IUT Nancy Charlemagne
Université de Lorraine
2 ter Boulevard Charlemagne
54052 Nancy Cedex
Dépt. Informatique

Étude de la recherche dans de grandes quantités de données astronomiques

Rapport de stage DUT Informatique
Entreprise : Observatoire de Strasbourg

BISCH Yann
2012/2013

IUT Nancy Charlemagne
Université de Lorraine
2 ter Boulevard Charlemagne
54052 Nancy Cedex
Dépt. Informatique

Étude de la recherche dans de grandes quantités de données astronomiques

Rapport de stage DUT Informatique
Observatoire de Strasbourg, 11 rue de l'Université, 67000 Strasbourg

BISCH Yann

Tuteurs de stage : SCHAAFF André, DERRIERE Sébastien, BOCH Thomas
Parrain de stage : MANGEOL Bernard

Remerciements

Tout d'abord, je remercie tout le personnel de l'Observatoire qui m'a permis de travailler dans de bonnes conditions avec une bonne ambiance, et plus particulièrement :

Mon maître de stage, Monsieur André Schaaff, ingénieur de recherche à l'Observatoire, qui m'a permis de travailler sur un sujet très intéressant et de plus en plus à la mode. Je le remercie également pour son soutien ainsi que l'aide qu'il m'a apporté.

Je remercie également Sébastien Derrière, astronome-adjoint à l'Observatoire qui a également été mon tuteur de stage tout comme monsieur Schaaff, et qui m'a également aidé ainsi que posé de questions très pertinentes qui m'ont fait avancer.

Je remercie Thomas Boch, ingénieur de recherche à l'Observatoire, mon troisième tuteur de stage, pour ses questions également très pertinentes ainsi que son aide.

Je remercie mon parrain de stage, Monsieur Bernard Mangeol, pour son soutien et ses conseils avisés.

Je remercie également toutes les personnes nous ayant accueilli et présenté l'observatoire dont : Patricia Vannier, Gilles Landais, Thomas Boch, Anaïs Oberto, André Schaaff, Thomas Keller et Laurent Michel.

Table des matières

I) Introduction

Dans le but de finaliser mon DUT et d'obtenir ce diplôme bac+2, je devais réaliser un stage en entreprise. Monsieur Bernard Mangeol mon parrain de stage pour celui-ci, m'avait parlé de la possibilité de faire un stage à l'Observatoire de Strasbourg. Intéressé par ce milieu, j'ai donc décidé de prendre contact avec monsieur André Schaaff, ingénieur de recherche à l'Observatoire.
Il existait déjà à ce moment là environ 5 sujets différents qu'il était possible de choisir mais également un certain nombre de candidats à ceux-ci. Deux d'entre eux avaient attiré particulièrement mon attention malheureusement, il y avait déjà des candidats postulant pour ceux-ci qui étaient presque déjà désignés.
Étant beaucoup moins intéressé par les autres sujets, je commençais à rechercher d'autres entreprises lorsque monsieur Schaaff me proposa un sujet sur le « Big Data » ou plus particulièrement la recherche dans de grandes quantités de données qu'il n'avait pas encore mis en place mais qui semblait très intéressant. J'ai donc accepté sa proposition.

Le but de ce stage était donc d'étudier les possibilités de recherche dans de grande quantités de données astronomiques à l'aide de moteurs d'indexation et de recherche. En effet, l'Observatoire et plus précisément le Centre de Données astronomiques de Strasbourg (CDS), dispose d'une masse importante de données et bien qu'il existait déjà un service puissant pour la recherche, celui-ci ne proposait pas de correction orthographique, ou la possibilité de rechercher dans la totalité du texte.

Ce rapport va donc résumer et expliquer la manière dont s'est déroulé ce stage selon la structure suivante :
Dans un premier temps, je présenterai l'Observatoire astronomique de Strasbourg et les différentes équipes et différents services qui le composent ainsi qu'une explication plus détaillée du sujet traité. Ensuite, je parlerai de l'État de l'Art établi sur les différents moteurs et plate-formes existant(e)s en rapport avec la recherche dans de grandes quantités de données. Puis je présenterai les tests effectués sur les données du CDS suivi par une partie expliquant le fonctionnement et de l'un des moteurs de recherche testés pour enfin finir avec la présentation d'un prototype que j'ai réalisé.

II) Présentation

1) Présentation de l'Observatoire Astronomique de Strasbourg

L'Observatoire Astronomique de Strasbourg est un Observatoire des Sciences de l'Univers (OSU),
un UFR de l'Université de Strasbourg, et une Unité Mixte de Recherche (UMR 7550) entre l'Université et le CNRS.
Il est actuellement dirigé par Hervé Wozniak.

L'Observatoire avait pour vocation initiale l'astronomie de position et l'observation de comètes, de météorites et d'étoiles variables. Plus tard, on s'y intéressa à la photométrie et à l'observation d'étoiles doubles. Ce n'est qu'à partir de 1965 que les astronomes comprirent que l'observation au sol avait atteint ses limites et proposèrent le concept du Satellite Hipparcos à l'agence Spatiale Européenne.

Historiquement, il y a eu trois Observatoires Astronomiques. Le premier fut construit en 1673 sur les tours d'enceintes de la ville, avec l'aide de l'astronome Julius Reichelt. Le second a quant à lui été bâti en 1828 sur les bâtiments de l'Académie de Strasbourg. L'Observatoire Astronomique obtint son emplacement actuel en 1881, année de son inauguration. Celui-ci fait partie, avec le jardin botanique situé juste à côté, des éléments qui symbolisent la décision politique de faire de Strasbourg une vitrine, prise par l'empereur Guillaume 1er D'Allemagne à la suite de la guerre de 1870.

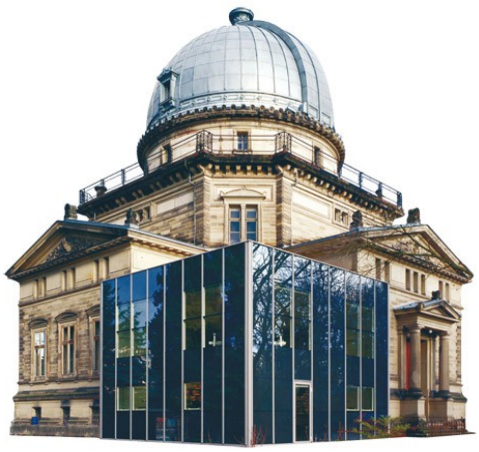


Illustration 1: Photo de la grande coupole

L'Observatoire actuel est composé de trois bâtiments dont le plus célèbre est celui de la Grande Coupole, qui contient la 3ieme plus grande lunette de France par sa taille.
L'Observatoire héberge également le Planétarium de Strasbourg dont il a eu la responsabilité de 1986 à 2008. Celui-ci est désormais intégré au Jardin des Sciences de l'Université de Strasbourg. L'Observatoire dispose également d'un riche patrimoine d'instruments et d'ouvrages anciens.

L'Observatoire est structuré en trois équipes de recherche et deux Services d'Observation de l'Institut National des Sciences de l'Univers (INSU),
le Survey Science Centre d'XMM-Newton (SSC-XMM) et le Centre de Données astronomiques de Strasbourg (CDS).

Son statut d'OSU (voir décret numéro 85-657 du 27 juin 1985) place l'Observatoire astronomique au cœur du dispositif national mis en œuvre par l'INSU.
Par conséquent, l'Observatoire astronomique de Strasbourg a pour mission de contribuer aux progrès de la connaissance par :

- l'acquisition de données d'observation ;
- le développement et l'exploitation de moyens appropriés ;
- l'élaboration des outils théoriques nécessaires.

De plus, l'Observatoire est également chargé :

- de fournir des services liés à son activité de recherche ;
- d'assurer la formation des étudiants et des personnels de recherche ;
- d'assurer la diffusion des connaissances ;
- de prendre part à des activités de coopération internationale.

2) Équipes de recherche

2.1) Galaxies

L'équipe " Galaxies " étudie la formation et l'évolution des galaxies
au travers de leurs populations stellaires et de la dynamique des étoiles et de la matière noire.
Elle est impliquée dans la préparation de la mission satellitaire astrométrique Gaia de l'Agence Spatiale Européenne, dont le lancement est prévu pour octobre prochain, ainsi que dans le grand relevé cinématique RAVE, qui fournira des données d'une précision inédite sur la Voie Lactée.
Les recherches sont concentrées sur les caractéristiques et les propriétés physiques de notre galaxie et de ses plus proches voisines.
Afin de pouvoir étudier la dynamique gravitationnelle qui régit les mouvements du gaz et des étoiles, on analyse les évolutions de la population stellaire de ces galaxies.

Les résultats obtenus permettent de reconstituer les événements clés de la vie d'une galaxie.

L'équipe mène également d'autres recherches sur les amas stellaires, qui sont des composants fondamentaux d'une galaxie.
Grâce aux connaissances accumulées concernant les mouvements gravitationnels des étoiles, il est possible de simuler numériquement l'évolution d'un amas stellaire, afin de mieux comprendre la formation des galaxies.

2.2) Hautes Énergies

L'équipe " Hautes Énergies " s'intéresse aux sources galactiques et extragalactiques émettrices en rayons X,
objets compacts (étoiles à neutron, naines blanches, etc.) et noyaux actifs de galaxies.
Elle est impliquée dans le SSC-XMM, un consortium international de laboratoires sélectionné par l'ESA et labellisé par l'INSU comme Service d'Observation,
qui est en charge de fournir des catalogues complets de sources X observées par le satellite XMM-Newton à la communauté internationale.

2.3) CDS

Le Centre de Données astronomiques de Strasbourg (CDS) est à la fois une équipe de recherche et un Service d'Observation.
Les services de bases de données (Simbad, VizieR) et de visualisation (Aladin),
développés par le CDS, sont utilisés par l'ensemble de la communauté astronomique mondiale.
Celui-ci est l'un des acteurs majeurs du développement de l'Observatoire Virtuel International en astronomie.
Fin 2008, le CDS a été labellisé "Très Grande Infrastructure de Recherche" (TGIR),
ce qui le place au même niveau que des infrastructures internationales comme l'European Southern Observatory ou RENATER à l'échelon national.
J'ai effectué mon stage dans ce service, c'est pourquoi il va maintenant être présenté plus en détail.

3) Le Centre de Données Astronomiques de Strasbourg

Créé en 1972 sous le nom de "Centre de Données Stellaires", il devint le Centre de Données astronomiques de Strasbourg en 1983.

Il héberge maintenant la base de donnée de référence mondiale pour l'identification d'objets astronomiques.
Par conséquent, ses missions consistent :
- à rassembler des informations utiles concernant les objets astronomiques sous forme informatisée ;
- à mettre à jour ces données en les critiquant et en les comparant ;
- à les distribuer à la communauté internationale ;
- à mener des recherches utilisant ces données.

Le CDS emploie actuellement 8 chercheurs/astronomes-adjoints, 1 chercheur postdoctoral, 11 informaticiens, 10 documentalistes et 3 administratifs,
soit 33 personnes au total.
Les services principaux du CDS se nomment Simbad, VizieR et Aladin.

3.1) Simbad

Simbad est la base de données de référence pour la nomenclature et la bibliographie des objets astronomiques situés hors du système solaire, qu'il a accumulé pendant plus de 40 ans.
Ce service permet aux astronomes de trouver facilement des informations sur plus de 7 millions d'objets, grâce à plus de 280 mille références bibliographiques.
De plus, la base de donnée Simbad contient un résolveur de noms permettant de retrouver l'objet désigné par l'un des 18 millions d'identifiants qu'elle contient.
Des liens peuvent bien évidemment être faits avec d'autres services fonctionnant sur Internet.
Simbad recevait en 2011 une moyenne 284 000 requêtes par jour. Pour le mois dernier, il a dû traiter 18 985 164 requêtes soit environ 7 requêtes par secondes.

