. Ce qui indique que la limite n'est autre que le nombre de points du graphique. Ceci n'est pas un problème en soi mais imaginons que le fichier traité contienne une centaine de milliers de valeurs, cela impliquerait une barre de transparence entre 1 et 100 000 pour environ 200 pixels à l'écran soit environ un décalage de 500 pour chaque pixel, ce qui rend les points absolument invisibles !

Pour remédier à ce problème, la solution la plus simple était de baser l'échelle du curseur sur une échelle logarithmique par exemple. Chose impossible avec le curseur net-Sphène qui pour des raisons incompréhensibles n'exécute pas certaines fonctions. Nous avons donc finalement opté pour le curseur de l'API externe de Google qui permet le changement d'échelle pour :

- les valeurs en fonction de la position,
- la position du curseur sur la barre,
- les traits indiquant que l'échelle est logarithmique.

Voici la version finale retenue. On constate que seule la valeur courante est affichée et que l'échelle est logarithmique car ici, la valeur maximale correspond a 4800 lignes environ.

3. Les formulaires

Comme indiqué précédemment, le formulaire est situé sur différents panels, or le bouton d'envoi du formulaire n'est pas destiné à une méthode POST (qui transmet automatiquement les valeurs des éléments de formulaire). Ici, une méthode POST nous priverait de faire une requête asynchrone. Il faut donc récupérer les champs et leurs valeurs, les ajouter aux paramètres de l'URL puis remplacer l'URL de l'image par la requête ainsi construite.

L'utilisation d'une fonction de recherche récursive, s'est révélée indispensable pour récupérer l'intégralité des paramètres qui peuvent être au premier ou à plusieurs niveaux d'imbrications. De plus, les perpétuelles modifications d'apparence, de structure et autres, nous obligeraient trop souvent à vérifier que chaque paramètre est correctement récupéré.

Chaque élément de formulaire, pour être reconnaissable, implémente l'interface MyForm qui possède une fonction pour récupérer ses paramètres. Les listes fermées sont généralement de type « select » comme dans n'importe quel autre document html. Cependant pour répondre à certains besoins, l'application nécessite des listes dont l'expression est libre en plus de fournir diverses possibilités. C'est par exemple le cas pour les paramètres des axes qui peuvent être constitués d'une seule série de données ou de la combinaison de plusieurs.

1.Le zoom

La seconde difficulté du projet est la fonction de zoom sur l'image de plot. L'idée est de pouvoir faire un zoom dans la partie client sur l'image qui renvoie à la partie serveur une requête contenant les bornes inférieures et supérieures voulues. Le serveur refait une nouvelle image, ce qui permet un zoom tout en ayant un repère qui s'adapte aux valeurs.

Il faut donc pouvoir dessiner une fenêtre de zoom sur l'image et pouvoir la redimensionner, l'ajuster. Il existe des canvas qui permettent d'avoir un repère au dessus d'une image pour dessiner. Cependant sur trois canvas testés et/ou installés, un seul a pu être implémenté et ce après de nombreuses difficultés, mais des problèmes persistaient. Il semblerait que le canvas, qui n'est pas spécifique à GWT, n'ait pas été prévu pour un changement dynamique d'image, ce qui m'a amené à rechercher une fois encore un autre moyen de faire cela.

Thomas Boch a alors trouvé une librairie en Javascript pur qui convenait parfaitement à ce que nous souhaitions : la librairie permet de créer le cadre de zoom de manière totalement dynamique ce qui devrait solutionner le problème précédant. Une des possibilité de GWT explicitée au dessus est d'utiliser du Javascript natif en créant des fonctions Java particulières. Ce mécanisme s'appelle JSNI (JavaScript Native Interface) et permet de communiquer de GWT vers Javascript, de Javascript vers GWT etc

```
// Java method declaration...
native String JSNIfunction() /*-{

 // ...implemented with JavaScript
 var res = ma_variable;
 return res;

}-*/;
```

Suite à la réception de la première image, pour activer le zoom il faut récupérer les valeur minimales et maximales des axes X et Y pour pouvoir calculer la correspondance en valeur de la fenêtre de zoom qui est récupérée en pixel. Pour cela nous interrogerons GWT par le mécanisme de RPC afin de récupérer un objet.

Pour les zooms suivants, les valeurs calculées par le client restent en mémoire ce qui évite de refaire un appel asynchrone.

