Web sémantique
Timeline plugin
...

Présenté par:
Rémi Pourtier
Jonathan Pujol

Sommaire

- Introduction
- Système interactif de notre projet
- Structure de notre RDF et RDFs
- Récupération des données: Dbpedia
- Récupération des données : Youtube
- Traitement des données
- Démonstration
- Conclusion

Introduction

- Contexte:
  - Le projet Timeline
  - Notre rôle dans ce projet

- Technologies utilisées
  - Au départ: JavaScript
  - Finalement: Java, serveur Java appelé SPRING, API Jena
  - Des requêtes faites sur Dbpedia et YouTube

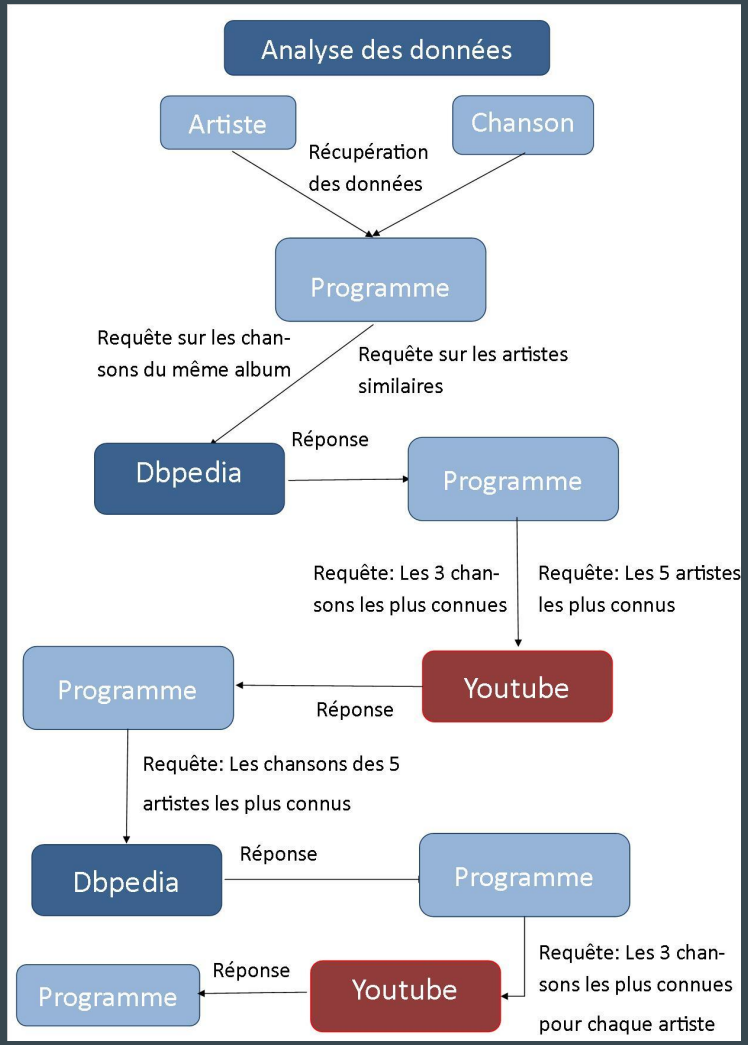
Système interactif de notre projet

Communication entre:

- Notre programme et DBepdia
- Notre programme et Youtube

Plusieurs étapes

- Récupérer les données
- Préparer les données à la construction de notre RDF



Structure de notre RDFs

- On utilise Jena
  - Création d'ontologie
  - Importation d'ontologie

- Langage OWL
  - Création des classes
  - Création des propriétés
  - Règle d'inférence

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:j.0="http://notreOnthologie#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
  <owl:Ontology rdf:about="http://notreOnthologie#"/>
  <owl:Class rdf:about="http://notreOnthologie#Song"/>
  <owl:Class rdf:about="http://notreOnthologie#Profil"/>
  <owl:Class rdf:about="http://notreOnthologie#Artist"/>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#haslistened">
    <rdfs:range rdf:resource="http://notreOnthologie#Song"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Profil"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#hasPopularSong">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:about="http://notreOnthologie#hasArtist"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="http://notreOnthologie#Song"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Artist"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#hasArtist">
    <rdfs:range rdf:resource="http://notreOnthologie#Artist"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#hasCloseSong">
    <rdfs:range rdf:resource="http://notreOnthologie#Song"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#hasCloseArtist">
    <rdfs:range rdf:resource="http://notreOnthologie#Artist"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://notreOnthologie#hasAlbumSong">
    <rdfs:range rdf:resource="http://notreOnthologie#Song"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:ObjectProperty>
  <owl:DatatypeProperty rdf:about="http://notreOnthologie#hasYoutubeView">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#long"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="http://notreOnthologie#hasBeenSeen">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="http://notreOnthologie#Song"/>
  </owl:DatatypeProperty>
</rdf:RDF>
```

Exemple de RDF obtenu

- Résultat sur Yesterday des Beatles

- On conserve les résultats précédents

