AVIGNON UNIVERSITÉ

Rapport TP Genie Logiciel : SCRUM

Bilal Latrach
Yelhan Frendi
Hani Agguini
Nabil Bedjih
Omar El Baraka

18 décembre 2022

Licence Informatique
Parcours : Génie Logiciel
UE UE : Genie Logiciel

Responsables
Juan-Manuel Torres

UFR
SCIENCES
TECHNOLOGIES
SANTÉ

AVIGNON UNIVERSITÉ
CERI

CENTRE
D'**ENSEIGNEMENT**
ET DE **RECHERCHE**
EN **INFORMATIQUE**
ceri.univ-avignon.fr

Sommaire

1 Titre

Sprint 5 : Parseur d'articles scientifiques en format texte / XML

2 Abstract

Les chercheurs du Laboratoire Informatique d'Avignon (LIA) doivent lire des articles scientifiques publiés partout dans le monde. Et pour cela ils voudraient à avoir un système qui les présente un aperçu de l'article.
Les articles scientifiques qui sont en format PDF sont pas facile à analyser par les systèmes de Traitement Automatique de Langues (TAL), c'est pour cela qu'on doit les transformer en format txt pour pouvoir mieux les analyser.
Le logiciel que nous avons conçu , est un logiciel que nous avons realiser en language C, et qui consiste a extraire les sections de l'article PDF que nous avons deja transformer en fichier txt en utilisant la methode pdftotext, dans une version TXT ou XML, le choix revient à l'utilisateur.

3 Méthode

Dans cette section nous allons expliquer les différentes méthodes de notre logiciel.

3.1 Méthode 1 : listerPdf()

permet de convertir tous les fichiers pdf en txt en utilisant la methode pdftotext et les mettre dans un dossier "resultat".

3.2 Méthode 2 : listerPdfWithOptionLayout()

permet de convertir tous les fichiers pdf en txt en utilisant la methode pdftotext avec l'option -layout et les mettre dans un dossier "resultat". cette option permet de garder la structure du fichier

3.3 Méthode 3 : listerPdfWithOptionRaw()

permet de convertir tous les fichiers pdf en txt en utilisant la methode pdftotext avec l'option -raw et les mettre dans un dossier "resultat". cette option permet d'écrire la partie gauche de fichier avant la partier droite si le fichier est diviser sur deux côté

3.4 Méthode 4 : trouvertitre(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche le titre.
Cette fonction nous permet d'extraire le titre du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les conditions :
- prend la première ligne du fichier qui ne contient pas de caractères spéciaux et s'arrête lorsqu'une ligne contenant des caractères spéciaux est trouvée.

3.5 Méthode 5 : trouverauteurs(char * nomFichier)

En paramètres elle prend : nom du fichier dont on cherche les auteurs.
Cette fonction nous permet d'extraire les auteurs du fichier donner en paramètres on s'appuyant sur quelques condisions.

Les conditions :
on prend les ligne qui se trouve après le titre et qui contient les les caractères ',' '/' ou 'and' et il s'arrête lorsque il trouve le mot 'department', 'from' ou 'abstract'.

3.6 Méthode 6 : trouverabstract(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche l'abstract.
Cette fonction nous permet d'extraire l'abstract du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
-Trouver le mot abstract dans le fichier, puis commencer a prendre toutes les lignes qui suivent ce mot et s'arrête une fois le mot intoduction trouver

3.7 Méthode 7 : trouverIntroduction(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche l'introduction.
Cette fonction nous permet d'extraire l'introduction du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
-Trouver le mot introduction dans le fichier, puis commencer a prendre toutes les lignes qui suivent ce mot et s'arreter une fois qu'on trouve 2 ou II (càd début du corps).

3.8 Méthode 8 : trouverCorps(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche le corps.
Cette fonction nous permet d'extraire le corps du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
-Trouver le mot 2 ou II , puis commencer a prendre toutes les lignes qui suivent ces mots et s'arreter une fois qu'on trouve les mot discussion ou conclusion.

3.9 Méthode 9 : trouverConclusion(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche la conclusion.
Cette fonction nous permet d'extraire la conclusion du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
D'abord, la fonction cherche le dernier mot 'conclusion' qui se trouve dans une petite ligne "strlen(ligne)<55" (parce que le text peux contenir beaucoup de mots conclusion même il sont pas des vrais conclusion de notre pdf)
après, à partire cette ligne on prend tous les ligne jusqu' on trouve le mot "Aknowledg" ou "reference".
il retourne "il n y a pas" si la conclusion n'existe pas dans le fichier.

3.10 Méthode 10 : trouverDiscussion(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche la discussion.
Cette fonction nous permet d'extraire la discussion du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
D'abord, la fonction cherche le dernier mot 'discussion' qui se trouve dans une petite ligne "strlen(ligne)<55" (parce que le text peux contenir beaucoup de mots discussion même il sont pas des vrais discussion de notre pdf).

après, à partire cette ligne on prend tous les ligne jusqu' on trouve le mot "Aknowledg", "reference" ou "conclusion".
il retourne "il n y a pas" si la discussion n'existe pas dans le fichier.