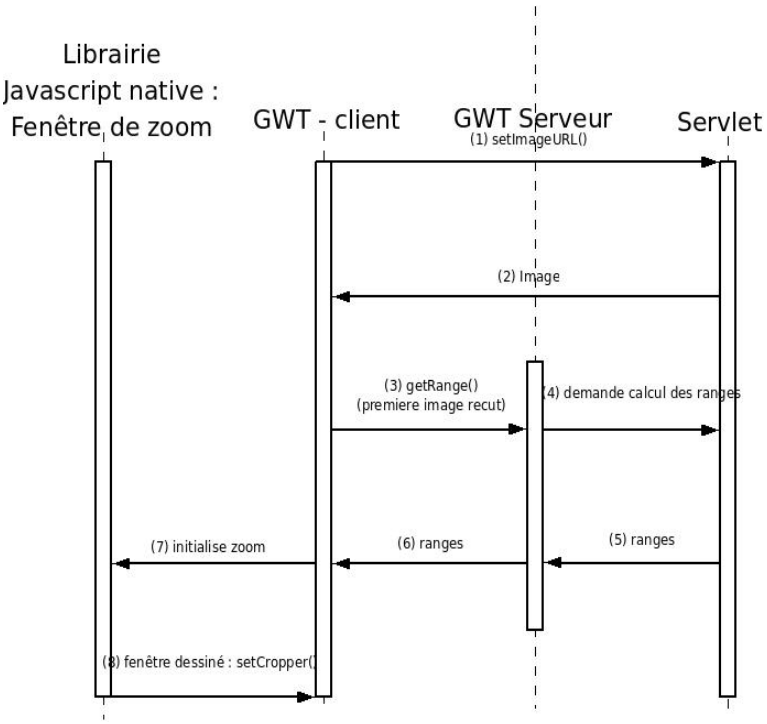


Figure 11 - diagramme de séquence du zoom

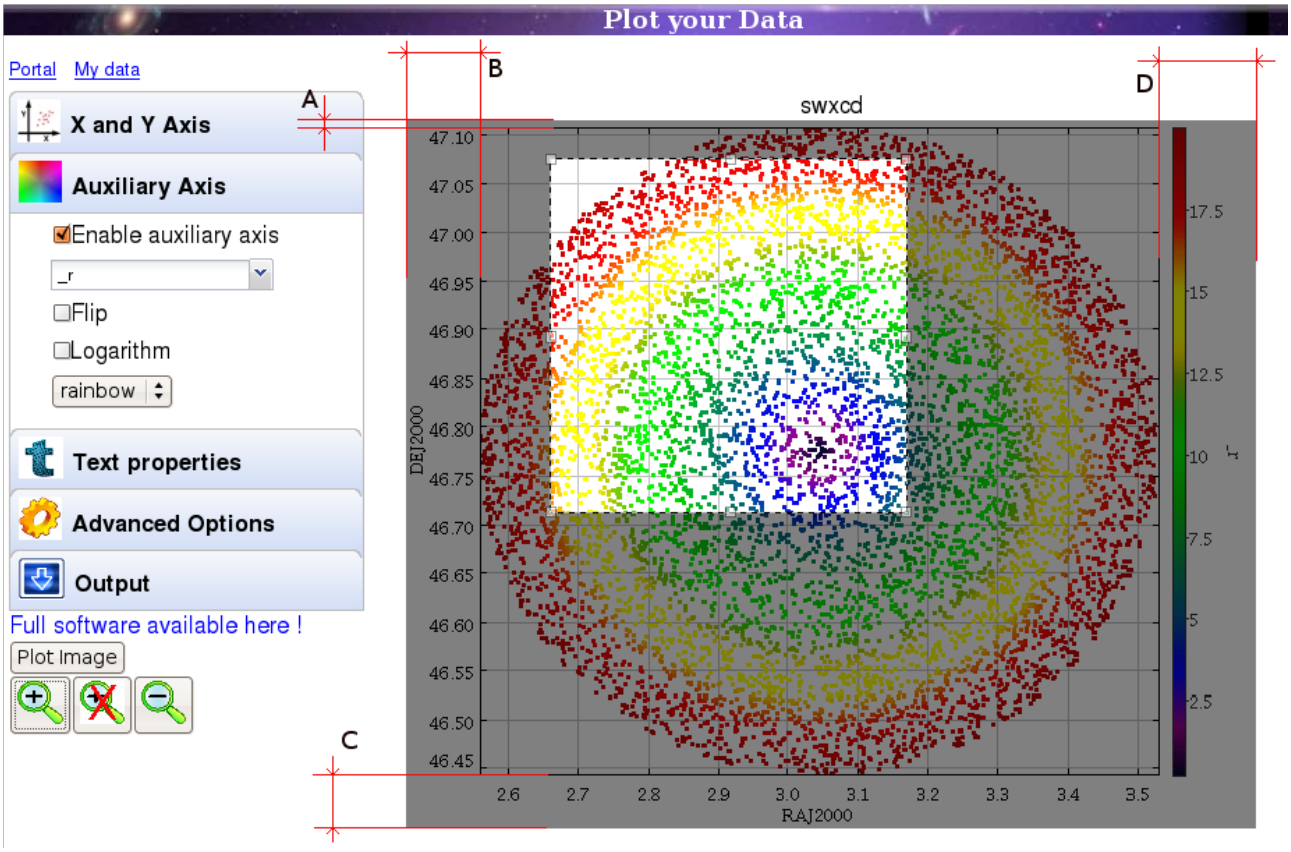


Figure 12 - Aperçu de la fenêtre de zoom contenant les marges à définir.

Une fois la librairie Javascript mise en place, la fenêtre de zoom fonctionnelle, il fallait récupérer les origines du repère dans l'image. Celles-ci varient selon la taille du texte en légende, du nombre de caractères maximal des valeurs affichées sur l'axe des ordonnées, ainsi que la présence ou non d'un axe auxiliaire sur le schéma. Il s'agit de connaître les valeurs de A, B, C et D (cf. figure 12).

Pour cela, j'ai profité de la présence de Mark Taylor à Strasbourg à l'occasion de la réunion de l'IVOA se tenant du 24 au 29 mai 2009 pour lui poser les questions nécessaires à la continuité de mon travail. Cependant, pour pouvoir présenter la fonctionnalité de ce zoom durant le meeting, j'ai temporairement choisi des valeurs arbitraires.

3. La servlet

Coté serveur, c'est la servlet qui exécute les tâches et non GWT qui ne fait que la liaison entre le navigateur du client et les services du CDS. Ce choix permet l'utilisation des services dans d'autres contextes ou de permettre à d'autres programmes d'accéder aux mêmes possibilités.

1. Création de colfits

Le temps de calcul sur les gros fichiers étant parfois conséquent (plusieurs dizaines de secondes), il s'agit alors de convertir le fichier VOTable au format colfits qui permet alors une lecture plus rapide.

Imaginons que dans un fichier VOTable (basé sur XML donc textuel), seules les colonnes VTmag et pmRA nous intéressent, le programme devra lire le fichier intégralement pour obtenir les informations contenues à chaques lignes.

	VTmag	BTmag	...	pmRA	
<TR>	<TD>5</TD>	<TD>65.452</TD>	...	<TD>89.756</TD>	</TR>
<TR>	<TD>5</TD>	<TD>65.201</TD>	...	<TD>60.589</TD>	</TR>
<TR>	<TD>7</TD>	<TD>64.536</TD>	...	<TD>72.548</TD>	</TR>
<TR>	<TD>8</TD>	<TD>63.589</TD>	...	<TD>68.456</TD>	</TR>

Figure 13 - Représentation des données dans un VOTable

Dans un fichier colfits, qui lui est organisé en colonnes, il suffit au programme, arrivé sur BTmag, d'ignorer la portion du fichier relative à ce champ et de passer à la suivante sans avoir à lire les données relatives à BTmag ce qui permet une lecture des seules informations utiles. Ajoutons à cela que le colfits est un format binaire, ce qui en plus de le rendre plus performant pour le traitement de données, le rend moins volumineux, de moitié environ .

