|  |  |
| --- | --- |
|  | **2011** |
|  | Leonhard hermle  Julien Cellier  Xavier Mourgues |

|  |
| --- |
| **[Rapport Projet tuteurè]** |
| Rapport du projet tuteuré XML exprimant les besoins, les objectifs et les moyens de mise en œuvre du projet ainsi que les problèmes rencontrés et le résultat atteint en fin de projet. |

Université Toulouse I – IUT de Rodez

Année 2010/2011

TABLE DES MATIERES

[REMERCIEMENT 1](#_Toc287254065)

[Présentation du sujet 1](#_Toc287254066)

[Les besoins 2](#_Toc287254067)

[Les contraintes 2](#_Toc287254068)

[L’univers de travail 2](#_Toc287254069)

[Répartition des tâches 2](#_Toc287254070)

[Java 2](#_Toc287254071)

[Base de données 3](#_Toc287254072)

[Restitution des données 3](#_Toc287254073)

[NAVIGATION 4](#_Toc287254074)

[Problèmes 6](#_Toc287254075)

REMERCIEMENTS

Nous souhaitons tout d’abord adresser nos remerciements à ceux qui nous ont permis de réaliser ce projet tuteuré, à ceux qui nous ont encadrés et qui nous ont rendus cette expérience agréable et enrichissante.

Nous remercions PATRICK SABI, notre professeur qui nous à encadré tout au long de notre projet et de nous avoir apporté des conseils.

Merci également à BRUNO BELIERE d’avoir mis à disposition deux bases de données Oracle qui nous ont permis de mener à bien notre projet.

Un grand merci à ALLOCINE qui autorise le parcours des pages HTML de son site afin de permettre la récupération de données textuelles.

# Présentation du sujet

L’objectif du projet est de restituer à partir de sites web existants des données de films (critiques, commentaires, notes, bande annonce, informations diverses…). Ainsi, les sites exploités seront ceux comportant des informations diverses pour un même élément recherché (un film en particulier). Les commentaires et les notes d’utilisateurs et des sites seront rassemblés permettant une vue d’ensemble sur le film recherché, tout ceci pouvant servir à comparer et fusionner les avis d’un film donné.

Plus simplement, il sera possible de regrouper les commentaires et notes restitués via une interface graphique développé en langage web, notamment en PHP. Cette partie du projet sera expliquée plus en détails dans le chapitre « Restitution de données ». Le parcours (parsing) des pages des sites web se fera en java.

Le projet se présente donc en 3 étapes :

* La **récupération de données** à travers divers sites web évolutif et dynamique (plus en détails dans le chapitre « JAVA »).
* Le **stockage de ces données** dans une base de données Oracle v10.2 (grâce à JAVA et plus en détails dans le chapitre «Base de données »).
* La **restitution visuelle des données** pour les utilisateurs (plus en détails dans le chapitre «Restitution de données »).

A l’origine, les sites web que nous avons sélectionnés étaient Allociné et IMDB. Malheureusement, se posait un souci d’autorisation. Ce problème est mis en avant dans le chapitre des problèmes en fin de rapport sous «IMDB». Par conséquent, le choix d’un second site autre qu’IMDB contenant une quantité importante de données de films, non négligeables. IMDB et l’une plus grande base de données de films mondiale rassemblant des informations sur tous les films disponibles, c’est pourquoi l’impossibilité d’exploitation de leur site internet est très regrettable étant donnée que la comparaison de données entre une sites de référence comme IMDB et un second peu montrer des différences importantes.

Nous aurons une première phase de découverte visant à étudier les différentes techniques de mise en œuvre d’un tel projet.

Plus tard, dans un souci d’évolution, plus d’information pourront être récupérer et d’autres sites dit « source d’information » pourront être rajoutés.