Illustration 2: Logo de Simbad

3.2) VizieR

VizieR est une base de données qui regroupe à l'heure actuelle plus de 11 000 catalogues d'objets astronomiques.

Ces catalogues sont constitués de données relevées durant des missions d'observation de l'espace puis ajoutées à VizieR par les documentalistes du CDS.

Ce procédé permet d'homogénéiser les données astronomiques afin de pouvoir les comparer et de les exporter sous différents formats.

Les interrogations sur la base de catalogues peuvent porter sur de multiples critères comme la longueur d'onde avec laquelle les objets ont étés observés

ou encore le nom de la mission correspondante.

Entre 2011 et 2012, les requêtes de VizieR ont augmenté d'une moyenne de 370 000 requêtes par jour à une moyenne de 377 000 requêtes par jour. Le service connaît des pics d'utilisation pouvant atteindre 2 millions de requêtes en un jour.

Illustration 3: Logo de VizieR

3.3) Aladin

Aladin est un un atlas interactif du ciel qui permet de visualiser et de comparer des images astronomiques.

Il a été entièrement développé en Java pour l'ensemble de la communauté internationale par Pierre Fernique, Thomas Boch et François Bonnarel.

Ces images proviennent d'observatoires sol et spatiaux et peuvent être utilisées pour former des cartes du ciel en trois dimensions (boules HEALPix).

Elles peuvent être fournies par l'utilisateur, par la base de données d'Aladin ou par d'autres bases de données telles que la NASA Extragalactical Database.
Aladin est à la fois une application téléchargeable et une applet Java, toutes deux disponibles sur le site internet du CDS.
Aladin recevait en 2012 une moyenne de 18586 requêtes par jour, ce qui représente une hausse de 11% par rapport à l'année précédente, dont la moyenne était de 16800 requêtes par jour.
Cependant, on peut remarquer que l'usage d'Aladin se resserre sur l'application standalone au détriment de l'applet.
Cela témoigne d'une diminution de l'audience "grand public" au profit d'une utilisation scientifique.

Illustration 4: Logo d'Aladin

4) L'International Virtual Observatory Alliance

L'International Virtual Observatory Alliance (IVOA) est une organisation internationale regroupant 19 projets d'Observatoire Virtuel (VO) provenant de 17 pays différents : Arménie, Australie, Brésil, Canada, Chine, France, Allemagne, Hongrie, Inde, Italie, Japon, Russie, Ukraine, Espagne, Royaume-Uni, Argentine et États Unis.
Ce projet international a un but similaire au World Wide Web Consortium : mettre en place des standards et décider quels seront les travaux les plus importants à faire ainsi que les stratégies à adopter. Ceci permet de faciliter les échanges internationaux et la collaboration nécessaire à l'élaboration de standards d'interopérabilité et au déploiement d'outils, de systèmes et de structures organisés nécessaires pour permettre une utilisation internationale des archives astronomiques (images, catalogues, etc..). Telle est la mission confiée à l'IVOA lors de sa création en juin 2002.
Le CDS participe activement à ce projet notamment par la réalisation des services cités précédemment : Aladin, Simbad et Vizier.

5) Présentation du sujet

5.1) L'existant

Les services VizieR et Simbad hébergent une quantité importante de données. Il existe d'ores et déjà un service permettant la recherche parmi les descriptions des 11,000 catalogues de la base VizieR, qui fonctionne très bien.

En effet, il est basé sur du langage C et des fichiers binaires pour permettre une recherche très rapide parmi la grande quantité de catalogues. Cette recherche se fait sur l'index réalisé à partir des « parfiles », des fichiers dont je parlerai dans la partie « Tests sur les données du CDS » et dont vous pouvez voir un exemple à la l'annexe « A ». Cet index contient principalement les « keywords » (mots clés en français), les auteurs et le nom des catalogues. Il ne contient pas par exemple la description présente dans ces « parfiles ».

De plus, la recherche actuelle ne permet pas la correction des mots de la recherche, ce qui peut être assez contraignant.

5.2) Les besoins

Ainsi, malgré un service déjà bien performant, le CDS désirait avoir un moteur de recherche plus performant encore capable de proposer des corrections orthographiques, de catégoriser les mots indexés ou encore de proposer dynamiquement à partir de ce qui est tapé dans le champ de recherche, différents mots possibles existants dans l'index.

Mon stage visait donc à étudier les possibilités de recherches dans ces grandes quantités de données à l'aide de moteurs de recherche de type Google. Pour mener à bien ce projet, je suis passé par **« trois »** phases principales d'étude et de réalisation.

III) État de l'Art

La première phase de mon stage consistait à établir un état de l'art, en recherchant les différents moteurs et plate-formes logicielles qui pouvaient déjà exister et en les testant pour voir comment ils fonctionnaient.

1) Les projets dans le domaine de la recherche d'information

En effet, à l'aide des indications de mon maître de stage et de mes recherches, j'ai pu trouver un certain nombre de plate-formes offrant une possibilité de mise en place de l'indexation de documents et de recherche sur l'index ainsi réalisé.
Néanmoins, j'ai pu remarquer que la plupart de ces moteurs si ce n'est tous, étaient basés a minima sur Hadoop qui semble être la plus basse couche pour réaliser des moteurs de recherche sur une ou plusieurs machines (j'y reviendrai dans la cinquième partie du chapitre 5 de ce rapport page 28).

1.1) Hadoop

Hadoop est un des nombreux projets de la fondation Apache Software Foundation, dont le but est de pouvoir développer des logiciels basés sur l'informatique distribuée et donc dans le cadre de mon stage, de permettre la recherche sur une ou plusieurs machines via un ou plusieurs cœurs (voir chapitre 5 partie 5 « La notion de cœur » page « 28 »). Je m'arrêterai ici pour la description d'Hadoop étant donné sa complexité, que ce n'était pas le sujet de mon stage et qu'Hadoop n'est pas intervenu directement dans la réalisation de celui-ci.

Illustration 5: Logo Hadoop

1.2) Lucene

Comme dit précédemment, Apache a de nombreux projets et Hadoop n'est qu'une couche de base d'autres projets notamment d'indexation et de recherche « Full Text » c'est à dire dans du texte brut. J'ai en effet trouvé le projet Apache Lucene qui est un moteur de recherche basé sur Hadoop permettant de rechercher grâce à la réalisation d'un index fait à partir de texte brut.

Illustration 6: Logo de Lucene

1.3) Solr

Illustration 7: Logo de Solr

Mais une couche encore au dessus, il existe le projet Apache Solr, basé sur le moteur Apache Lucene (et donc sur Hadoop), il offre une plus grande maniabilité et beaucoup de composants en plus de ceux existants dans Lucene.

1.4) ElasticSearch

Mon maître de stage m'avait également parlé d'ElasticSearch, développé par Shay Banon seul puis aidé par une petite communauté grandissante, qui est plus qu'une API étant donné qu'ElasticSearch permet une intégration rapide sans avoir besoin de connaître le langage de programmation Java. Ce projet est également basé sur Lucene, et est une sorte de concurrent à Solr, ce qui explique leur ressemblance.

Illustration 8: Logo d'ElasticSearch

1.5) Autres : Katta, Nutch, IndexTank

J'ai également pu trouver d'autres plate-formes ayant également pour but l'indexation et la recherche. Il y a notamment Katta que dont j'ai pu tester la démonstration, Nutch qui est orienté « web crawler », IndexTank et d'autres encore...

2) Choix et premiers tests

Tout d'abord, il faut savoir que j'ai commencé par tester les démonstrations proposées par chacun des moteurs pour plusieurs raisons :
Réaliser une voire plusieurs interfaces pour chacun des moteurs m'aurait pris énormément de temps du fait que je ne connaissais pas le fonctionnement global de ces moteurs et qu'en réaliser 6 n'était pas réaliste étant donnée la durée du stage. Les derniers projet cités ci-dessus étant beaucoup moins connus que Solr et ElasticSearch, j'ai décidé d'en écarter la plupart pour me concentrer sur les plus grandes références de moteurs dans le domaine de l'indexation et de la recherche, et ceux qui semblaient répondre le mieux aux besoins.
Ainsi, bien que Nutch semblait avoir une certaine place, celui-ci était orienté « web crawler » ce qui ne répondait pas aux besoins cités plus haut. En effet, ce moteur également basé sur Lucene servait à indexer et à faire des recherche au sein de pages Web (html) alors que je devais rechercher dans des fichiers texte.
IndexTank était trop peu connu, de même que Sphinx que j'ai découvert assez tard (ce qui montre qu'il n'était pas très connu), j'ai donc décidé de les écarter également de mon projet.
Ainsi j'ai testé les démonstrations de 4 moteurs : Katta, Lucene, Solr et ElasticSearch.

2.1) Katta

Dans un premier temps j'ai testé Katta qui est également basé sur Lucene tout comme ElasticSearch et Solr. . Je me suis arrêté au test de la démonstration du fait qu'il manquait de documentation claire mais également car il était moins complet que Solr ou ElasticSearch. Katta a donc été assez rapidement écarté, du fait du manque de temps, de son manque de popularité et d'une documentation peu exhaustive.

2.2) Lucene

Apache Lucene permet à l'aide de la démonstration d'indexer n'importe quel fichier. J'ai testé avec de simples fichiers texte et pu ainsi tester les requêtes. En effet, Lucene possède de base une sémantique de requête compréhensible par l'être humain. Par exemple, mettre un « + » devant un mot indique qu'il doit apparaître dans les documents retournés lors de la recherche. Un « - » indiquera que ce mot ne doit pas être présent. De la même manière que lorsque vous tapez dans Google des mots encadrés de guillemets (ex : « hello world »), Lucene recherchera parmi les documents indexés ceux qui contiennent cette phrase. Pour une liste plus exhaustive des différentes syntaxes possibles de requêtes, veuillez vous référer à la sitographie, partie documentation page « 41 ».

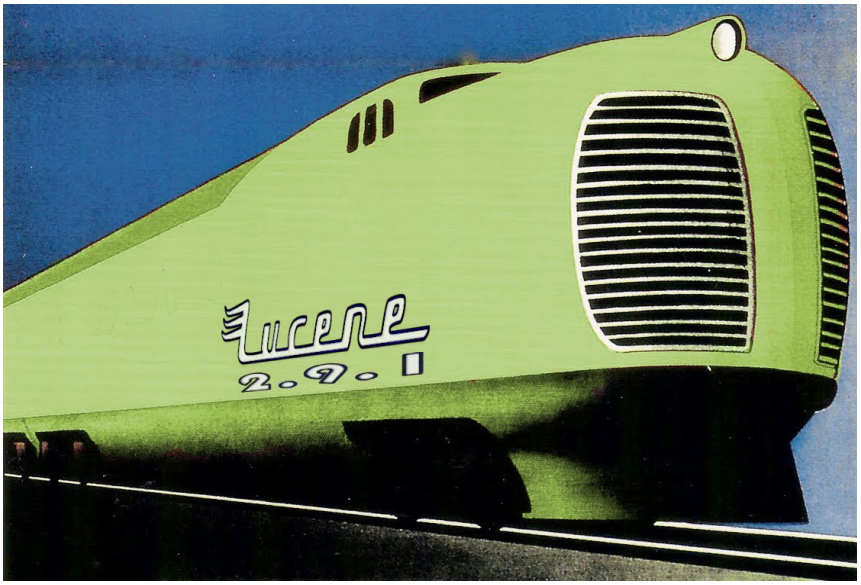


Illustration 9: Caricature de Lucene

2.3) Solr

Apache Solr, comme dit précédemment, est une couche au dessus de Lucene, il possède donc de base toutes les librairies de Lucene. Néanmoins, la syntaxe des requêtes est un peu modifiée : par exemple, avec Lucene nous pouvons indiquer un « boost » (voir la partie intitulé « Le score » page « 26 ») négatif pour un mot tandis que Solr lui n'accepte pas ceci. Il garde tout de même les boost positif, il est donc possible d'obtenir une requête similaire via le signe « - » cité plus haut. Pour une liste plus exhaustive des différentes syntaxes de requêtes Solr, veuillez vous référer à la sitographie, partie documentation page « 41 ».