VTmag	5	5	7	8
BTmag	65.452	65.201	64.536	63.589
...	...	...	...	...
pmRA	89.756	60.589	72.548	68.456

Figure 14 - Représentation des données dans un colfits

La conversion s'avère utile à partir du moment où le temps de conversion ajouté au temps de création de l'image est supérieur au temps de création avec un fichier VOTable non binaire. Ce qui est difficilement calculable car on ne peut pas prévoir le nombre de fois que l'utilisateur décidera de plotter une image. Cependant l'analyse du document (Figure 15) nous indique différentes choses :

- On remarque que, sur de petits fichiers (1er graphique), la conversion (■) ajoutée à la création d'image à partir du fichier colfits (◆) est égale ou inférieur à la création d'image à partir du fichier VOTable (▼) . Ce qui signifie que, sur des petits fichiers, la conversion n'allonge en rien le traitement et n'est donc pas un problème.
- Ensuite, sur le deuxième graphique, on remarque que la conversion (■) est à peine plus longue que la génération d'image à partir d'un fichier non converti (▼). Cependant on constate également que la génération d'images à partir de fichiers convertis lui ne croît que très peu et par conséquent une conversion + deux images à partir de colfits est inférieure à

deux image à partir de VOTable.

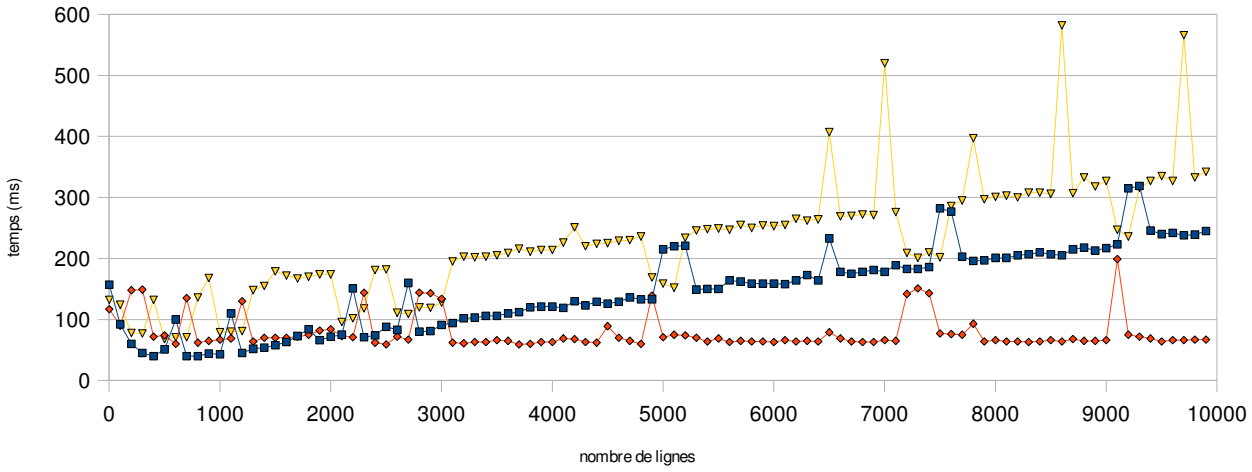
- Enfin on notera qu'un gros fichier, d'environ un million de lignes, met 30 secondes pour être converti.

On en conclut que l'opération de conversion est rentable, au minimum, dès la deuxième image ce qui semble être intéressant puisque l'utilisateur réglera probablement quelques paramètres supplémentaires après le premier aperçu. Enfin, pour justifier cette conversion, je tiens à ajouter que la conversion de colfits est préalablement demandée par le client de manière asynchrone en prévision d'un plot. Donc le temps que l'utilisateur choisisse les paramètres de son plot est bien souvent suffisant à la conversion, ce qui rend son temps de calcul négligeable.

2. Gestion d'un cache

Le colfits, une fois généré, est conservé par un système de cache qui conserve ces fichiers dans un dossier. On attribue une valeur maximale ainsi qu'un ratio à ce cache qui, lorsqu'il est plein, se vide des plus anciens fichiers jusqu'à atteindre ratio * valeur max du cache. Ce qui signifie que le fichier peut persister plusieurs jours si le service n'est pas très utilisé.

3. Création des images



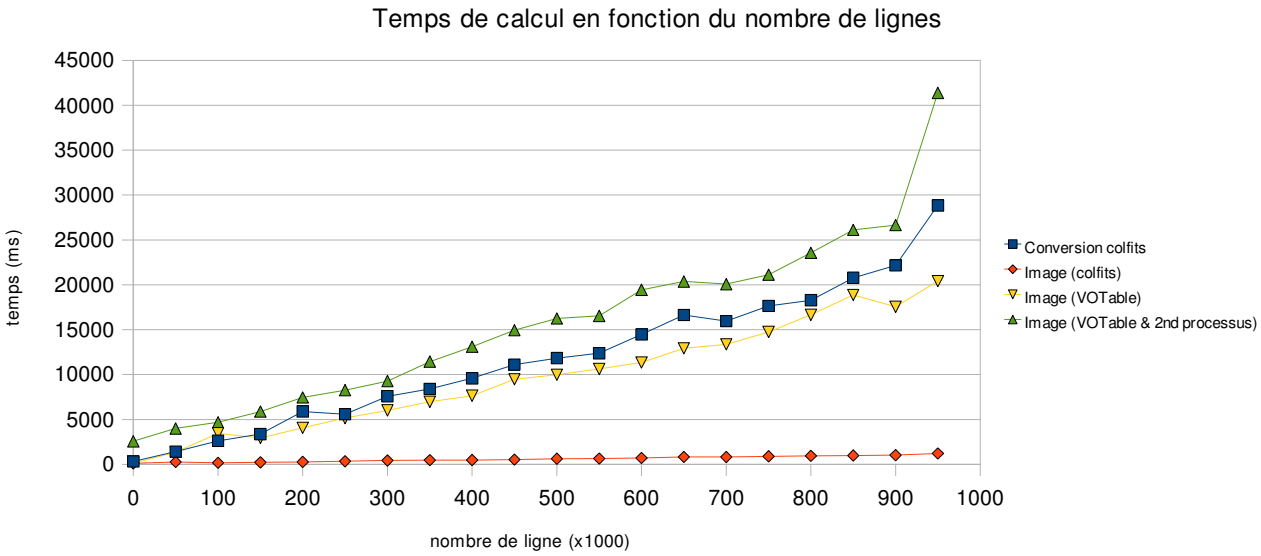


Figure 15 – Statistiques sur le temps de génération d'une image en fonction du nombre de lignes

La création d'image est la seule action que le client déclenche directement sans passer par la servlet car le navigateur gère déjà le chargement d'images de manière asynchrone. Pour générer l'image, on vérifie en premier lieu que le colfits existe par présence ou non dans la liste des fichiers en cours de traitements de la servlet. S'il est présent, on attend jusqu'à ce que le traitement ait été effectué.