# Les besoins

Nous avons besoin de récupérer des données de sites web qui ne sont pas structuré en XML mais en HTML ce qui nous posera problème lorsque voudront traiter ces données. En effet, un site étant évolutif il est possible que sa structure HTML change. Pour parer cet effet, nous devrons réussir à soutirer uniquement les informations nécessaires et non les informations inutiles (publicités …). Pour cela l’utilisation de la technologie DOM sera indispensable.

Dans un deuxième temps, nous aurions besoin d’une machine avec un accès à une base de données Oracle ainsi que qu’un accès libre a Internet (sans proxy bloquant les sites tel qu’Allocine).

# Les contraintes

L’une des premières contraintes que nous avons est l’utilisation d’Oracle v10.2.

D’autre part l’extraction des données se fera par des primitives XSLT mais la structure de données sera en HTML ce qui nous forcera à analyser syntaxiquement ces informations afin d’avoir ce qui se rapproche le plus d’une structure XML.

# L’univers de travail

Nous avons décidé que pour la récupération des données : nous utiliserons un univers java afin de profiter de sa souplesse d’utilisation de ressources tiers tel que le SGBD Oracle via JDBC, le logiciel JTIDY et les primitives XSLT.

Le stockage des données sera partiellement inséré sous format XML et stocké dans une base Oracle.

Pour finir la restitution des données se fera via une PUI en PHP/HTML.

# Répartition des tâches

Nous avons décidé de séparer le projet en 3 parties au sein du groupe.

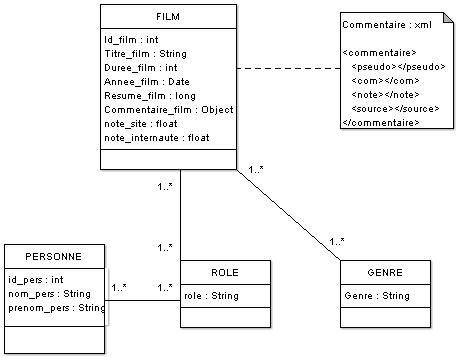
* M. Leonhard Hermle prendra à sa charge la PUI en PHP/HTML
* M. Cellier Julien s’occupera du stockage en base de données et XML avec interfaçage pour le PHP et le Java.
* M. Mourgues Xavier traitera la récupération des données en Java.

# Java

# Base de données

Au début de notre projet s’est posé le problème du choix du type de base de données que nous allions utiliser. En effet, il nous est mis à disposition deux bases de données Oracle XMLP1 et XMLP2. Il fallait donc choisir entre une base de données relationnelle qui prévoit l’utilisation de SQL2 ou une base de données objet utilisant du SQL3.

Pour une utilisation de ce genre il s’est révélé plus simple d’utiliser une base de données relationnelle.



# Restitution des données

La restitution s’effectue principalement sous le langage de programmation PHP. Bien sûr on retrouve aussi Du HTML et du CSS permettant la mise en forme des pages de l’interface.

L’interface doit être ergonomique et homogène afin de faciliter la navigation mais aussi sobre tout en respectant un choix de couleur épuré. Pour ce faire, Nous avons établis des règles permettant d’atteindre facilement l’interface recherchée. Nous avons aussi établis des aspects fonctionnels et techniques permettant une navigation améliorée sur l’interface.

POLICE DE CARACTERE : Nous avons choisis la police de caractères « verdana » sans serif où les empattements des caractères sont inexistants et donnent une impression de modernité. Le nombre de police est limité à une seule car trop de diversité donne un aspect lourd à l’interface qui devient désagréable à regarder. Cela porterait atteinte au confort de navigation de l’utilisateur. Aussi l’utilisation d’un nombre de polices de caractères trop élevé sera limitée et nous n’utilisons pas de types de police gras afin de remédier à cet effet.

COULEURS : Une interface optimale ne contient pas plus de trois couleurs. Ainsi, nous nous basons sur deux couleurs principales qui sont le Orange et le Kaki.

LIENS : Afin d’optimiser la souplesse de l’interface nous utilisons au maximum des liens notamment dans la barre de menu qui est le moyen le plus efficace de recherche sur les interfaces.

