POLYTECH
NICE-SOPHIA

Université
Nice
Sophia Antipolis

Membre de UNIVERSITÉ CÔTE D'AZUR

Fahim BELLILI - Tarik HAMLAT
Master 2 IFI Web

Rapport du projet web sémantique

Recherche sémantique et recommandation dans les documents

2018 - 2019

Sommaire

1. Introduction

Au fil des ans, Internet est devenue l'un des principaux réseaux de communication à travers le monde et est de plus en plus utilisé comme source principale d'information et de communication. Il est donc essentiel que l'expérience utilisateur soit approfondie et améliorée. Ce projet consiste en la création d'une application Web exploitant le Web sémantique. Pour y parvenir, il était nécessaire d'acquérir et d'approfondir nos connaissances dans le web sémantique, pour mener à bien ce projet.

Nous avons utilisé l'outil Protégé et le langage de recherche SPARQL pour rechercher efficacement des articles extraits du journal The New York Times, ainsi que de fournir à l'utilisateur des informations intéressantes en fonction de l'historique des recherches.

2. Architecture du système

Cela étant une conception relativement complexe avec plusieurs composants, deux langages de programmation et plusieurs outils ont été utilisés. Les langages utilisés sont Python et Java, et les outils utilisés sont : Protégé, Apache Jena et Apache Tomcat. Ces deux langages remplissent des fonctions différentes, Python pour le processus d'extraction et de traitement de l'information et Java pour développer la partie Web du projet. Au début du projet, il était nécessaire de définir quel type d'informations et de données serait utilisé et comment cela serait arrangé. À cette fin, nous avons pu définir une ontologie à travers l'outil Protégé, qui structure et crée une connexion entre les données comme s'il s'agissait d'un réseau.

Après avoir créé l'ontologie, il était nécessaire de la remplir avec des données. Pour cela, nous avons créé un petit programme en Python qui récupère des données à partir de **L'API Top Stories du New York Times (https://developer.nytimes.com/apis)** et les enregistre dans un fichier JSON. Une fois l'extraction des données terminée, le programme va peupler l'ontologie avec les données obtenues, afin d'avoir une ontologie avec toutes les données organisées et liées. Une fois cette étape terminée, nous procédons ensuite à la création de l'interface Web. À ce stade, il y a deux parties principales, l'interface utilisateur et le l'ontologie.

Pour manipuler l'ontologie, nous avons utilisé l'outil Apache Jena pour faire correspondre les données avec les classes et les propriétés. Grâce à cet outil et à l'aide du langage de recherche pour RDF, SPARQL, il est possible de faire des recherches de manière simple et efficace. Enfin, pour développer l'interface graphique, nous utilisons JSP (langage JAVA) qui nous permet d'interconnecter les requêtes selon les demandes de notre ontologie. Nous avons aussi utilisé le serveur web Apache Tomcat pour le déploiement.

3. Description de l'ontologie

Une ontologie est un modèle de données qui représente un ensemble de concepts au sein d'un domaine et leurs relations. En règle générale, ils décrivent 4 aspects fondamentaux:

- Individus - objets de base, tels que des personnes, des animaux, objets concrets, etc.
- Classes - ensembles, collections ou types d'objets, par exemple un groupe de personnes.

- Attributs - propriétés, caractéristiques que les objets peuvent avoir ou partager.
- Relations - la façon dont les objets se rapportent les uns aux autres.

Avant de créer l'ontologie, il était nécessaire de déterminer quelles données seraient pertinentes et comment elles seraient mises en relation. Pour cela, nous avons d'abord vérifié quelles informations l'API utilisée fournissait et quelles parties nous seraient utiles. Après cela, il était alors nécessaire de réfléchir à la manière dont elles seraient liées.

Après avoir pensé et élaboré la structure de l'ontologie, nous sommes passés à sa création. Pour cela, nous avons utilisé l'outil Protégé, qui nous aide à créer et exporter l'ontologie. Dans cet outil, une classe a été créée ainsi que ses propriétés et relations.

3.1. Classes

Comme la seule information que nous devions garder sur l'ontologie était celle des articles du New York Times, il suffisait de créer une seule classe :

- Articles - classe qui représente chaque article et contient toutes ses informations.

3.2. Propriétés

Les propriétés de la classe ont été pensées en tenant compte des informations que nous souhaitions voir affichées et mises à la disposition de l'utilisateur. Ainsi, comme indiqué ci-dessus, nous avons examiné les informations fournies par l'API et choisi d'enregistrer uniquement dans l'ontologie ce qui serait utile.

Les enregistrements contiennent deux attributs qui leur sont associés, le domaine qui définit les classes qui contiennent la propriété et la plage qui fait référence au type de la propriété, par exemple String, Integer, Date, etc.

Les propriétés créées ont toutes le même domaine, la classe Articles, et leurs types est décrit ci-dessous à côté des propriétés:

- Section [String] - propriété de la classe qui contient la section.
- Subsection [String] - propriété de la classe qui contient la sous-section.
- Title [String] - propriété de la classe qui contient le titre de l'article.
- Abstract [String] - propriété de la classe qui enregistre le résumé sur de l'article.
- Author [String] - propriété de la classe qui contient l'auteur de l'article.
- URL [String] - propriété de la classe qui stocke l'URL de l'article. Original du site Web du New York Times.
- Published date [DateTimeStamp] - propriété de la classe qui stocke la date de publication de l'article.
- Subject description [String] - propriété de la classe qui contient les balises sur la description du sujet de l'article.
- Organization [String] - propriété de la classe qui contient les balises sur les organisations mentionnées dans l'article.
- Person [String] - propriété de la classe qui contient les balises sur les personnes qui sont mentionnées dans l'article.