Ensuite il faut exécuter la ligne de commande associée après vérification des paramètres. Pour cela j'ai initialement utilisé la commande « exec » de java qui permet de créer un processus mais comme STILTS est écrit en Java, le temps de lancement de la JVM représente une grosse partie du temps d'exécution ce qui est une perte inutile sachant que la servlet est elle même dans une JVM. On peut en effet remarquer sur le second graphique du document Figure 15 que la courbe verte est toujours supérieur d'un minimum de deux seconde aux autres valeurs. Il a donc fallu trouver une voie alternative grâce aux outils contenus dans STILTS. La différence est nette concernant les images, moins concernant la conversion des gros fichiers mais reste visible tout de même.

4. Fonctionnalités asynchrones

Les erreurs liées à la découverte d'une technologie sont toujours déconcertantes au début, mais pas plus que la simplicité avec laquelle on développe par la suit. Un des grands atouts de GWT est la facilité avec laquelle, tout en faisant du Web, on peut utiliser des méthodes de communication asynchrones.

1. Schéma

Les nombreuses fonctionnalités asynchrones permettent non seulement un chargement progressif de la page mais également d'exécuter des traitements en prévision des futures requêtes. Ici au chargement initial, le client va demander une création du colfits, récupérer les propriétés comme l'adresse du servlet d'image et les noms des colonnes utilisables. Ensuite, à la réception de la première image, il demande les bornes de celle-ci pour la fonction de zoom qui peut être demandée par la suite.

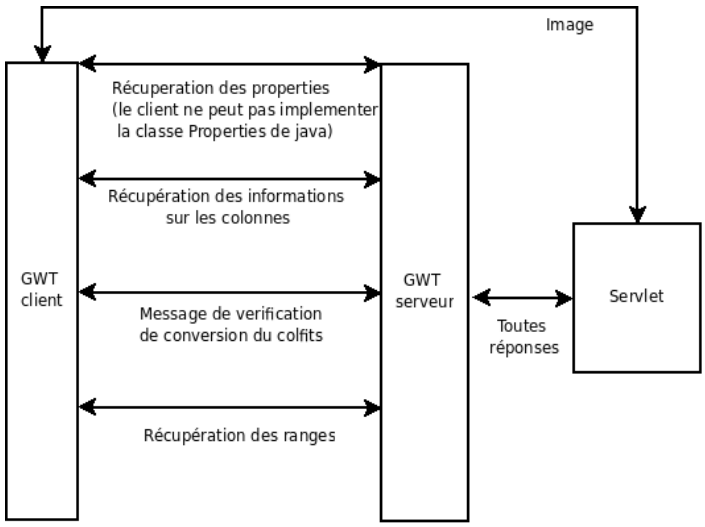


Figure 16 - Résumé des échanges client - serveur

2. Récupération des métadonnées

La récupération des métadonnées, nécessaire pour connaître les noms des colonnes numérique, se fait normalement en parcourant la table. Cependant, les grosses tables peuvent être longues à analyser et cette étape doit être très rapide pour rester transparente à l'utilisateur. Nous avons donc décidé de créer une copie de la table ne contenant que les métadonnées ainsi que le nombre de lignes du fichier réel (utilisé pour la transparence). De cette manière, le fichier à analyser ne devrait pas dépasser une centaine de lignes et le temps d'exécution devrait alors être négligeable.

3. L'Histogramme

Un autre outil très utilisé par les astronomes pour étudier certains phénomènes est l'histogramme. Le diagramme en bâton permet d'avoir une représentation quantitative de certaines mesures. Il est donc important de pouvoir visualiser ce genre d'informations et surtout de pouvoir le comparer à d'autres mesures. J'ai donc implémenté une interface basée sur le même modèle que le « Plotter » qui comporte la plupart des fonctionnalités de ce dernier à quelques différences prêt.

La principale différence avec le « plotter » est que l'histogramme peut prendre plusieurs valeurs en abscisse. Chacune de ces valeurs a différents critères, ile peuvent :

- être pondérés par une autre valeur,

- avoir leur propre couleur, celle-ci peut alors être prise arbitrairement par le programme ou choisie par l'utilisateur dans une liste ou par sa valeur hexadécimale en couleur rouge, vert et bleu RRGGBB,
- être représentés en bâton (BTmag), en sommet (Vtmag), en ligne (B-V), etc...
  les représentations qui ne sont pas en bâton peuvent alors être en pointillées et avoir une largeur plus ou moins importante.