TAILLE DES ELEMENTS : Les pages seront assez courtes et le nombre d’éléments affichés sont limités. Cela nécessite le développement d’un système de pagination en PHP permettant de répartir les films sur un nombre de pages nécessaires. Aussi, toute liste de film n’affiche que les informations capitales, utile pour sa recherche. En cliquant sur ce dernier, la fiche entière du film sera affichée, permettant ainsi de voir tous les détails relatifs au film, tels que nom d’auteur, année de production, note attribuée, commentaires, etc.…

### NAVIGATION

Page principale (affiche tous les films par défaut).

Menu

Ajouter un film (permet la recherche d’un film grâce à un formulaire via l’application java)

Page recherche simple (fonction Ajax qui fait une recherche approximative d’un film par nom)

Page genres (affichage des genres disponibles dans la base de données Oracle)

Page films (propose un tri par alphabet)

Affichage des films disponibles sur les sites. Permet de choisir le bon résultat.

Page liste film (liste les films disponibles selon les options sélectionnées. C’est un affichage simple)

Affichage détaillé d’un film

MENU : Le menu comporte les principaux liens nécessaires à une recherche rapide de données. En effet, ici on retrouve des liens vers différentes pages tels que la recherche par alphabet ou par genres. Aussi on y retrouve le formulaire de recherche simplifié.

Ici nous avons à faire à un menu de type classique tel qu’on les retrouves souvent. L’avantage est la pertinence et l’efficacité de recherche car l’utilisation d’un menu est universelle pour toute application. L’utilisateur ne sera donc pas perturbé et trouvera rapidement les données de film recherchés.

CHAMP DE RECHERCHE : Il permet une recherche rapide d’un fim donné sous condition de connaitre un minimum d’information sur ce dernier (nom…). C’est une recherche approximative et plusieurs résultats peuvent être trouvés et affichés.

Le formulaire sera accompagné d’un script d’**auto-complétion** en ajax. Démuni d’un bouton de validation de la recherche, pour chaque caractère entré dans le champs, les résultats affichés seront réactualisés. Cette méthode apporte un confort conséquent à l’utilisateur mais peut cependant ralentir le système étant donné que la requête est ré-executée pour chaque nouveau caractère ajouté. Dans notre cas, les requêtes utilisées pour la recherche de données dans Oracle sont simples, n’utilisant que peu de jointure empéchant un ralentissement trop important de la recherche. Aussi, le nombre de film ne dépassant pas les quelques milliers, le parcours de la base Oracle se fera assez rapidement.

PAGINATION :

L’interface de restitution des données doit afficher un nombre de données assez important en l’occurrence d’une liste de films assez nombreux. En effet, ils existent trop de films pour pouvoir les afficher dans une seule page. De ce fait, nous utilisons la solution de pagination pour les films et commentaires dès que le nombre d’éléments donnés est d’ordre trop important. Cette solution consiste en la répartition du nombre de film sur un nombre de pages donnés.

Exemple : (nombre total d’élément à afficher / nombre de films par pages)

=> Résultat arrondi à l’unité supérieure

En php :

$elements\_totaux = 9 ;

$nombre\_par\_page = 2 ;

//ceil() permet d’arrondir à l’unité supérieur ;

//ne pas utiliser round() ;

$nb\_pages = ceil($elements\_totaux/$nombre\_par\_page) ;