3.11 Méthode 11 : trouverBiblio(char * fichier)

En paramètres elle prend : nom du fichier dont on cherche la bibliographie.
Cette fonction nous permet d'extraire la bibliographie du fichier donner en paramètres on s'appuyant sur quelques condisions.
Les Conditions :
-Trouver le mot references, puis commencer a prendre toutes les lignes qui suivent ce mot et s'arreter a la fin du fichier .

elle prend : nom du fichier dont on cherche les emails des auteurs. et on retourne les ligne qui contient des '@' et qui se trouve entre le debut de fichier et l'abstract

3.12 Méthode 13 : execute(int x)

En paramètres elle prend : un int qui peut etre 1 pour txt ou 2 pour xml
Dans cette fonction on a commencer par cree un dossier resume-fichier dans le quel on met les fichier txt ou xml qui contiennent les sections qu'on a extrait des pdf.
Puis on a utiliser utiliser la boucle while qui parcourt tous les fichiers et dans laquelle on appelle toutes les fonctions précédentes qui seront exécutées sur fichier après fichier.

3.13 Méthode 14 : main()

Dans la fonction main on donne le choix a l'utilisateur de choisir entre xml ou txt.

4 Résultats

Après évaluation de la qualité du système qui est réalisée en fonction de la précision obtenue sur un corpus de référence.
La précision sera calculée comme suit : Précision = Sections-correctes-trouvées-par-le-système / Sections-trouvées-par-le-système. Voila les résultats sur notre logiciel :

	Nom	Titre	Auteur	Abstract	Introduction	Corps	Conclusion	Discussion	bibliographie
A Benders Decomposition Approach toCorrelation Clustering									
A memetic algorithm for community detectionin signed networks	trouvée			trouvée	trouvée		trouvée	trouvée	trouvée
An Improved Branch and Cut Code for the Maximum Balanced Subgraph of a Signed Graph	trouvée	trouvée	trouvée	trouvée	trouvée	trouvée		trouvée	trouvée
Cabrera RESUMES 2019	trouvée	trouvée	trouvée		trouvée	trouvée		trouvée	trouvée
Conversational Networks for Automatic Online Moderation	trouvée		trouvée	trouvée	trouvée		trouvée	trouvée	trouvée
Dynamical Models Explaining Socia Balance and Evolution of Cooperation	trouvée	trouvée	trouvée			trouvée		trouvée	
Exact Clustering via Integer Programming and Maximum Satisfiability	trouvée			trouvée	trouvée	trouvée	trouvée	trouvée	
LDA resume	trouvée	trouvée	trouvée	trouvée	trouvée		trouvée	trouvée	trouvée
Partitioning large signed two mode networks : Problems and prospects	trouvée	trouvée			trouvée	trouvée			trouvée
Polibits 42 02	trouvée	trouvée		trouvée	trouvée	trouvée	trouvée	trouvée	trouvée

Alors on voit que notre programme a une précision de :
90% pour les noms
60% pour les titres
50% pour les auteurs
60% pour les abstracts
80% pour les introductions
60% pour les corps
50% pour les conclusions
80% pour les discussions
70% pour les bibliografiques

Alors la précision totale de notre programme est 66.67%

5 Conclusion

Tout au long du semestre, notre objectif principal était de concevoir et de créer un logiciel informatique fonctionnel et opérationnel.
Nous avons pu travailler en groupe au quotidien pendant toute la période de la réalisation du projet , le scrum master a su générer une grande force et une source de motivation, ce qui nous a donné l'ambition de rester motivés. l'ambiance était parfaite , et dans des intervalles réguliers l'équipe réfléchissait aux solutions plus efficaces, pour résoudre les problèmes rencontrés, en changeant son comportement en conséquence.
Pour Réaliser un logiciel de qualité on s'est basé sur ce qu'on a vu durant le cours, plus précisément la méthode agile, après avoir compris sa flexibilité et ses avantages majeurs.
Le "product backlog", nous a permit de partager rapidement les taches entre les membres du groupe, et de passer trop vite à la réalisation.
Ensuite,grace aux mêlées des sprint, on a pu trouver rapidement les problèmes de notre

logiciel et réagir plus rapidement pour résoudre cela de façons plus efficace.
En fin, la revue de sprint, nous a permit de savoir ceux qu'il fallait améliorer lors du prochain Sprint, de façon à être toujours agiles.
On a pris on considération Les changements fréquentes du client (qui était dans notre cas les sujets des différents Sprints) et les imprévus, ont été pris en compte et l'équipe du projet a pu réagir rapidement.
Pour conclure, notre logiciel a besoin d'une amélioration, pour le rendre plus opérationnel et efficace avec le plus de PDF, en tenant compte des exceptions que nous pouvons rencontrer dans ces fichiers .