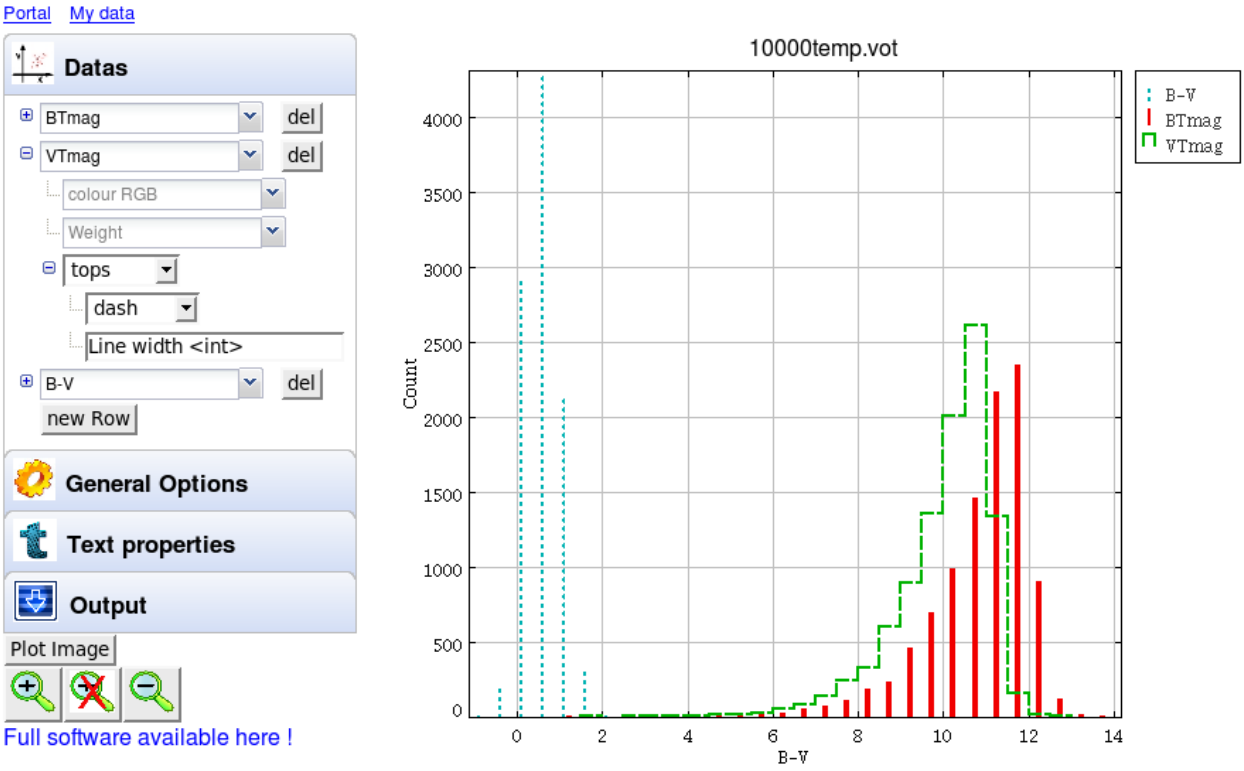


Figure 17 - Aperçu de l'outil « histogramme » en cour de développement

4. Perspectives

1. L'histogramme

L'histogramme est en cours de développement et n'est donc pas encore tout à fait au point.

Il reste la fonction de zoom à définir car celui-ce s'effectue sur plusieurs valeurs strictement positives ce qui le rend différent du « plotter ». Cette fonction devra donc être réimplémenté tout en factorisant ce qui est commun au deux outils. Quelques modifications vont donc suivre.

2. La carte de densité

Afin de permettre aux utilisateurs de voir de quel endroit du ciel les données proviennent, des carte de densité existent. La figure 18 est une représentation aplatie du ciel, un planisphère du ciel en d'autre termes. les zones les plus claires sont les zones que nous connaissons le mieux dans le ciel, c'est à dire celles dont nous possédons le plus de lignes dans le fichier. Cette image a été réalisée avec l'ensemble des objets répertoriés dans Simbad (4 millions) et, est en coordonnées galactiques (dont le plan de référence est la voie lactée). Il en existe d'autres par exemple en coordonnées équatoriales qui utilise l'équateur comme plan de référence.

Cette partie n'a pas encore été commencé mais devrais être développer dans les semaines à venir.

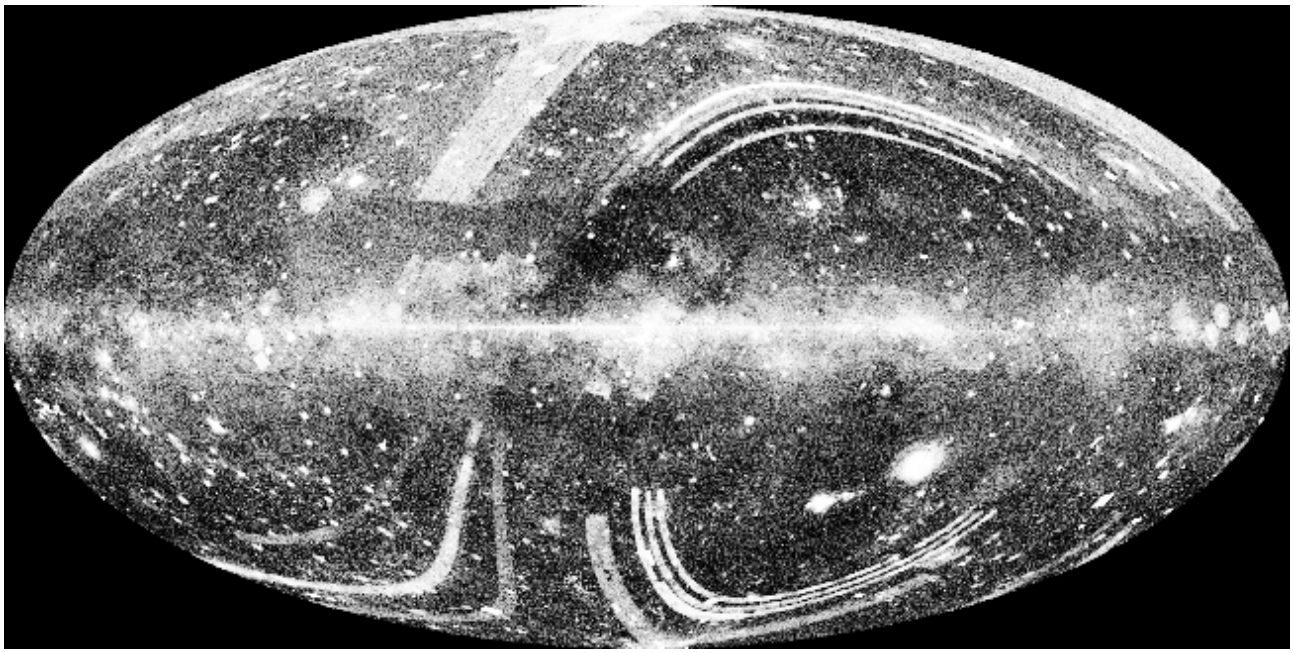


Figure 18 - Carte de densité des objets Simbad en coordonnées galactiques

3. Finalisation

Au moment de la rédaction de ce rapport, le « plotter » reste à finaliser, la fonction de zoom notamment doit prendre en compte les marges à l'interieur de l'image pour en connaître la position exacte. Cela devrais être réalisé par analyse graphique de l'image car Mark Taylor n'est pas en mesure de récupérer cette information aussi aisément que nous le pensions.

L'export d'images, le zoom de l'histogramme, la carte de densité, différents débogage restent à effectuer.