//affichage des pages

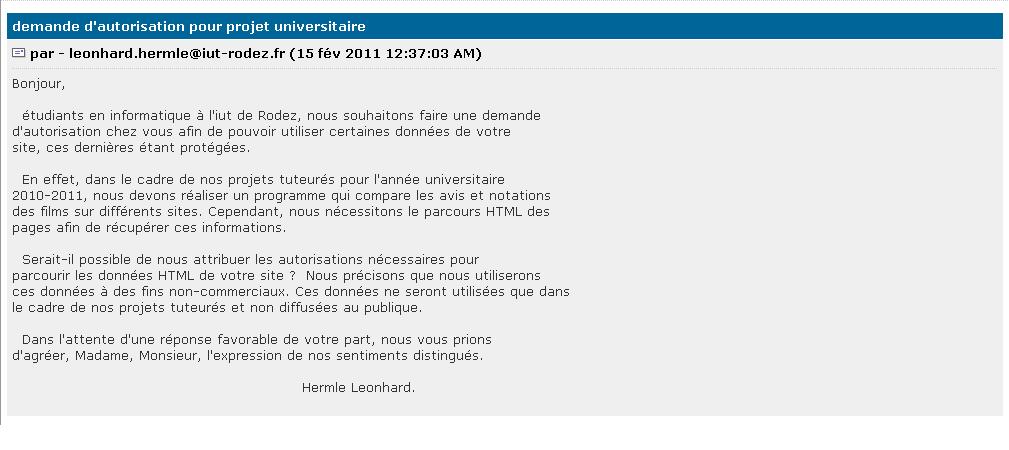
if ($nbpages > 1 ){ for ($i=1; $i<=$nb\_pages; $i++){ echo $i ;

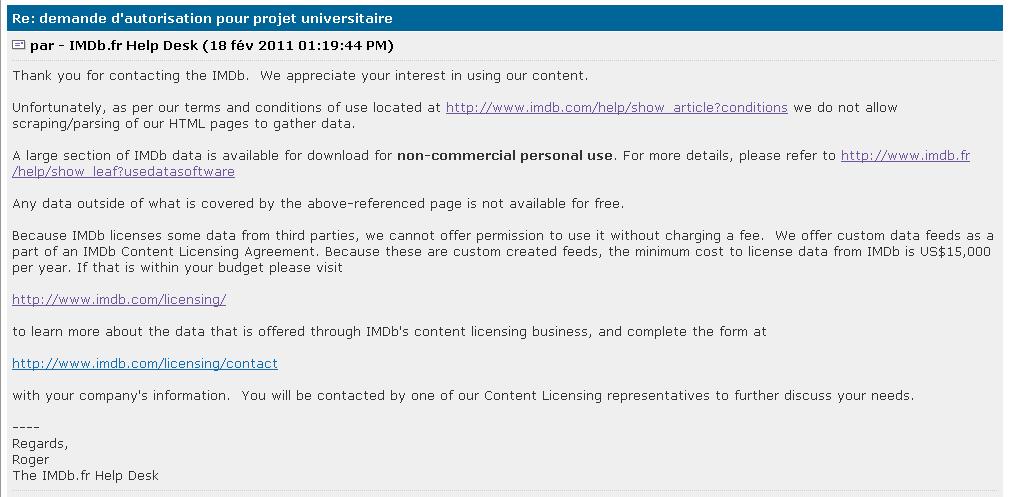
} }

# Problèmes

Base de données :

**IMDB** : Le site web IMDB refuse strictement le parcours des pages par un robot grâce à la protection par proxy. Cela pose problème pour notre projet car ce site contient l’une des plus grandes bases de données de films, très complet, fiable mais aussi détenteur de nombreux commentaires et critiques de film qui nous auraient été très utiles. Par conséquent, nous avons décidé de prendre contact avec cette société en expliquant les faits, afin de demander une autorisation de parcours des données. Malheureusement, nous avons reçus une réponse négative. En effet, la charte stipule l’interdiction de parsing / scraping de données. Cependant, une sélection de données peut être téléchargée gratuitement où encore une licence permettant l’accès à la totalité des données peut être acheté. Dans les deux cas, cette solution va à l’encontre de notre projet qui consiste à parcourir des pages spécifiques dans le but d’en extraire les données. Voici les messages envoyés à IMDB et leur réponse vis-à-vis de ce dernier :

Demande d’autorisation d’exploitation des données auprès de IMDB

Réponse de IMDB suite à notre demande

**Souci de délais :** Tous les objectifs n’ont pas pu être atteint.