La démonstration de Solr dispose également d'une interface administrateur pour gérer les différents cœurs (expliqué plus loin), visualiser la configuration de ceux-ci etc. et d'une interface orientée

utilisateur.

2.4) ElasticSearch

ElasticSearch est une sorte de concurrent de Solr. Basé également sur Lucene, il a une différence notable qui est est le passage obligatoire par des requêtes HTTP pour accéder aux résultats mais surtout pour modifier la configuration d'ElasticSearch. Il existe d'autres différences plus subtiles et plus complexes à comprendre à moins d'être un expert dans ces deux plate-formes logicielles.

Ces trois dernières démonstrations fonctionnant assez bien, j'ai décidé de les tester sur les données réelles du CDS.

IV) Tests sur les données du CDS

Afin de pouvoir mieux comparer ces différents moteurs et observer leurs performances, mon maître de stage m'a fourni des données.

1) Les données

En effet, j'ai disposé de plus de 11 000 fichiers texte (11013 pour être exact mais il en existe plus et cette quantité s'accroît de plusieurs centaines chaque année). Ces fichiers texte appelés « Readme » sont des résumés relativement structurés des catalogues stockés notamment dans la base VizieR.Vous pouvez en trouver un exemple juste en dessous. La taille de l'ensemble des 11 000 Readme est d'une quarantaine de Mégaoctets. Ces fichiers texte ont donc été simple à indexer dans Lucene (voir la présentation de Lucene page « 15 » pour les explications) mais au contraire ElasticSearch et Solr nécessitait respectivement des fichiers aux formats JSON et XML, ce format n'était donc pas approprié.

```
II/2B       Two-Micron Sky Survey (TMSS)          (Neugebauer+ 1969)
================================================================================
The Two-Micron Sky Survey
    Neugebauer G., Leighton R.B.
   <NASA SP-3047 (1969)>
   =1969IRC...C......0N
   =1969QB6.N47........
================================================================================
ADC_Keywords: Photometry, infrared; Surveys

Description:
    The catalog, giving sources of emission in the 2.2-micrometer region
    for more than 5000 stars, represents a systematic survey of the
    Northern Hemisphere for stars brighter than third magnitude. The
    survey was carried out with a telescope at Mount Wilson, California,
    having a 62-inch diameter and an f/l aluminized epoxy mirror mounted
    equatorially. Radiation at an effective wavelength of 2.2 micrometers
    was detected by a lead sulfide photoconductive cell cooled by liquid
    nitrogen. In addition to the 2.2-micrometer detector array, radiation
    at an effective wavelength of 0.84 micrometers was detected by a
    simple silicon photovoltaic cell. The catalog includes right ascension
    and declination (B1950.0), K and I magnitudes, number of measurements,
    V magnitude, spectral types, cross identifications to the numbering
    systems of the General Catalogue, the Durchmusterung catalogs, the
    Bright Star Catalogue, and star names.


File Summary:
--------------------------------------------------------------------------------
 FileName    Lrecl  Records   Explanations
--------------------------------------------------------------------------------
ReadMe          80        .   This file
catalog.dat    164     5612   Catalog Data
chisq.dat       36     2574   Chi-square Excess
remarks.dat     78      140   Remarks
--------------------------------------------------------------------------------

See also:
  II/94 : The Revised AFGL Sky Survey Catalog and Supplement (Price+ 1983)
 II/216 : Catalog of Infrared Observations, Edition 4 (CIO) (Gezari+ 1997)
```

Illustration 10: Exemple d'un fichier Readme

2) *Réalisation de traducteurs*

Du fait de ce problème de structure, j'ai donc réalisé des générateurs de XML et de JSON à partir des « Readme ». J'ai néanmoins été confronté à des problèmes lors de la lecture des « Readme » car ceux-ci n'avaient pas de structure entièrement fixe. En effet, il pouvait parfois manquer un(e) ou plusieurs champs/catégories, ce qui m'a amené à utiliser une autre version des « Readme »: les « parfiles ». Ils ne disposent pas d'autant d'informations que les « Readme » mais sont bien formatés : chaque nouveau champs/catégorie est symbolisé en début de ligne par un « % » suivi d'une lettre indiquant de quel champs il s'agit. Pour un exemple concret, voir l'annexe « A ». J'ai ainsi pu aisément générer les fichiers en XML et JSON grâce à un simple parcours ligne par ligne.

```
{

    date_creation: 1997/05/27
    date_modification: 2008/06/14
    identificateur: II/2B
    title: Two-Micron Sky Survey (TMSS) (Neugebauer+ 1969)
    original_title: The Two-Micron Sky Survey
    authors: [
        Neugebauer G.
        Leighton R.B.
    ]
    Description: The catalog, giving sources of emission in the 2.2-micrometer
region for more than 5000 stars, represents a systematic survey of the Northern
Hemisphere for stars brighter than third magnitude. The survey was carried out
with a telescope at Mount Wilson, California, having a 62-inch diameter and an f/l
aluminized epoxy mirror mounted equatorially. Radiation at an effective wavelength
of 2.2 micrometers was detected by a lead sulfide photoconductive cell cooled by
liquid nitrogen. In addition to the 2.2-micrometer detector array, radiation at an
effective wavelength of 0.84 micrometers was detected by a simple silicon
photovoltaic cell. The catalog includes right ascension and declination (B1950.0),
K and I magnitudes, number of measurements, V magnitude, spectral types, cross
identifications to the numbering systems of the General Catalogue, the
Durchmusterung catalogs, the Bright Star Catalogue, and star names.
    Keywords: Photometry, infrared; Surveys

}
```

Illustration 11: Exemple d'un Readme « traduit » en JSON pour ElasticSearch

```xml
<?xml version="1.0" encoding="UTF-8"?>
<add>
  <doc>
    <field name="id">2002</field>
    <field name="date_creation">1997-05-27T00:00:00Z</field>
    <field name="date_modification">2008-06-14T00:00:00Z</field>
    <field name="identificateur">II/2B</field>
    <field name="title">Two-Micron Sky Survey (TMSS) (Neugebauer+ 1969)</field>
    <field name="original_title">The Two-Micron Sky Survey</field>
    <field name="authors">Neugebauer G., Leighton R.B.</field>
    <field name="description">The catalog, giving sources of emission in the
2.2-micrometer region  for more than 5000 stars, represents a systematic survey of
the Northern Hemisphere for stars brighter than third magnitude. The survey was
carried out with a telescope at Mount Wilson, California, having a 62-inch
diameter and an f/l aluminized epoxy mirror mounted equatorially. Radiation at an
effective wavelength of 2.2 micrometers was detected by a lead sulfide
photoconductive cell cooled by liquid nitrogen. In addition to the 2.2-micrometer
detector array, radiation at an effective wavelength of 0.84 micrometers was
detected by a simple silicon photovoltaic cell. The catalog includes right
ascension and declination (B1950.0), K and I magnitudes, number of measurements, V
magnitude, spectral types, cross identifications to the numbering systems of the
General Catalogue, the Durchmusterung catalogs, the Bright Star Catalogue, and
star names.</field>
    <field name="keywords">Photometry, infrared; Surveys</field>
  </doc>
</add>
```

Illustration 12: Exemple d'un Readme « traduit » en XML pour Solr

3) *Les outils de tests*

Afin de visualiser les résultats de chacun des moteurs Solr et ElasticSearch, j'ai trouvé différents outils permettant celle-ci de manière simple et efficace.

J'ai en effet trouvé le plugin « head » pour ElasticSearch offrant une interface Web assez simple d'utilisation m'ayant permis de jongler avec différents paramètres de recherche que le moteur proposait et d'en visualiser le résultat.

Un exemple de cette interface est présent à l'annexe « B ».

Solr quant à lui dispose déjà de deux interfaces web codées en JavaScript et dont le code source était présent dans le dossier d'exemple.

La première est l'interface « Solr admin » permettant d'administrer Solr. Elle comprend plusieurs menus dont l'un pour la recherche, d'autres pour visualiser les fichiers de configuration dont je parlerai dans la quatrième partie, et également divers menus donnant des informations sur le journal des erreurs, ou encore sur l'état de santé des cœurs. Pour une visualisation de cette interface,

veuillez vous référer à l'annexe « C ».
La deuxième interface est, elle, orientée utilisateur. En effet, elle dispose d'un champ où taper sa recherche, une liste des documents correspondants trouvés et différentes menus avec des listes correspondants aux différents composants dont je parlerai dans la quatrième partie de ce rapport.
Il est possible de visualiser cette interface à l'annexe « D ».

4) Résultats

D'un point de vue du temps de réponse, on voyait les résultats presque instantanément sauf pour la première requête qui mettait toujours un peu plus de temps pour afficher le résultat (pour plus de détails, se référer au paragraphe « Temps de réponses » de la 5eme partie intitulée « Prototypage »).
En testant différentes requêtes (voir à la partie documentation de la sitographie page « 41 »), je me suis rapidement rendu compte que les différents moteurs n'affichaient pas le même résultat pour une même requête ce qui me parut surprenant et déroutant.
J'ai donc décidé d'approfondir mes connaissances du fonctionnement de ces moteurs.

V) Fonctionnement et configuration

En effet, il devenait nécessaire de comprendre le fonctionnement de ces moteurs de recherche. Malheureusement, le manque temps me poussa à devoir choisir un seul moteur de recherche à approfondir aux niveaux fonctionnement et configuration.

1) Le choix du moteur

Mon choix s'est porté sur Solr pour plusieurs raisons :
Tout d'abord, ElasticSearch et Solr étant tous les deux basés sur Lucene, il aurait peut-être paru judicieux de choisir Lucene. Mais Lucene manquait de beaucoup de fonctionnalités dont disposaient Solr et ElasticSearch comme la notion de champs. De plus, il semblait moins malléable que ces deux plate-formes. J'ai donc écarté Lucene.
Ensuite il m'a fallu choisir entre ElasticSearch et Solr. L'une des différences notables est qu'ElasticSearch nécessitait l'envoi via des requêtes HTTP certaines configurations au format JSON au niveau des champs et de la manière de les indexer. Cela n'étant pas très pratique, j'ai donc choisi Solr qui dispose de deux fichiers de configurations en XML et semblait plus simple à configurer ainsi.

2) Analyse de l'indexation et de la recherche

Il existe deux parties très importantes de la configuration qui assurent le bon fonctionnement des moteurs de recherche basés sur Lucene : l'indexation et la recherche. A chacune de ces parties est réservé un analyseur, et ceci est un concept de base de Lucene qui est donc présent à la fois dans Solr et dans ElasticSearch. Dans le cas de Solr, ils sont définis dans un fichier nommé « Schema.xml ».