- Location [String] - propriété de la classe qui contient les balises sur le mentionné dans l'article.
- Image [String] - propriété de la classe contenant l'URL de l'image. De l'article.

4. Modules

Comme il s'agit d'un projet Web sémantique, le plus important étaient la recherche sémantique. Ce mécanisme offre à l'utilisateur une meilleure expérience en lui permettant de rechercher les sujets qui l'intéressent le plus.

Dans ce qui suit, nous expliquerons comment cette partie a été développée pour mieux la comprendre.

Recherche sémantique

La recherche sémantique est possible grâce au langage de recherche SPARQL. Ce dernier utilise des requêtes pour rechercher le fichier OWL qui contient l'ontologie remplie avec les données des articles. Un peu similaire à la recherche dans le langage SQL.

Plusieurs méthodes ont été créées pour effectuer une recherche, afin de fournir à l'utilisateur une meilleure expérience et davantage de moyens pour interagir avec le site. Les modes de recherche sont les suivants:

- Normal.
- Section.
- Sous-section.
- Auteur.
- Les étiquettes.

Pour effectuer la recherche de ces différentes méthodes, une fonction globale a été créée, qui a reçu plusieurs paramètres, puis utilisée pour effectuer la recherche, en tenant compte de ce qui était prévu.

Dans le cas d'une recherche normale, si l'utilisateur écrit un mot à rechercher, la requête tente de rechercher à la fois dans le titre et dans le résumé des articles à la recherche de ceux dans lesquels le mot était présent. Ceci a été réalisé grâce à l'utilisation de l'union et de la regex.

Il est à noter que pour améliorer la recherche, le fait que les caractères soient en majuscule ou en minuscule a été ignoré, les résultats ont été obtenus de manière ordonnée en fonction de la date de publication et ils étaient toujours différents pour éviter une arithmétique répétitive dans les résultats.

5. Structure du programme

Comme mentionné précédemment, le programme a été développé avec deux langages, Python et Java. Python pour peupler l'ontologie et Java pour l'interface et la recherche.

5.1. Python

Dans cette partie du programme, il était nécessaire d'extraire les données via l'API du New York Times et de les enregistrer dans des fichiers JSON, de manière qu'elles puissent être utilisées pour l'ontologie créée avec Protégé.

Les fichiers nécessaires à l'exécution de cette partie du projet se trouvent dans le dossier "**semantic_web_project/Ontology**" et sont organisés comme suit:

- **populate.py** - programme Python qui extrait des données via l'API, les enregistre dans des fichiers JSON, puis les utilise pour remplir le fichier ontologie. Ces processus ont été exécutés avec l'aide des librairies JSON et RDFLIB pour Python. À la fin de l'exécution, un fichier contenant les données insérées dans l'ontologie est créé.
- **query.py** - Programme Python créé pour tester, entre autres, la recherche sur les ontologies.
- **populated.owl** - fichier généré par le programme Python qui contient l'ontologie remplie avec les données.
- Dossier "**Data**"
    - Dossier "**Parsed**" - dossier contenant les fichiers avec les informations obtenues à partir de l'API de manière organisée pour une meilleure lecture et compréhension des données obtenues.
    - Dossier "**Retrieved**"- dossier contenant les fichiers JSON obtenus via l'API.
    - **parser.py** - Programme Python qui lit les fichiers JSON et les transforme en fichiers txt avec seulement les données nécessaires de manière organisée.

5.2. Java

Nous avons utilisé l'IDE Intellij Idea car ça nous a facilité l'utilisation des bibliothèques externes nécessaires pour Apache Jena ainsi que pour Apache Tomcat.

Cette partie est divisée en deux dossiers principaux, le premier contenant les classes pour la recherche la seconde contient l'interface Web.

- Dossier "**src**"
    - **main/java/beans/SearchBean.java** - Classe contenant la fonction ontologie ainsi que la fonction qui obtient les recommandations.
    - **main/java/entités/Article.java** - classe qui définit un article.
    - **main/java/recommendation.txt** - fichier avec les "journaux" des sections visitées par l'utilisateur pour créer les recommandations.

- Dossier "**webapp**"
    - **index.jsp** - fichier jsp contenant l'interface Web du projet.
    - **css/style.css** - fichier CSS contenant tous les CSS nécessaires pour améliorer l'apparence de l'interface Web.
    - **js/style.css** fichier javascript contenant tous les éléments pour l'interface Web.

6. Conclusions et perspectives

De nos jours, le Web sémantique est présent dans un grand nombre de sites, même dans ceux auxquels on s'attend le moins. C'est une partie importante de l'interaction de l'utilisateur avec Internet : présence sur les sites de recherche, achats, actualités, etc.

Dans ce projet, nous avons acquis des connaissances sur le web sémantique et les différentes parties qui l'intègrent. Les connaissances nécessaires au développement d'un tel système ont également été acquises car l'application a été développée à partir de zéro.

En ce qui concerne la continuité, l'un de nos objectifs est d'approfondir la connaissance de ce type de systèmes car cette technologie est en constante croissance et évolution.

7. Bibliographie

- http://jena.apache.org/
- http://www.w3.org/
- http://rdflib.readthedocs.org/
- http://www.tutorialspoint.com
- http://www.w3schools.com/
- http://stackoverflow.com/
- http://docs.python.org
- http://docs.oracle.com
- http://developer.nytimes.com/
- http://getskeleton.com/
- http://wikipedia.org