IV – Conclusion

1. Bilan personnel

L'expérience vécue à l'observatoire astronomique de Strasbourg me laissera en mémoire, à n'en pas douter, de très bon souvenirs. J'aime les sciences physiques depuis que je suis très jeune et pouvoir participer aux développements de celles-ci au travers de l'informatique fut pour moi presque un honneur.

L'observatoire se situant en plein coeur de la construction d'un observatoire virtuel et universel est un des facteurs de cet enthousiasme car se dire que son travail est utile, utilisé, et reconnu est pour moi un élément des plus motivants lorsque je me lève le matin. Le cadre de travail ajoute également à cette ambiance sympathique avec le parc dans lequel on mange le midi ou discute avec ses collègues.

J'ai eu plusieurs occasions de travailler dans des entreprises auparavant mais rarement avec une aussi bonne ambiance, je n'ai eu aucun mal à m'intégrer, aucun problème avec les gens. La seule chose ayant pu perturber mon humeur ici est d'avoir eu le sentiment de n'avoir rien accomplie durant ma journée lorsque pour résoudre un problème il m'aura fallu recommencer à zéro.

Je suis donc content à l'idée de prolonger ce travail pendant 2 mois en CDD.

2. Bilan professionnel

Ce stage m'a beaucoup apporté sur le plan professionnel par l'apprentissage de nouvelles technologies comme GWT qui est intéressant et qui me servira sûrement encore à l'avenir. Les servlets, que je ne connaissais que de nom avant mon stage, m'ont permis d'y voir plus clair sur ce qu'il est possible de faire avec une application Web. Mais ce qui m'a le plus apporté d'un point de vue découverte informatique c'est la gestion des projets et le fonctionnement du travail en équipe, où chacun travaille de son coté mais discute avec les autres des possibilités, des ouvertures, des fonctionnalités ainsi que des contraintes à ne pas oublier.

L'importance qu'a pris l'anglais pendant mon stage, tant du point de vue informatique que relationnel m'a beaucoup intéressé. Voulant continuer mes études en Erasmus, j'ai entériné cette décision ici en discutant avec de nombreux étudiants étrangers travaillant à l'observatoire. Ils m'ont convaincu plus encore que je ne l'étais déjà, des opportunités d'une telle situation. J'ajouterais que pendant le meeting où l'anglais était omniprésent, j'ai regretté d'avoir des lacunes mais j'ai apprécié de pouvoir comprendre une grande partie des choses présentées.

3. Conclusion

Le stage m'ayant énormément apporté tant sur le plan personnel que professionnel, j'ai été heureux en retour de pouvoir faire profiter le CDS de mes connaissances acquises lors de mon DUT, mais aussi des connaissances acquises par moi-même notamment d'un point de vue développement Web.

Les problèmes rencontrés, et surtout les moyens d'y remédier, sont autant de cartes que je ne regrette pas d'avoir en main aujourd'hui. Elles me seront très utiles à l'avenir grâce aux expériences acquises pour les surmonter, ainsi que pour les traiter la prochaine fois que j'y serais confronté. Il m'a fallu parfois apprendre à être patient face à des problèmes qui semblaient insurmontables, mais qui en fin de compte, après une analyse approfondie, se résolvaient en posant le problème sur papier, renforçant sa compréhension et débouchant sur une solution adéquate.

Enfin, faire un stage dans un milieu universitaire m'a donné l'envie de continuer mes études plus que je ne le pensais car apprendre et découvrir semblent être compatibles avec travail et ambition tout autant qu'avec intérêt et investissement personnel.

V – Annexes

1. Lexique

Framework : Ensemble de bibliothèques, d'outils et de conventions permettant le développement d'applications. Ces composants sont organisés pour être utilisés en interaction les uns avec les autres.

GWT (Google Web Toolkit) : Framework développé par google qui permet de créer des pages web dynamiques en utilisant les technologies AJAX.. C'est un logiciel libre sous licence Apache 2.0. (cf. http://code.google.com/intl/fr/webtoolkit/)

Servlet : Application Java qui permet de créer dynamiquement des données au sein d'un serveur HTTP.

VOTable : Format XML de tables de données astronomiques défini par l'IVOA. Actuellement la version la plus récentes est la 1.20. (cf. http://www.ivoa.net/Documents/latest/VOT.html)

CSV (Comma Separated Values) : Format de tables de données astronomiques communément utilisé par les astronomes. (cf. http://www.rfc-editor.org/rfc/rfc4180.txt)

TSV (Tab Separated Values) : Format de tables de données astronomiques sans réelle définition mais communément utilisé par les astronomes dont les valeurs sont séparées par des tabulations.

IVOA (International Virtual Observatory Alliance): Groupe ayant pour but de définir les standards nécessaires à la création d'un observatoire virtuel internationale.
(cf. http://www.ivoa.net/)

STIL (Starlink Tables Infrastructure Library) : Librairies Java d'outils pour la manipulation de tables de données astronomiques. (cf. http://www.star.bristol.ac.uk/~mbt/stil/)

STILTS (STIL Tool Set) : Outils complémentaires pour le traitements de tables de données astronomiques. (cf. http://www.star.bristol.ac.uk/~mbt/stilts/)

RIA (Rich Internet Application): Application Internet s'exécutant dans un navigateur, qui offre des caractéristiques similaires aux logiciels traditionnels installés sur un ordinateur.

2. Exemples

1. VOTable

Voici un exemple de fichier VOTable provenant de Vizier. On peut sconstater qu'il contient à la fois les metadonnées et les données d'une table de données d'objet astronomiques.

- La balise FIELD, en rouge, rend visible la description d'un champ de la table (qui pourrais être représenté comme l'entête d'un colonne).
- La balise TR, en vert, est l'une des lignes de la table qui contient de nombreux champs vides pour l'exemple.