2.1) Structure d'un analyseur

Voici la structure globale d'un analyseur :

- Dans le cas de Solr, on peut commencer par définir un « char filter » ou un filtre sur les caractères afin de filtrer les caractères selon les besoins. Il est possible de mettre plusieurs filtres de ce type ou de ne pas en mettre.
- Dans un second temps, on définit la manière dont on va découper le texte ce qui correspond au « tokenizer ». Il est présent à la fois chez Solr et ElasticSearch, et l'un des plus utilisés est le « StandardTokenizer » (présent de base dans Lucene donc dans ces deux plateformes) du fait qu'il découpe le texte en enlevant les espaces et la ponctuation ne gardant ainsi que les mots. Il en existe d'autres moins « agressifs » (par exemple découper uniquement sur les espaces) ou d'autres plus agressifs. On ne peut en définir qu'un seul, il faut donc le choisir précautionneusement en fonction de ses besoins.
- Enfin, il est possible d'ajouter à cet analyseur des « token filters » ou plus simplement des filtres qui permettront par exemple de supprimer des mots considérés comme inutiles à indexer (donc inutile à la recherche), le risque étant d'empêcher la recherche sur ces mots. Il existe une multitude de filtres, et il est possible d'en ajouter plusieurs pour un même analyseur. Néanmoins, il peut y avoir des problèmes de compatibilité c'est à dire que deux filtres peuvent ne pas donner ce qui était attendu, le tout dépendant de l'ordre dans lequel on les déclare et de ce que font ces filtres (l'un peut empêcher l'autre d'avoir un effet et vice-versa).

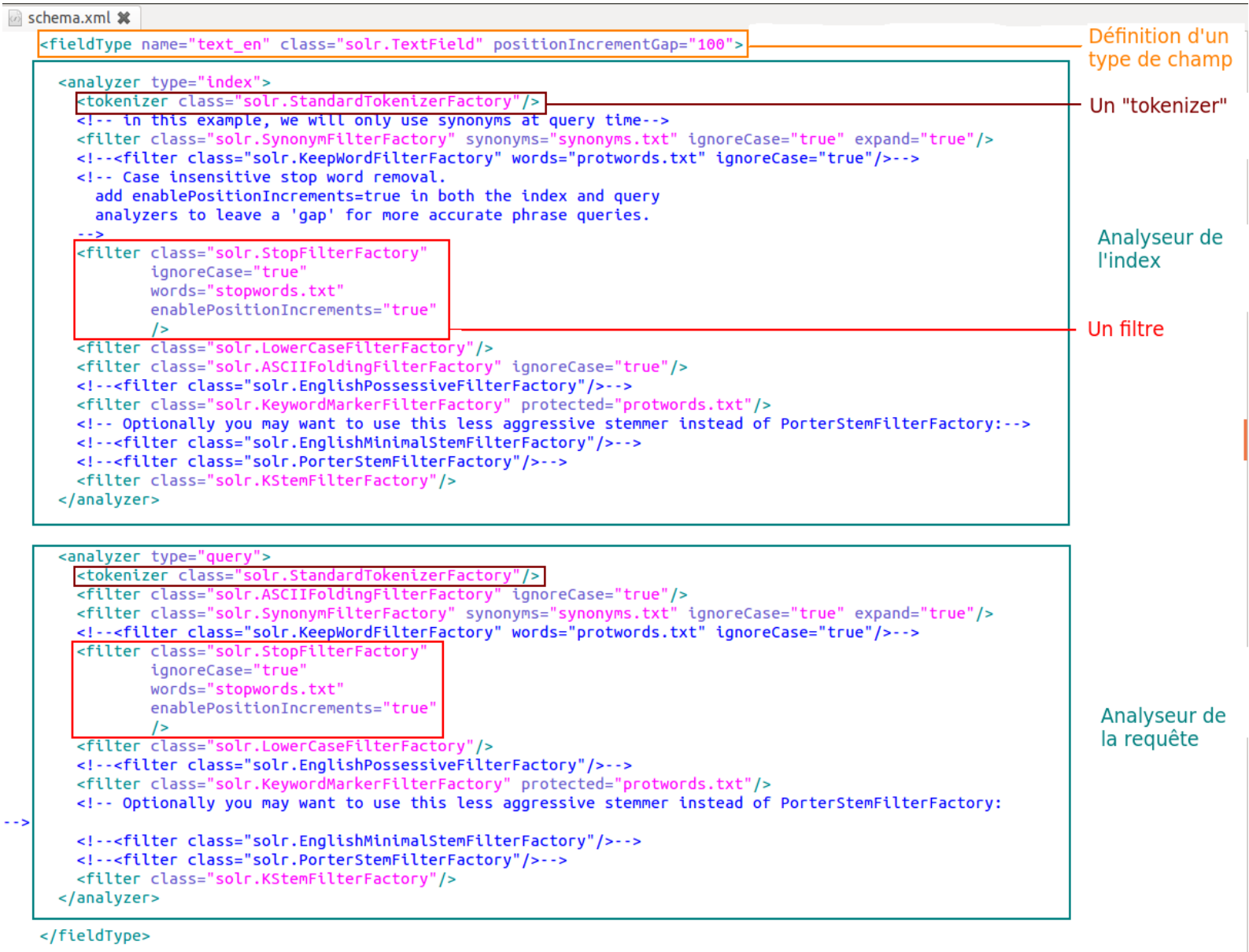


Illustration « 13 » : Ci-dessus, un exemple de définition d'un type de champ avec ses deux analyseurs

2.2) Application

Comme décrit plus avant, il y a un analyseur pour l'indexation et un autre pour la recherche. En vérité, il est possible dans Solr de ne définir qu'un analyseur qui sera alors appliqué dans les deux situation.

La différence entre ces deux analyseurs est que le premier, l'analyseur de l'index s'appliquera sur le contenu de ce que l'on indexe (dans mon cas par exemple : les « Readme ») tandis que l'analyseur utilisé lors de la recherche sera appliqué sur le contenu de la requête c'est à dire directement sur ce que l'utilisateur tapera dans la barre de recherche.

Dans tous les cas, il est souvent plus pertinent d'avoir les mêmes analyseurs pour l'indexation et la recherche de manière à ce que les deux puissent se correspondre. Par exemple, il ne serait pas judicieux d'utiliser un filtre mettant toutes les lettres en minuscules lors de l'indexation et ne pas mettre ce filtre lors de la recherche, car si l'utilisateur tapait en majuscule, il n'obtiendrait aucune réponse quand bien même il existerait le mot orthographiquement parlant.

2.3) Différence entre Solr et Lucene

La différence entre Solr et Lucene (qui est aussi le cas entre ElasticSearch et Lucene) au niveau des analyseurs est que l'on peut en définir selon les champs indexés.

En effet, Lucene ne demande pas et ne permet pas de recherche structurée par des champs. C'est là un avantage de Solr (et également d'ElasticSearch) car comme vous pouvez le constater sur l'illustration « 8 » (illustration « 7 » pour ElasticSearch), il y a plusieurs champs (title, original_title..., keywords) correspondant aux champs que l'on retrouve dans les « parfiles » (voir l'annexe « A »).

C'est également dans le fichier Schema.xml que l'on définit les champs pouvant exister au sein d'un document. D'ailleurs, un document se doit de n'avoir que des champs définies dans ce fichier sans quoi une erreur apparaîtra.

A chacun de ces champs, on administre un type champ et c'est dans la définition de ce type que l'on définit les analyseurs. Ainsi, tous les champs d'un type par exemple « text_en » verront leur contenu traité par les analyseurs définis dans le type de champs « text_en ». Pour une visualisation concrète voir l'illustration « 9 ». ElasticSearch utilise un système similaire.

Ci-dessous, un exemple de définition de champs.

schema.xml

```
   Some fields are multiValued only because Tika currently may return
   multiple values for them. Some metadata is parsed from the documents,
   but there are some which come from the client context:
     "content_type": From the HTTP headers of incoming stream
     "resourcename": From SolrCell request param resource.name
-->
<field name="title" type="text_en" indexed="true" stored="true"/>
<field name="identificateur" type="text_general" indexed="true" stored="true"/>
<field name="date_creation" type="date" indexed="true" stored="true"/>
<field name="date_modification" type="date" indexed="true" stored="true"/>
<field name="original_title" type="text_en" indexed="true" stored="true"/>
<field name="description" type="text_en" indexed="true" stored="true"/>
<field name="authors" type="text_general" indexed="true" stored="true"/>
<field name="keywords" type="text_en" indexed="true" stored="true"/>

<field name="subject" type="text_general" indexed="true" stored="true"/>
<field name="comments" type="text_general" indexed="true" stored="true"/>
<field name="category" type="text_general" indexed="true" stored="true"/>
<field name="resourcename" type="text_general" indexed="true" stored="true"/>
<field name="url" type="text_general" indexed="true" stored="true"/>
<field name="content_type" type="string" indexed="true" stored="true" multiValued="true"/>
<field name="last_modified" type="date" indexed="true" stored="true"/>
<field name="links" type="string" indexed="true" stored="true" multiValued="true"/>
```

Illustration « 14 » : Exemple de définition de champs dans le fichier « Schema.xml »

3) Les paramètres de recherche

En plus de la manière de traiter l'indexation et la requête, il existe également un bon nombre de paramètres de recherche. De plus, il faut différencier deux types de paramètres : les paramètres de recherche donnés par l'utilisateur (ou syntaxe des requêtes) et les paramètres de requêtes qui sont définis dans un autre fichier de configuration : « solrConfig.xml ».

3.1) La syntaxe des requêtes

Tout d'abord, il est possible pour n'importe quel utilisateur du moment que celui-ci en est averti, d'utiliser la syntaxe déjà existante dans le moteur Lucene et réutilisé par la plateforme Solr (et ElasticSearch également) a quelques différences près.
Par exemple, tout comme vous pouvez le faire dans une barre de recherche Google, il est possible de mettre un signe « + » devant un mot ou une phrase pour dire « Cette phrase doit être présente dans les documents ». A l'inverse, vous pouvez mettre un signe « - » pour dire « Cette phrase ne doit pas être présente dans les documents ».
En revanche, comme expliqué dans la partie « Les démonstrations testées », Lucene accepte les « boost » négatif tandis que Solr ne les accepte pas.
Sur le site mentionné dans la sitographie partie documentation page « 41 » vous verrez toutes les syntaxes possible d'une requête écrite par l'utilisateur.
Ceux-ci n'ont rien à voir avec les paramètres de requêtes décris ci-après.

3.2) Les paramètres de requêtes de Solr

Les paramètres de requêtes sont définis, comme dit un peu plus haut, dans le fichier « solrConfig.xml ».
Ils sont définis dans des « requestHandler » ou manipulateur de requêtes. Il est possible de définir plusieurs manipulateurs qui auront des valeurs associés à ces paramètres différentes, ou bien en avoir moins ou plus, le tout dépendant des besoins. Il existe également plusieurs type de manipulateur de requêtes mais ceux dont je parlerai sont les plus utilisés.
Ils sont appelés « SearchHandler » ou « manipulateur de recherche ». Ils s'occupent de la gestion de la recherche. On y définit les paramètres qui seront appliqués par défaut lors de la recherche comme par exemple quel requête sera envoyée dans le cas d'un champ de requête laissé vide par l'utilisateur (C'est à dire s'il ne tape rien). C'est également dans ces « requestHandler » que l'on définit les paramètres des composants.

3.3) Les composants de recherche de Solr

Les composants sont définis dans ce même fichier « solrconfig.xml », mais leurs paramètres peuvent varier selon les différents « requestHandler », c'est pourquoi ce sont dans ces derniers que l'on y définit leurs paramètres.
Il existe environ 8 composants ayant chacun des buts totalement différents. Par exemple, l'un des composants permet de mettre en gras les mots ayant été trouvés. Un autre permet de proposer une correction en fonction des mots présents dans l'index et ce qu'avait tapé l'utilisateur. Ce dernier semblait tout approprié à l'un des besoins du CDS.