```
<j.0:Profil rdf:about="http://notreOnthologie#4300">
  <j.0:haslistened>
    <j.0:Song rdf:about="http://notreOnthologie#Yesterday">
      <j.0:hasCloseArtist>
        <j.0:Artist rdf:about="http://notreOnthologie#Red Hot Chili Peppers">
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Otherside"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Dani California"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Snow "/>
          </j.0:hasPopularSong>
        </j.0:Artist>
      </j.0:hasCloseArtist>
      <j.0:hasCloseArtist>
        <j.0:Artist rdf:about="http://notreOnthologie#Nirvana ">
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Heart-Shaped Box"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#In Bloom"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Smells Like Teen Spirit"/>
          </j.0:hasPopularSong>
        </j.0:Artist>
      </j.0:hasCloseArtist>
      <j.0:hasArtist>
        <j.0:Artist rdf:about="http://notreOnthologie#The_Beatles"/>
      </j.0:hasArtist>
      <j.0:hasCloseArtist>
        <j.0:Artist rdf:about="http://notreOnthologie#System of a Down">
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Toxicity"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Hypnotize"/>
          </j.0:hasPopularSong>
          <j.0:hasPopularSong>
            <j.0:Song rdf:about="http://notreOnthologie#Sugar"/>
          </j.0:hasPopularSong>
        </j.0:Artist>
      </j.0:hasCloseArtist>
```

Récupération des données: Dbpedia

Requête sur les genres de musique des artistes

```
SELECT DISTINCT ?list WHERE{ {SELECT DISTINCT ?value WHERE {

     <http://fr.dbpedia.org/resource/NameOfArtist> dbpedia-owl:genre ?
value

}}
?value dbpedia-owl:wikiPageWikiLink ?list

. filter( exists {?list dbpedia-owl:bandMember ?members})
```

Problème rencontré lors de l'ajout de la date dans la requête

Récupération des données: Dbpedia

Requête sur les chansons d'un même album

```
SELECT DISTINCT ?list ?value
    WHERE {
        {SELECT DISTINCT ?value
            WHERE {
                <http://fr.dbpedia.org/resource/Yesterday> dbpedia-owl:album ?value
            }
        }
        ?value dbpedia-owl:wikiPageWikiLink ?list
        .filter(exists{?list dbpedia-owl:recordDate ?members})
        .filter(?list!=?value)
    }
```

Requête sur les chansons d'un même groupe

```
SELECT DISTINCT ?list WHERE{
    <http://fr.dbpedia.org/resource/Aerosmith> dbpedia-owl:wikiPageWikiLink ?list
    .filter(exists{?list dbpedia-owl:artist <http://fr.dbpedia.org/resource/Aerosmith>})
}
```

Récupération des données: Youtube

- Deux requêtes distinctes

- 3 chansons les plus vues d'une liste

```
La chanson From Me to You a 7493973 vues
La chanson You Can't Do That a 45631378 vues
La chanson Help!  a 10817113 vues
```

- 5 artistes les plus vues

```
L'artiste System of a Down a 655452486 vues
L'artiste Nirvana  a 701901477 vues
L'artiste Bon Jovi a 738400390 vues
L'artiste Love  a 1299551571 vues
L'artiste Arctic Monkeys a 452448355 vues
```

Traitement des données

- Classe CleanRDF
  - Extraction des données depuis le fichier data.rdf
  - Nettoyage du fichier
  - Nettoyage des données

- Classe Application
  - Lancement du serveur
  - Appels des fonctions pour effectuer des requêtes et pour manager le contenu de data.rdf
  - Création des fichiers d'ontologies

Démonstration

Conclusion

- Problèmes rencontrés:
  - Choix des technologies
  - Adaptation des requêtes SPARQL
  - Extraction, nettoyage, affinage des données
  - Utilisation de Jena notamment l'importation d'une ontologie existante
  - Avec le language RDF: Comment créer privatiser les données d'un noeud ?
  - Requêtes sur Youtube limitées
- Amélioration à apporter
  - Améliorer le traitement des données
  - Développement d'une interface
  - Ajout d'un compteur de vues sur notre RDF
- Les résultats:
  - Une chanson écoutée -> 18 suggestions de chansons
  - Création d'une ontologie