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE version="1.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://www.ivoa.net/xml/VOTable/v1.1"
 xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.1 http://www.ivoa.net/xml/VOTable/v1.1">
 <DESCRIPTION>
  VizieR Astronomical Server: newviz.u-strasbg.fr    2009-05-26T13:15:46
  Explanations and Statistics of UCDs:                    See LINK below
  In case of problem, please report to:         question@simbad.u-strasbg.fr
 </DESCRIPTION>
<!-- VOTable description at http://www.ivoa.net/Documents/latest/VOT.html -->

<DEFINITIONS>
 <COOSYS ID="J2000" system="eq_FK5" equinox="J2000"/>
</DEFINITIONS>

<INFO ID="Ref" name="-ref" value="VOTx21854"/>
<INFO ID="MaxTuples" name="-out.max" value="50000"/>

<RESOURCE ID="yCat_2164" name="II/164">
 <DESCRIPTION>DDO Photoelectric Photometric Catalog (Mermilliod+ 1989)</DESCRIPTION>
 <TABLE ID="II_164_obs" name="II/164/obs">
  <DESCRIPTION>Catalog of individual observations</DESCRIPTION>
  <!-- Now comes the definition of each field -->
  <FIELD name="LID" ucd="meta.id;meta.main" datatype="char" arraysize="10*"><!-- ucd="ID_MAIN" -->
   <DESCRIPTION>Lausanne identification code (G1)</DESCRIPTION>
  </FIELD>
  <FIELD name="m_LID" ucd="meta.code.multip" datatype="char"><!-- ucd="CODE_MULT_INDEX" -->
   <DESCRIPTION>[D1239] Remarks on duplicity</DESCRIPTION>
  </FIELD>
  <FIELD name="f_LID" ucd="meta.code.error" datatype="char"><!-- ucd="CODE_VARIAB" -->
   <DESCRIPTION>[V] Variability flag</DESCRIPTION>
  </FIELD>
  <FIELD name="m48" ucd="phot.mag;em.opt.B" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_MAG" -->
   <DESCRIPTION>? Magnitude in the filter 48 of DDO system</DESCRIPTION>
```

```
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(48-51)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_48-51" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(45-48)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_45-48" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(42-45)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_42-45" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(41-42)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_41-42" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(38-41)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_38-41" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="C(35-38)" ucd="phot.color" datatype="float" width="6" precision="3" unit="mag"><!--
ucd="PHOT_DDO_35-38" -->
     <DESCRIPTION>? Colour index in DDO system</DESCRIPTION>
     <VALUES null="" />
    </FIELD>
    <FIELD name="l_Nobs" ucd="meta.code.error" datatype="char"><!-- ucd="CODE_LIMIT" -->
     <DESCRIPTION>[&gt;S*] Limit flag on Nobs, 'S' for standard '*' for an estimated number</DESCRIPTION>
    </FIELD>
<DATA>     <TABLEDATA>
<TR><TD>-206700604</TD><TD></TD><TD></TD><TD>10.860</TD><TD></TD><TD>1.272</TD><TD>0.790
</TD><TD>0.084</TD><TD></TD><TD></TD><TD> </TD></TR>
<TR><TD>-008900079</TD><TD></TD><TD></TD><TD>9.284</TD><TD></TD><TD>1.394</TD><TD>1.251<
/TD><TD>0.179</TD><TD>-0.300</TD><TD></TD><TD> </TD></TR>

...

<TR><TD>-007102179</TD><TD> </TD><TD>
</TD><TD></TD><TD></TD><TD>0.927</TD><TD>0.445</TD><TD>0.051</TD><TD>-0.914</TD><TD>1.029
</TD><TD> </TD></TR>
</TABLEDATA></DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2. Code GWT

Ceci est une fonction créant un cadre contenant des champs de formulaire, on remarque la ressemblance avec Swing de Java.

```
private FlowPanel createTextPanel() {
	FlowPanel fontP = new FlowPanel();

	String[] fontstyles = { "plain", "bold", "italic", "bold-italic" };
	MyListBox fontStyle = new MyListBox("fontstyle", "font
		Style :",fontstyles);

	String[] fontsizes = new String[15];
	for (int i = 1; i < 16; i++)
		fontsizes[i - 1] = String.valueOf(i * 2);

	MyListBox fontSize = new MyListBox("fontsize","size", fontsizes, 5);

	String[] fonts = { "bitstream_charter", "courier", "courier_10_pitch",
		"cursor", "default", "dialog", "dialoginput", "lucida_bright",
		"lucida_sans", "lucida_sans_typewriter", "luxi_mono", "luxi_sans",
		"luxi_serif", "monospaced", "sansserif", "serif", "utopia" };
	MyListBox font = new MyListBox("font", "font", fonts, 15);

	MyTextBox title = new MyTextBox("Graphic title", "title");
	MyTextBox xlabel = new MyTextBox("Label x axis", "xlabel");
	MyTextBox ylabel = new MyTextBox("Label y axis", "ylabel");
	MyTextBox auxlabel = new MyTextBox("Label aux Axis", "auxlabel");

	fontP.add(font);
	fontP.add(fontStyle);
	fontP.add(fontSize);
	fontP.add(title);
	fontP.add(xlabel);
	fontP.add(ylabel);
	fontP.add(auxlabel);

	return fontP;
}
```

3. Javascript généré par GWT:

Ceci est le code que GWT génère après compilation du code java dont on peut voir un extrait dans l'annexe 2. Comme on peut le remarquer, le code est illisible et compact pour prendre le moins de place possible. Ici la plupart des retours à la ligne sont dûs au manque d'espace, le fichier JavaScript, lui, est écrit sur une dizaine de lignes environs.