4) Le score

Pour trier les documents, Solr utilise un système de score (qui est de base chez Lucene et donc chez ElasticSearch). Bien sûr le score est calculé à chaque recherche et pour chaque document qui répondront aux paramètres de cette recherche.
Il existe différents paramètres ayant une influence sur ce score. Par exemple, le plus simple à comprendre et auquel on s'attend est bien entendu le nombre de fois où le mot ou les mots présents dans la requête apparaissent dans un document. Un autre est le nombre total de fois qu'apparaît le mot recherché dans l'index entier. Dans ce cas, plus un mot est rare dans l'index, plus le score du document le contenant sera élevé.
Pour une liste plus exhaustive des paramètres existants voir le site partie documentation de la sitographie page « 41 ».

Il existe également d'autres moyens directement à la portée de l'utilisateur comme les « Boost ». Il s'agit d'un paramètre qu'un utilisateur peut mettre (voir le lien de la syntaxe d'une requête) qui permet de donner une importance plus grande ou plus petite à un mot ou un champ dans la recherche. Ainsi, il est possible d'influencer le score des documents et donc d'obtenir un résultat plus pertinent qui répond mieux aux besoins de l'utilisateur.

5) La notion de cœur

Une autre notion que je n'ai pas eu le temps de réellement approfondir est la notion de cœur. Lucene utilise déjà cette notion de base. Un cœur correspond dans Solr à un index et à une configuration (Contenant toute la configuration dont j'ai parlé dans ce chapitre).

Dans le cas de mon stage, un seul cœur suffisait étant donné que je ne devais tester que sur un seul type de fichier à savoir les « Readme » et que je ne disposais pas d'un temps suffisant pour pouvoir tester toute la profondeur de configuration de Solr.

Néanmoins, si l'on souhaite avoir plusieurs index séparés afin de disposer de « plusieurs » moteurs de recherches ayant chacun un type de fichier indexé différents. Par exemple on pourrait avoir un cœur contenant l'index des « Readme » et un autre qui contiendrait l'index des objets astronomiques.

En vérité, il n'y aurait qu'un moteur, Solr, mais il serait utilisé de deux manières différentes, dépendantes du cœur sur lequel on souhaite faire notre recherche.

Solr permet également de faire de la recherche « multi-coeur » mais n'ayant pas eu le temps de tester ce genre de cas de figure, je ne ferai qu'en faire allusion pour informer les personnes susceptibles d'être intéressé par cette capacité.

Avec cette notion pourrait également venir la notion de cluster (possibilité de dispatcher la recherche sur plusieurs machines) mais là encore, je n'en ferai qu'allusion.

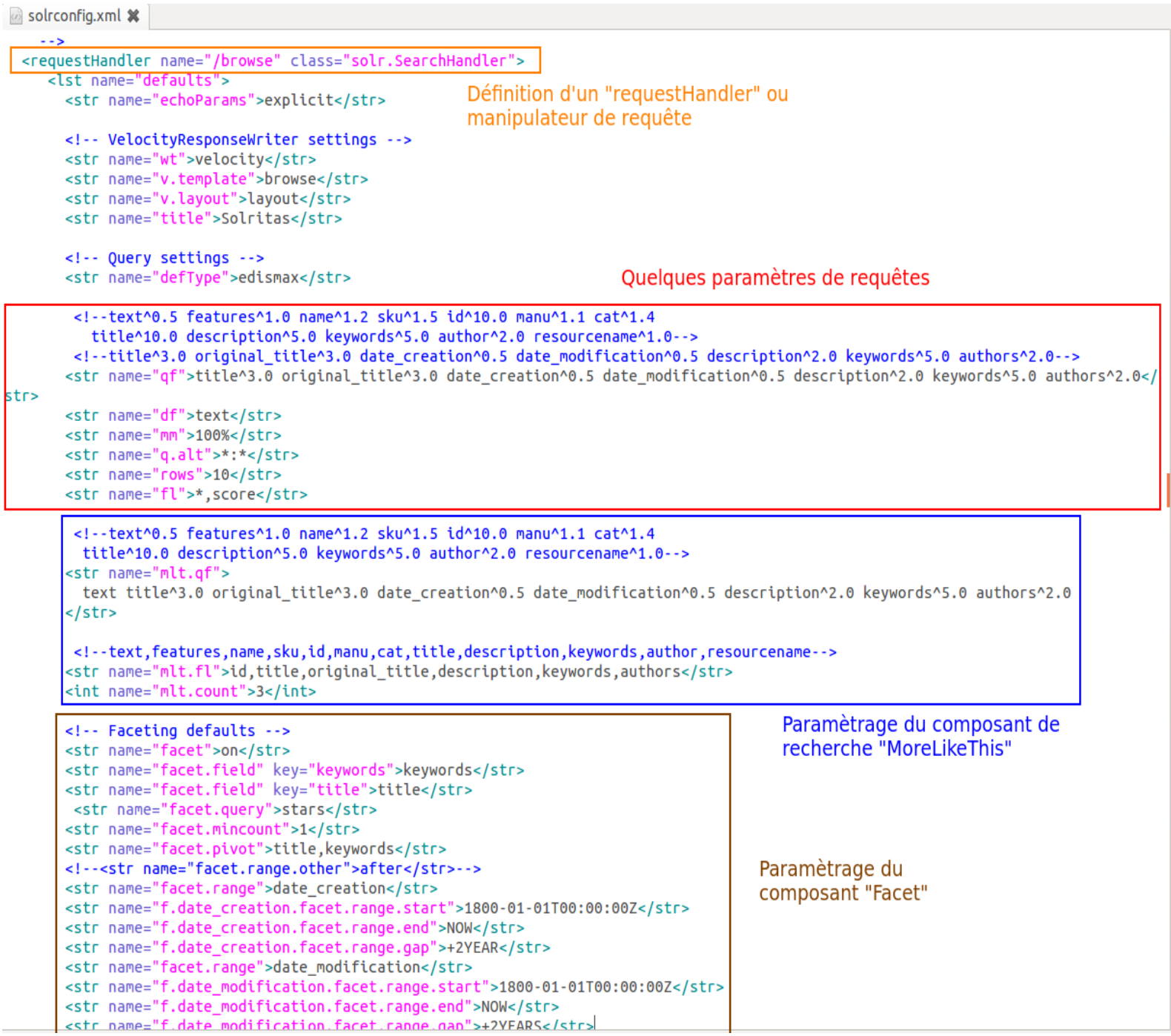


Illustration « 15 » : Exemple d'un « requestHandler » dans le fichier de configuration « solrconfig.xml »

VI) Prototypage

Après avoir obtenu une recherche assez pertinente à partir des éléments présents dans la démonstration, une autre question s'est posée : Comment fonctionne alors la librairie Solr et surtout comment l'utiliser ?

1) Le but de la réalisation d'un prototype

Dès lors, il a semblé nécessaire de réaliser un prototype qui permettrait de tester les différentes classes présentent dans cette librairie. C'est donc le premier but de ce prototype.
Le prototype réalisé a également servi à connaître les temps de réponses car les services actuels du CDS sont très rapides (les programmes ont été taillés sur mesure à la différence de Solr qui est plus général), il fallait donc avoir une idée des performances de Solr pour vérifier qu'il n'empêcherait pas la fiabilité du service et le confort des utilisateurs.

2) Une interface Java

Dans un premier temps, je devais réaliser une interface Web en JavaScript mais j'ai dû renoncer à cela pour deux raisons principales et liées : Tout d'abord, je n'avais que peu de connaissances en JavaScript et JQuery, cela m'aurait donc demandé du temps pour apprendre convenablement ce langage, et c'est ici qu'intervient la seconde raison : je ne disposais pas du temps nécessaire pour apprendre et savoir utiliser le JavaScript.
C'est pourquoi, j'ai demandé à faire une interface en Java, de manière à pouvoir atteindre les buts définis dans le paragraphe précédent. Ci-dessous quelques images de l'interface graphique réalisée.

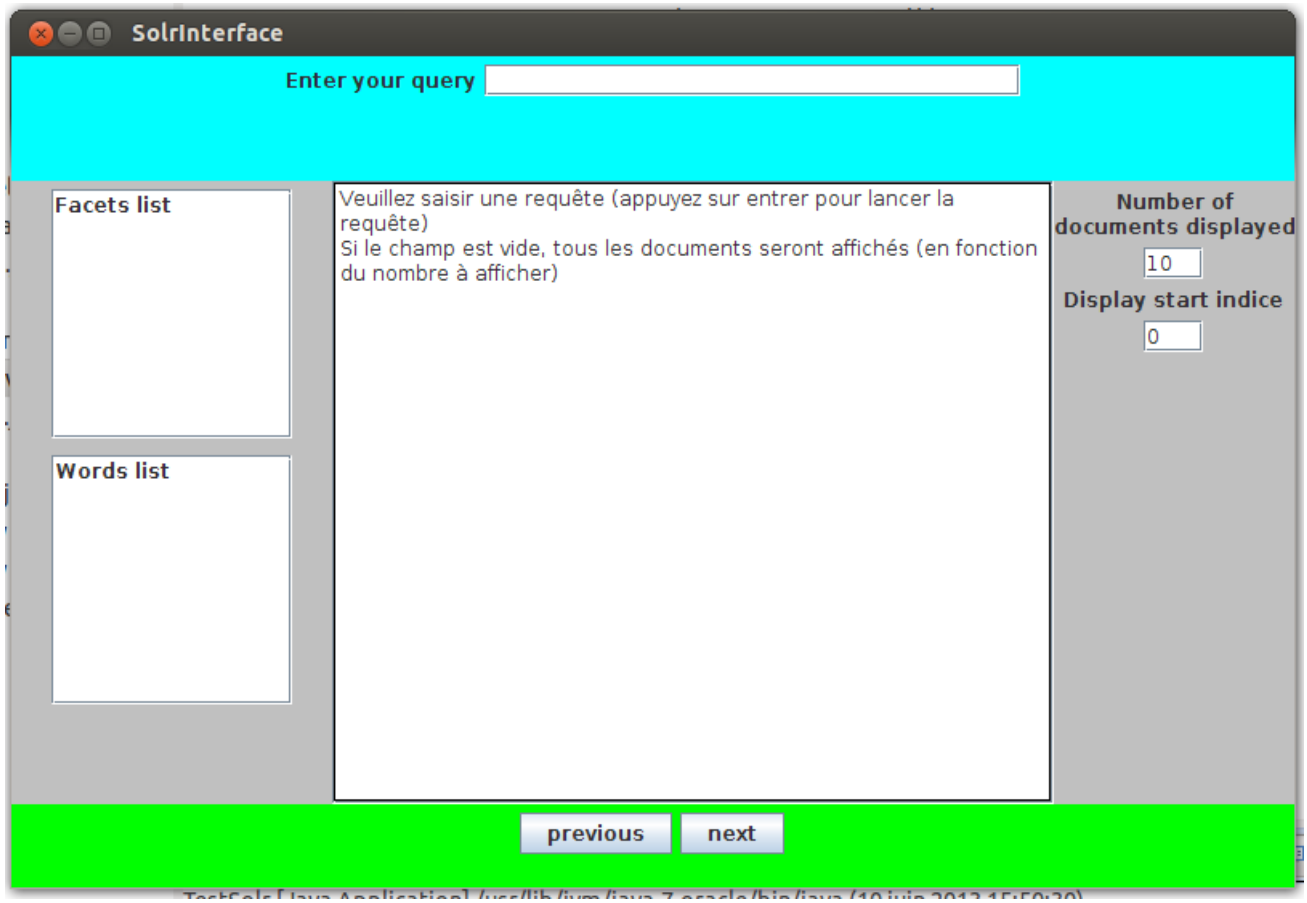


Illustration 16 : Une image de l'interface graphique réalisé lors de son état initial

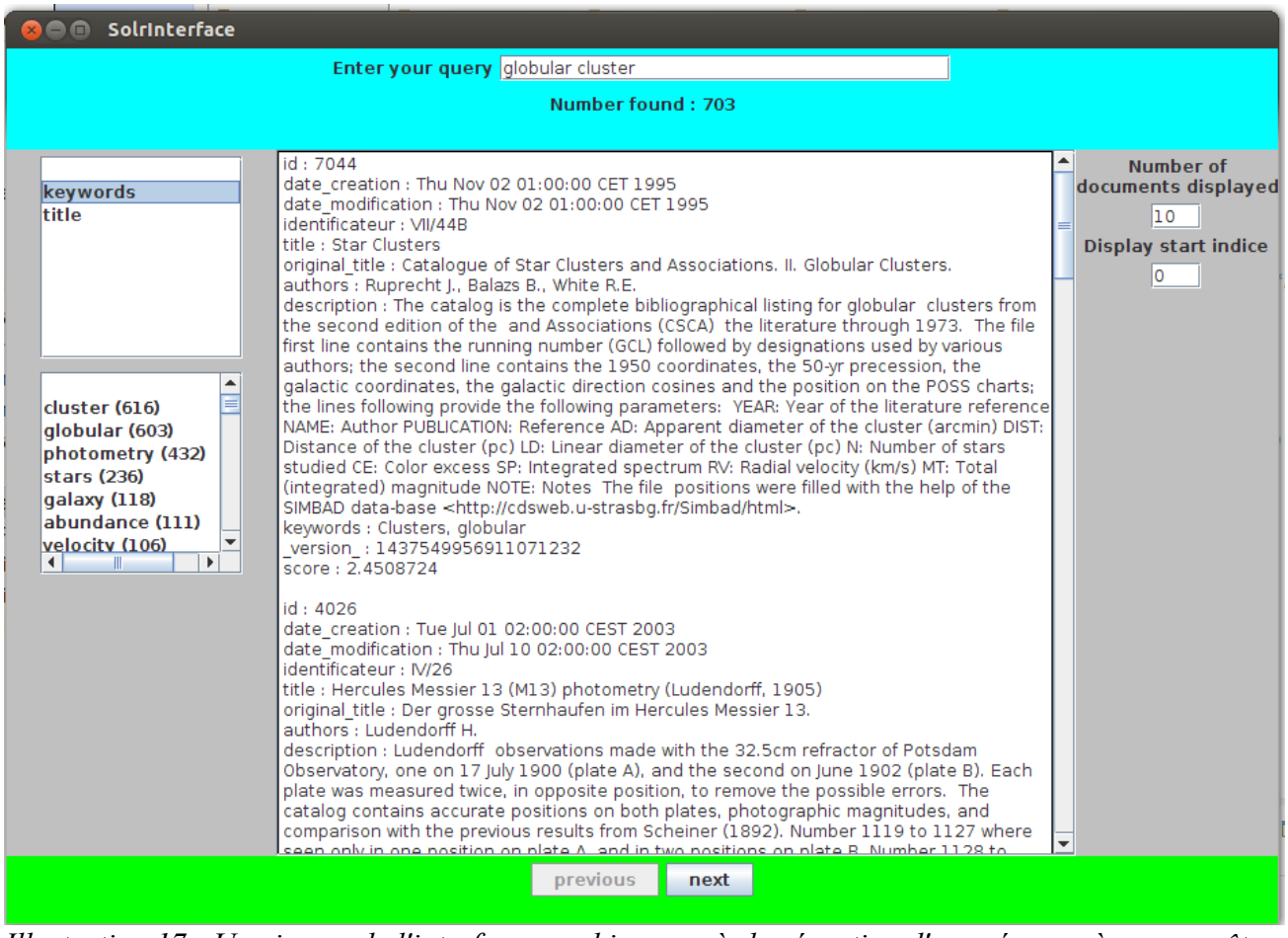


Illustration 17 : Une image de l'interface graphique après la réception d'une réponse à une requête

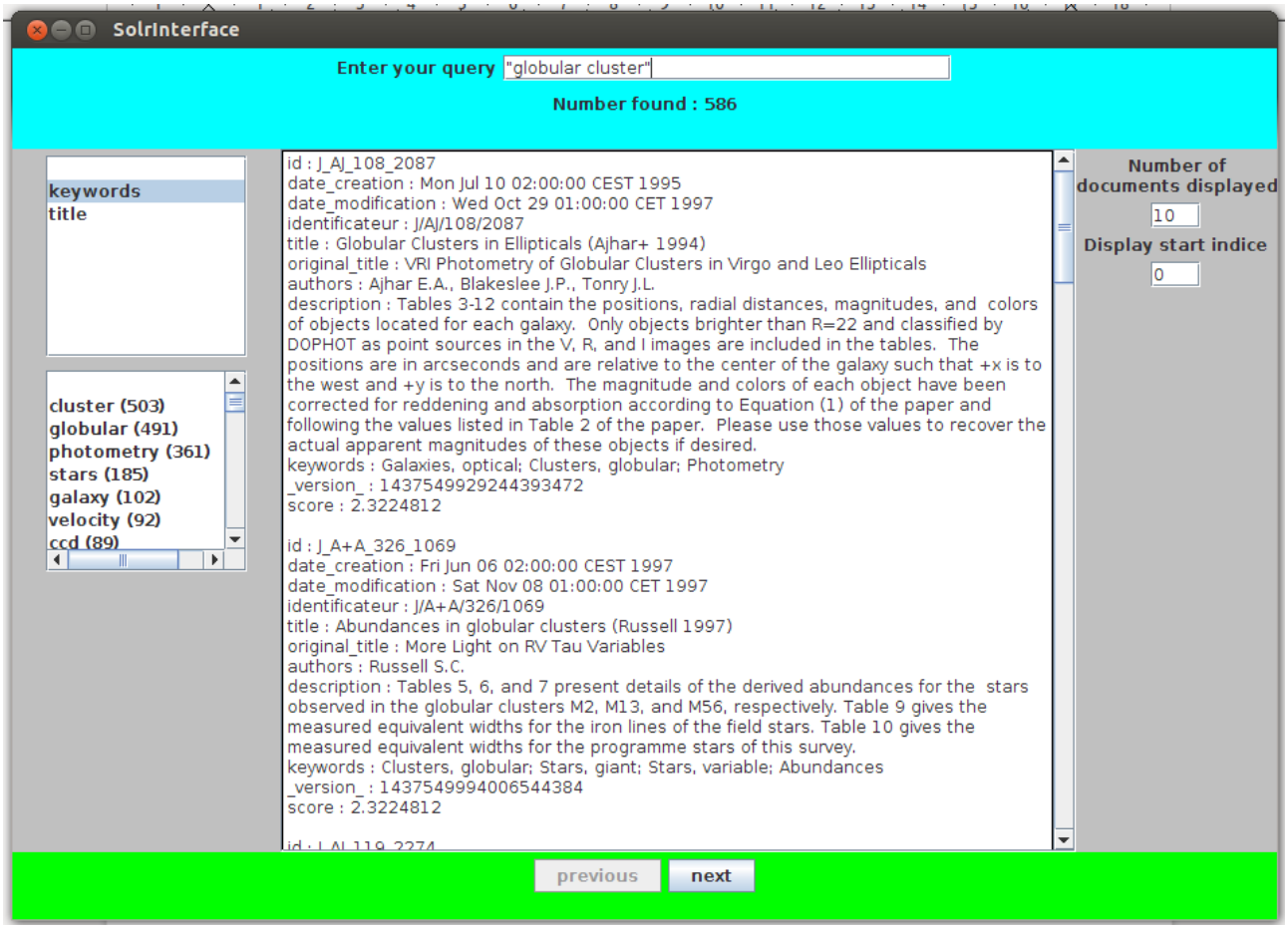


Illustration 18 : Une image de l'interface graphique répondant à une requête plus spécifique

Sur les deux dernières images, vous pouvez constater la différence de résultats entre une requête banale ne comportant que deux mots et cette même requête mais plus spécifiques, utilisant les guillemets pour indiquer une phrase à rechercher.

Les deux listes situées sur la droite correspondent aux facettes (l'un des composants dont j'ai fais allusion dans le paragraphe « Les composants de recherche de Solr » page « 27 »).

3) Temps de réponses

Un peu plus avant dans ce rapport, j'ai fais allusion aux temps de réponses et dans le premier paragraphe de ce chapitre, j'ai parlé de l'importance de celui-ci. Je vais donc vous expliquer globalement les temps de réponses obtenus en gardant en tête que chaque requête a été exécutée 100 fois d'affilée.

Dans un premier temps quelque soit la première requête après avoir initialisé le serveur et le cœur, celle-ci met plus d'une seconde à renvoyer une réponse. Ceci est dû à la mise en mémoire de l'index (en fonction de la requête).

Ensuite, plus on fait cette requête, plus le temps de réponse à celle-ci diminue jusqu'à une sorte de point d'équilibre où le temps de réponse reste constant à quelques millisecondes près.

Il faut également savoir que le temps de réponse varie selon la requête. En effet, une requête avec peu de mot et donc plus général entraînera un point d'équilibre à valeur élevé c'est à dire un temps de réponse plus grand qu'une requête avec beaucoup de spécifications (beaucoup de mots).

Pour un exemple concret, le point d'équilibre pour une requête ne contenant que le mot « test » varie entre 75 et 85 millisecondes environ. Pour une requête contenant les trois mots « test black hole » varie entre 20 et 40 millisecondes environ.

Ces nombres ne sont absolument pas fixes. Ils peuvent varier en fonction des requêtes faites précédemment, la taille du cache et le nombre d’occurrence du mot dans l'index entier.

VII) Conclusion

L'objectif du stage était donc l'étude de moteurs de recherche et le prototypage d'un service dans le but d'observer si l'amélioration du service de recherche existant proposé par le CDS pourrait être envisagée.
J'ai donc dans un premier temps établi un État de l'Art sur les différents outils permettant la mise en place d'un moteur de recherche à savoir pour les plus connus et les plus utilisés : Solr et ElasticSearch.
J'ai ensuite pu les tester, tout d'abord avec les démonstrations proposés par les communautés s'occupant des mis à jour et de la maintenance de ces moteurs puis sur des données réelles du Centre de Données astronomiques de Strasbourg.
Mais rapidement, j'ai dû approfondir ma compréhension de ces moteurs du fait de la complexité de ceux-ci pour l'obtention de la recherche la plus pertinente possible. J'ai donc appris à configurer les paramètres principaux et les plus importants du moteur Solr (pour être plus précis, le moteur Lucene mais avec la plate-forme Solr).
Enfin, j'ai réalisé un prototype afin de découvrir dans la structure même du code la mise en place d'un moteur de recherche basé sur cette plate-forme et son API.

Néanmoins, il reste beaucoup d'amélioration à apporter. En effet, tout d'abord au niveau de la configuration qui n'est pas si aisée que l'on pourrait le penser étant donné la quantité de paramètres modifiables, car la configuration actuelle bien qu'ayant une certaine pertinence est encore loin d'être parfaite. De plus, la documentation de Solr n'est pas achevée et Solr lui-même n'est pas achevée : il est apparu une nouvelle version au cours de mon stage mais j'ai dû rester dans la version de départ afin d'éviter tous problèmes de compatibilité. En ce qui concerne l'interface, il existe certains bugs qui n'ont pas encore été corrigés. Il manque également beaucoup de paramètres que l'on devrait permettre aux utilisateurs de modifier. La réalisation d'une page administrateur pour modifier le cœur peut être aussi envisagée .

Ce stage m'aura apporté une grande quantité de connaissance sur le fonctionnement des moteurs de recherche et sur la complexité de ceux-ci mais il m'aura également donné un avant goût de ce qu'est ce qui est populairement appelé le « Big Data ». Il m'aura également apporté une bonne expérience en entreprise avec des responsabilité même si minimes.