1function cds_portal_GWTPortal(){var l='',F='" for "gwt:onLoadErrorFn"',D='" for "gwt:onPropertyErrorFn"',n='"><\/script>',p='#',r='/',tb='0456FB409BAB41E7796E525A956FE8F7.cache.html',vb='4E8E81F4B153C6C8C30C8EB77B2B240D.cache.html',zb='<script defer="defer">cds_portal_GWTPortal.onInjectionDone(\'cds.portal.GWTPortal\')<\/script>',Db='<script id="',A='=',q='?',wb='BCF11D98936B175B71D3F38FF8DC7A2E.cache.html',C='Bad handler "',ub='DD4A2A7B457ECE37A1FCE22BACBCD9B0.cache.html',xb='DOMContentLoaded',sb='EB2D90DA56754ECBEA431C351BC30FDD.cache.html',o='SCRIPT',Cb='__gwt_marker_cds.portal.GWTPortal',s='base',nb='begin',cb='bootstrap',m='cds.portal.GWTPortal',u='clear.cache.gif',z='content',Bb='end',lb='gecko',mb='gecko1_8',yb='gwt.hybrid',E='gwt:onLoadErrorFn',B='gwt:onPropertyErrorFn',y='gwt:property',rb='hosted.html?cds_portal_GWTPortal',kb='ie6',ab='iframe',t='img',bb="javascript:'",pb='loadExternalRefs',v='meta',eb='moduleRequested',Ab='moduleStartup',jb='msie',w='name',gb='opera',db='position:absolute;width:0;height:0;border:none',ib='safari',qb='selectingPermutation',x='startup',ob='unknown',fb='user.agent',hb='webkit';var Fb=window,k=document,Eb=Fb.__gwtStatsEvent?function(a){return Fb.__gwtStatsEvent(a)}:null,tc,jc,ec,dc=l,mc={},wc=[],sc=[],cc=[],pc,rc;Eb&&Eb({moduleName:m,subSystem:x,evtGroup:cb,millis:(new Date()).getTime(),type:nb});if(!Fb.__gwt_stylesLoaded){Fb.__gwt_stylesLoaded={}}if(!Fb.__gwt_scriptsLoaded){Fb.__gwt_scriptsLoaded={}}function ic(){try{return Fb.external&&(Fb.external.gwtOnLoad&&Fb.location.search.indexOf(yb)==-1)}catch(a){return false}}
2function lc(){if(tc&&jc){var c=k.getElementById(m);var b=c.contentWindow;b.__gwt_initHandlers=cds_portal_GWTPortal.__gwt_initHandlers;if(ic()){b.__gwt_getProperty=function(a){return fc(a)}}cds_portal_GWTPortal=null;b.gwtOnLoad(pc,m,dc);Eb&&Eb({moduleName:m,subSystem:x,evtGroup:Ab,millis:(new Date()).getTime(),type:Bb})}}
3function gc(){var j,h=Cb,i;k.write(Db+h+n);i=k.getElementById(h);j=i&&i.previousSibling;while(j&&j.tagName!=o){j=j.previousSibling}function f(b){var a=b.lastIndexOf(p);if(a==-1){a=b.length}var c=b.indexOf(q);if(c==-1){c=b.length}var d=b.lastIndexOf(r,Math.min(c,a));return d>=0?b.substring(0,d+1):l}
4;if(j&&j.src){dc=f(j.src)}if(dc==l){var e=k.getElementsByTagName(s);if(e.length>0){dc=e[e.length-1].href}else{dc=f(k.location.href)}}else if(dc.match(/^\w+:\/\//)){}else{var g=k.createElement(t);g.src=dc+u;dc=f(g.src)}if(i){i.parentNode.removeChild(i)}}
5function qc(){var f=document.getElementsByTagName(v);for(var d=0,g=f.length;d<g;++d){var e=f[d],h=e.getAttribute(w),b;if(h){if(h==y){b=e.getAttribute(z);if(b){var i,c=b.indexOf(A);if(c>=0){h=b.substring(0,c);i=b.substring(c+1)}else{h=b;i=l}mc[h]=i}}else if(h==B){b=e.getAttribute(z);if(b){try{rc=eval(b)}catch(a){alert(C+b+D)}}}else if(h==E){b=e.getAttribute(z);if(b){try{pc=eval(b)}catch(a){alert(C+b+F)}}}}}}}
6function vc(d,e){var a=cc;for(var b=0,c=d.length-1;b<c;++b){a=a[d[b]]||(a[d[b]]=[])}a[d[c]]=e}
7function fc(d){var e=sc[d](),b=wc[d];if(e in b){return e}var a=[];for(var c in b){a[b[c]]=c}if(rc){rc(d,a,e)}throw null}
8var hc;function kc(){if(!hc){hc=true;var a=k.createElement(ab);a.src=bb;a.id=m;a.style.cssText=db;a.tabIndex=-1;k.body.appendChild(a);Eb&&Eb({moduleName:m,subSystem:x,evtGroup:Ab,millis:(new Date()).getTime(),type:eb});a.contentWindow.location.replace(dc+uc)}}
9sc[fb]=function(){var d=navigator.userAgent.toLowerCase();var b=function(a){return parseInt(a[1])*1000+parseInt(a[2])};if(d.indexOf(gb)!=-1){return gb}else if(d.indexOf(hb)!=-1){return ib}else if(d.indexOf(jb)!=-1){var c=/msie ([0-9]+)\.([0-9]+)/.exec(d);if(c&&c.length==3){if(b(c)>=6000){return kb}}}else if(d.indexOf(lb)!=-1){var c=/rv:([0-9]+)\.([0-9]+)/.exec(d);if(c&&c.length==3){if(b(c)>=1008)return mb}return lb}return ob};wc[fb]={gecko:0,gecko1_8:1,ie6:2,opera:3,safari:4};cds_portal_GWTPortal.onScriptLoad=function()

{if(hc){jc=true;lc()}};cds_portal_GWTPortal.onInjectionDone=function()
{tc=true;Eb&&Eb({moduleName:m,subSystem:x,evtGroup:pb,millis:(new
Date()).getTime(),type:Bb});lc()};gc();qc();Eb&&Eb({moduleName:m,subSystem:x,evtGroup:cb,millis:(new
Date()).getTime(),type:qb});var uc;if(ic())
{uc=rb}else{try{vc([kb],sb);vc([mb],tb);vc([lb],ub);vc([gb],vb);vc([ib],wb);uc=cc[fc(fb)]}catch(a){return}}var
oc;function nc(){if(!ec){ec=true;lc();if(k.removeEventListener){k.removeEventListener(xb,nc,false)}if(oc)
{clearInterval(oc)}}}
10if(k.addEventListener){k.addEventListener(xb,function(){kc();nc()},false)}var oc=setInterval(function(){if(/loaded|
complete/.test(k.readyState)){kc();nc()}},50);Eb&&Eb({moduleName:m,subSystem:x,evtGroup:cb,millis:(new
Date()).getTime(),type:Bb});Eb&&Eb({moduleName:m,subSystem:x,evtGroup:pb,millis:(new
Date()).getTime(),type:nb});k.write(zb)}
11cds_portal_GWTPortal.__gwt_initHandlers=function(i,e,j){var
d=window,g=d.onresize,f=d.onbeforeunload,h=d.onunload;d.onresize=function(a)
{try{i()}finally{g&&g(a)}};d.onbeforeunload=function(a){var c,b;try{c=e()}finally{b=f&&f(a)}if(c!=null){return
c}if(b!=null){return b}};d.onunload=function(a)
{try{j()}finally{h&&h(a);d.onresize=null;d.onbeforeunload=null;d.onunload=null}}};cds_portal_GWTPortal();