VIII) Références Bibliographiques

Site de téléchargements des différentes plate-formes ou moteurs :

http://lucene.apache.org/ : Site officiel du projet (ou moteur) Apache Lucene

http://lucene.apache.org/solr/ : Site officiel du projet (ou plate-forme) Apache Solr

http://www.elasticsearch.org/ : Site officiel du projet (ou plate-forme) ElasticSearch

http://cds.u-strasbg.fr/doc/catstd-3.1.htx : Site d'explication de la structure d'un Readme

http://katta.sourceforge.net/ : Site officiel du projet (ou plate-forme) Katta

Documentations :

http://wiki.apache.org/solr/ : Wiki officiel de la documentation de Solr

http://docs.lucidworks.com/display/solr/About+This+Guide : Site de documentation de Solr

http://g-rossolini.developpez.com/tutoriels/solr/ : Tutoriel en français pour Solr

http://www.solrtutorial.com/solr-query-syntax.html : Résumé des syntaxes de requête possibles

Service du CDS :

http://vizier.u-strasbg.fr/viz-bin/VizieR : Service VizieR du CDS (recherche dans les catalogues)

IX) Glossaire

CDS : Centre de Données astronomiques de Strasbourg, travaillant à l'observatoire.

ElasticSearch : Projet open source de Shay Bannon, plateforme logicielle pour le moteur Lucene.

Hadoop : Projet open-source de la communauté Apache, outils servant à manipuler de grande quantités de données à l'aide de l'informatique distribuée.

HTTP : « HyperText Transfer Protocol », protocole de base de l'Internet pour le transfert de données.

Java : Langage de programmation connu.

JavaScript : Langage de programmation servant à dynamiser les pages web.

JSON : « JavaScript Object Notation », fichier ayant une structure particulière, utilisé pour la structuration de données. Utilisé surtout pour le Web.

Katta : Projet open-source, plate-forme logicielle pour le moteur Lucene.

Lucene : Projet open source de la communauté Apache, moteur d'indexation et de recherche.

Readme : Fichier texte, résumé d'un catalogue astronomique référencé par le CDS.

Solr : Projet open source de la communauté Apache, plate-forme logicielle pour le moteur Lucene.

XML : « eXtensible Markup Language », modèle de structuration de données dans de simples fichiers texte à l'aide de balises « < » et « > » et d'attributs entre ces deux balises.

X) Annexes

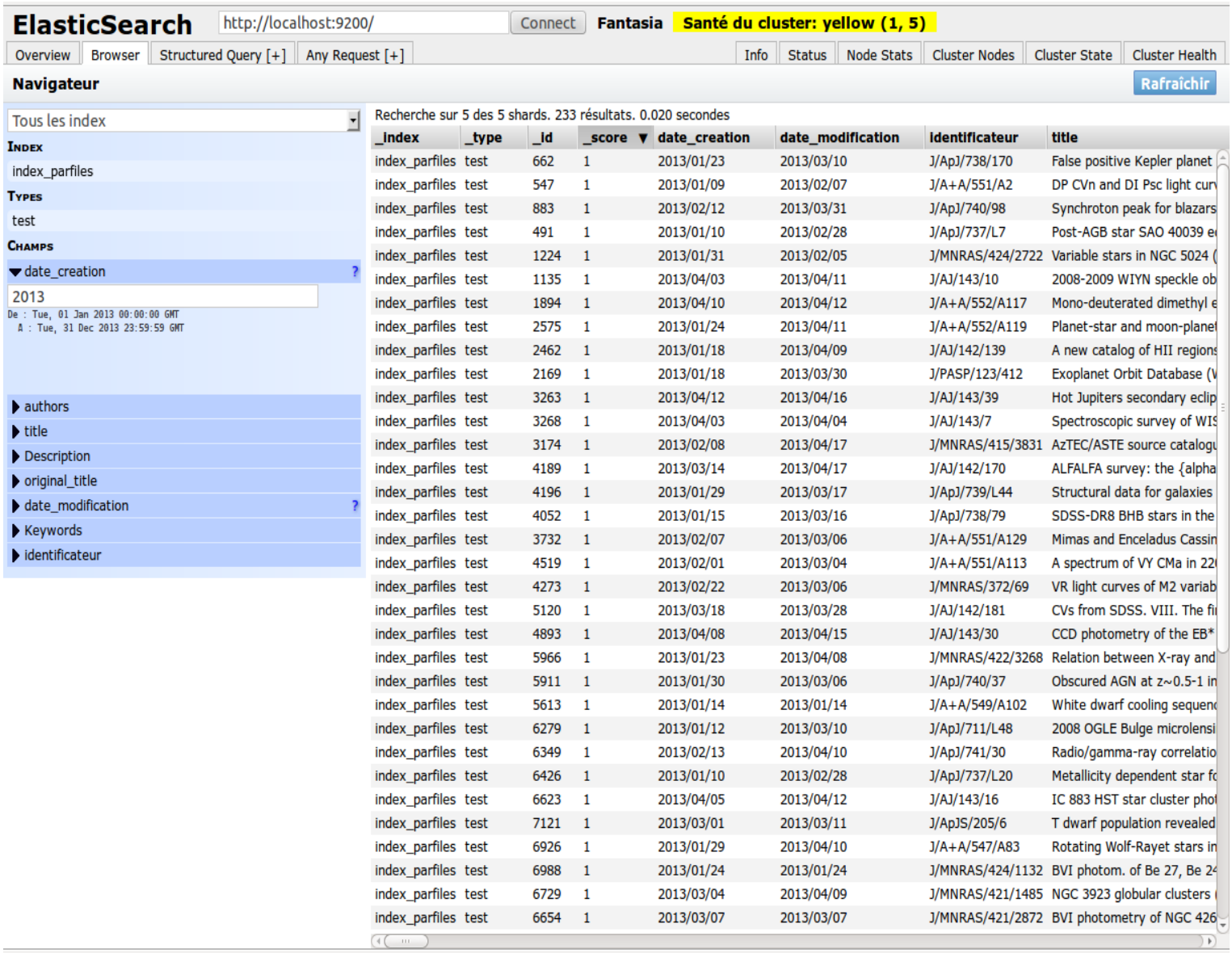
Annexe A :

Exemple d'un fichier « parfile »

```
%R 1997yCat.2002....0N
%D 1997-May-27
%d 2008.06.14
%I II/2B
%t Two-Micron Sky Survey (TMSS) (Neugebauer+ 1969)
%T The Two-Micron Sky Survey
%J 1969IRC...C......0N
%J 1969QB6.N47........
%A Neugebauer G., Leighton R.B.
%B The catalog, giving sources of emission in the 2.2-micrometer region
    for more than 5000 stars, represents a systematic survey of the
    Northern Hemisphere for stars brighter than third magnitude. The
    survey was carried out with a telescope at Mount Wilson, California,
    having a 62-inch diameter and an f/l aluminized epoxy mirror mounted
    equatorially. Radiation at an effective wavelength of 2.2 micrometers
    was detected by a lead sulfide photoconductive cell cooled by liquid
    nitrogen. In addition to the 2.2-micrometer detector array, radiation
    at an effective wavelength of 0.84 micrometers was detected by a
    simple silicon photovoltaic cell. The catalog includes right ascension
    and declination (B1950.0), K and I magnitudes, number of measurements,
    V magnitude, spectral types, cross identifications to the numbering
    systems of the General Catalogue, the Durchmusterung catalogs, the
    Bright Star Catalogue, and star names.
%K Photometry, infrared; Surveys
%F catalog.dat      5612x164  Catalog Data
%F chisq.dat        2574x36   Chi-square Excess
%F remarks.dat       140x78   Remarks
```

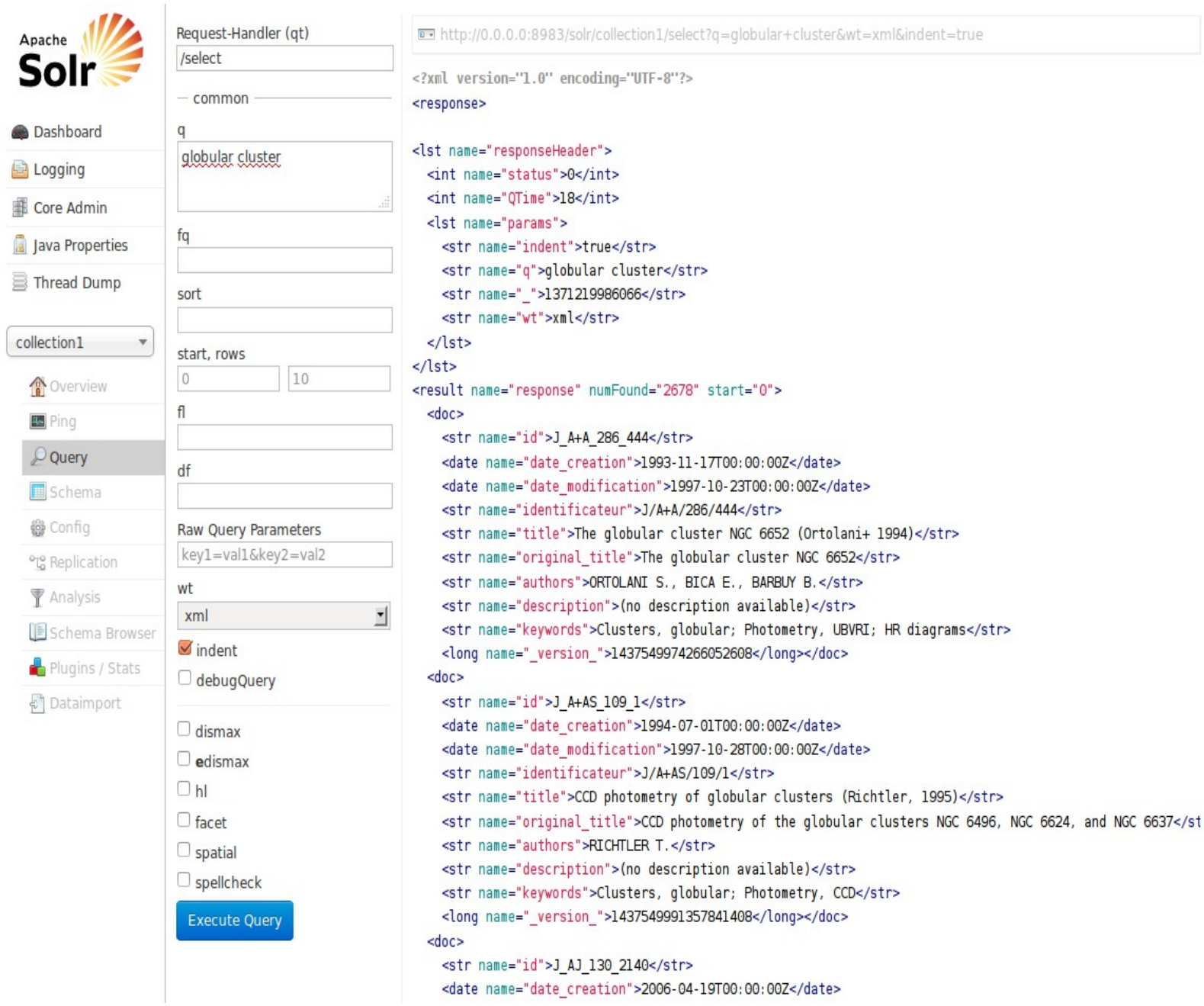
Annexe B :

Interface d'ElasticSearch



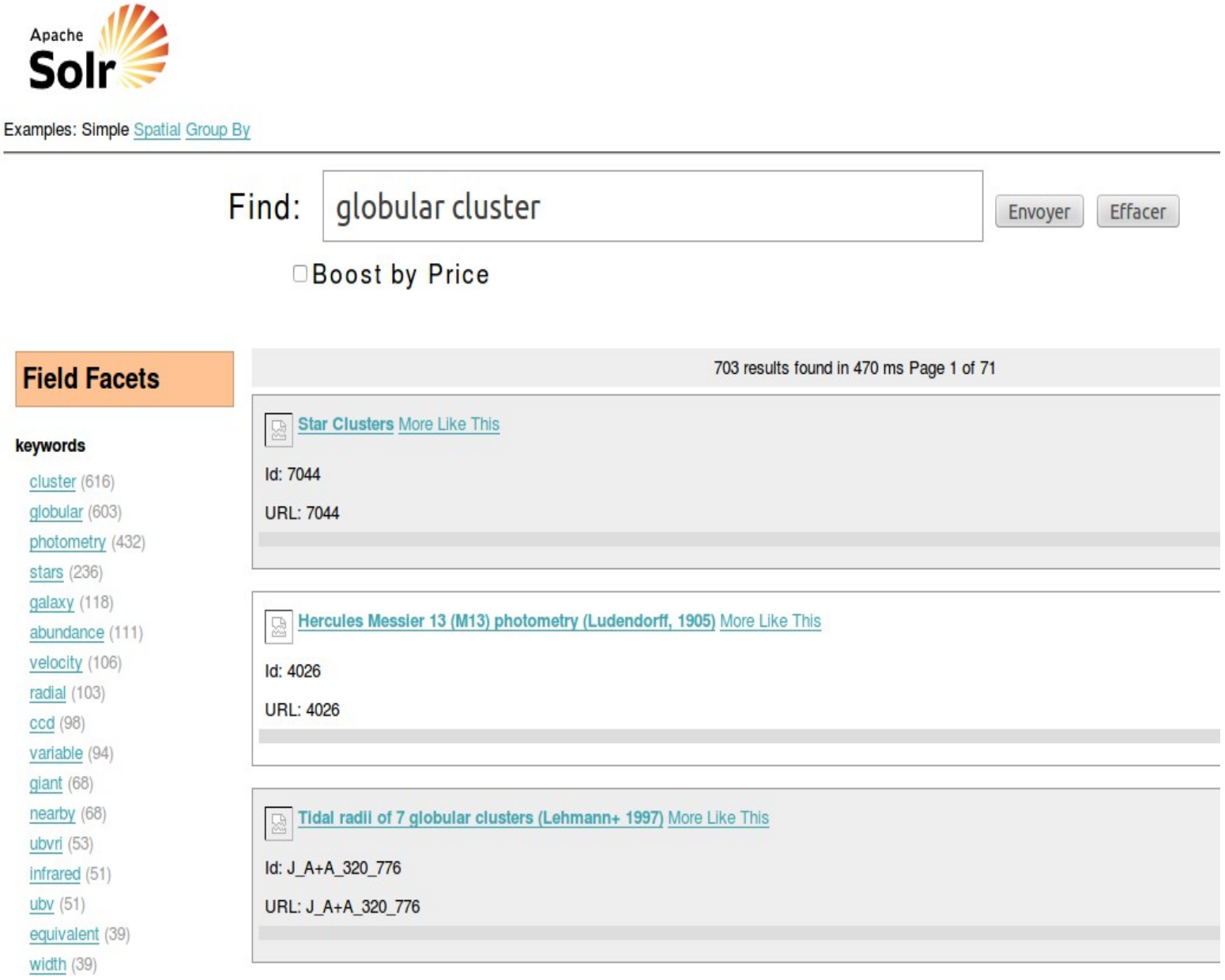
Annexe C :

Interface Administrateur de Solr



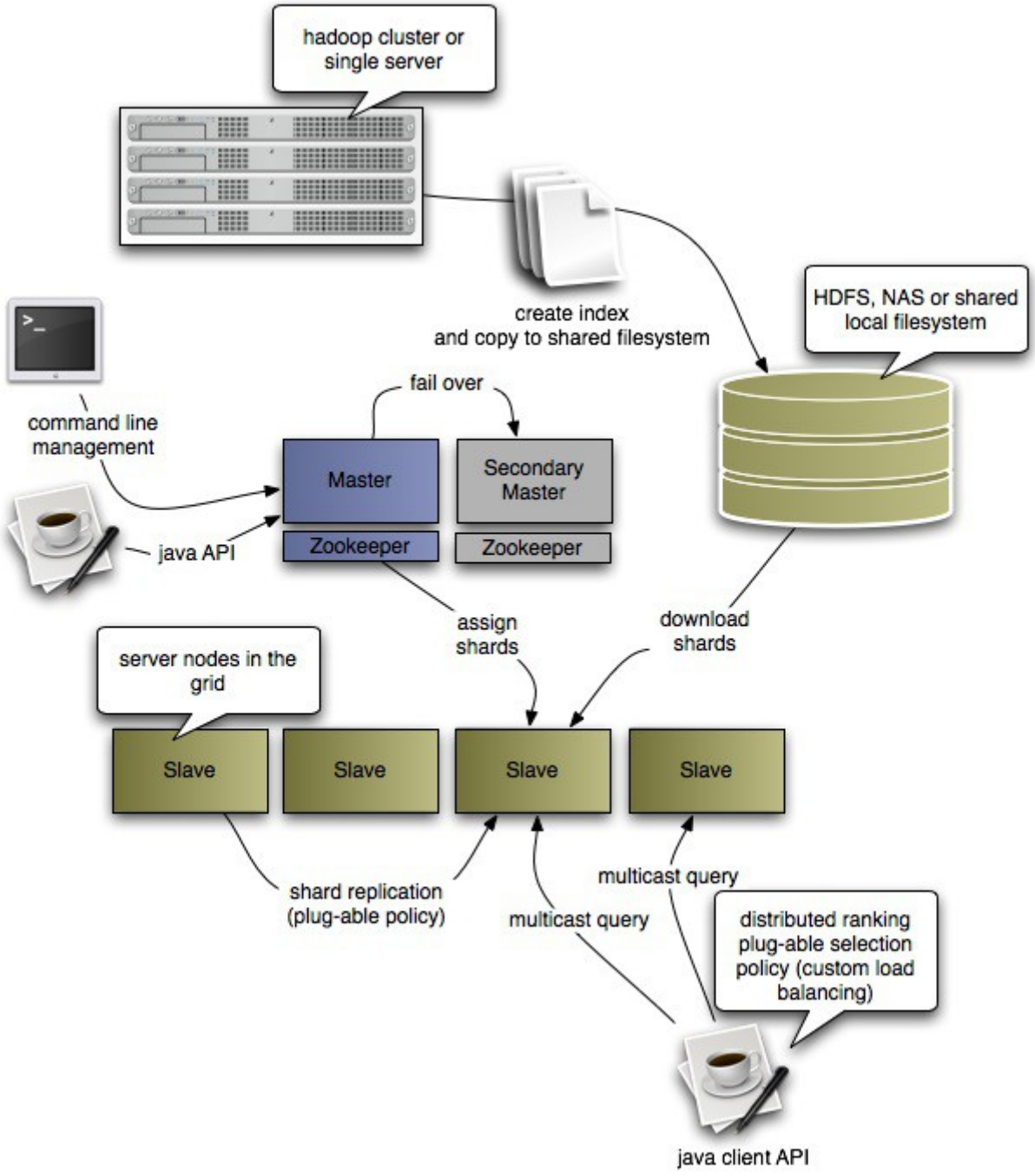
Annexe D :

Interface Utilisateur de Solr



Annexe E :

Schéma global du fonctionnement des moteurs de recherches présentés



Annexe F :

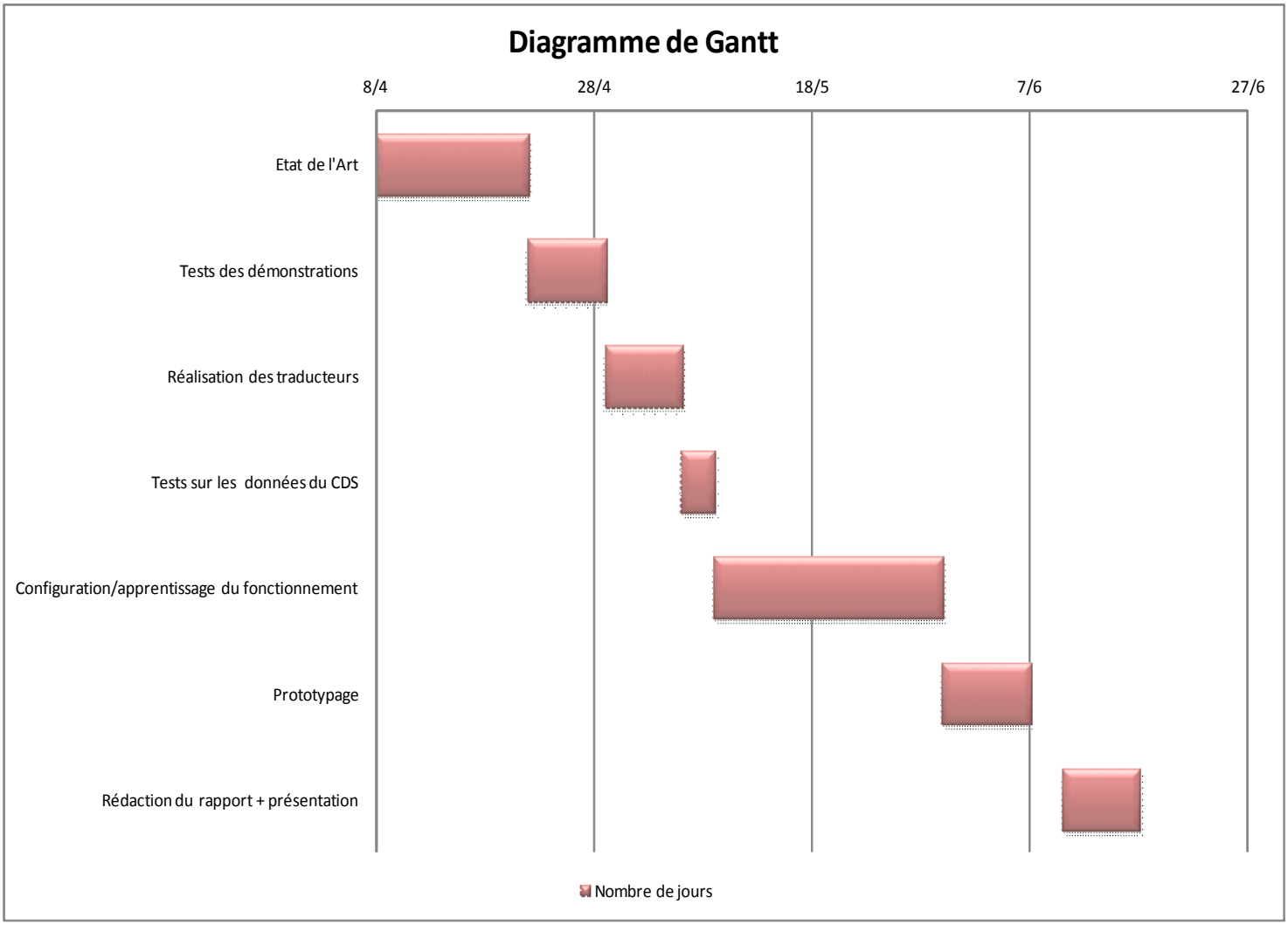


Diagramme de Gantt
8/4
28/4
18/5
7/6
27/6
Etat de l'Art
Tests des démonstrations
Réalisation des traducteurs
Tests sur les données du CDS
Configuration/apprentissage du fonctionnement
Prototypage
Rédaction du rapport + présentation
Nombre de jours