UNIVERSIDADE DE AVEIRO | Departamento de Eletrónica, Telecomunicações e Informática

Web Semântica – 2º Semestre 2020-21

Relatório Trabalho Prático 2

Curso:

Mestrado em Engenharia Informática

Data:

24 de Junho de 2021

Docentes:

Professor Hélder Zagalo

Discentes:
João Ferreira, 80041
João Magalhães, 79923

1. Introdução

Este projeto foi desenvolvido no âmbito da Unidade Curricular Web Semântica e tem como objetivo a melhoria da aplicação desenvolvida no primeiro projeto da mesma UC, adicionando uma ontologia, inferências, fontes externas de conhecimento (dbpedia e wikidata) e publicação de dados.

2. Objetivos

Para este projeto estabelecemos os seguintes objetivos:

- Criação de uma ontologia;
- Criação de inferências baseadas na ontologia;
- Melhoria de base de conhecimento através de fontes externas;
- Publicação dos dados;

3. Ontologia

Para o desenvolvimento deste projeto foi-nos pedida a criação de uma ontologia que descrevesse o domínio dos dados. Esta ontologia deve ter um vocabulário preciso, deverá ser orientada às entidades (e não às propriedades) e definir regras de inferência. De modo a definirmos a ontologia dos dados, recorremos a RDFS e OWL, definido classes e propriedades sobre o nosso conjunto de dados

3.1 Classes e subclasses do domínio

Uma ontologia deve ser orientada a entidades e, por este motivo, criámos as seguintes classes:

- Person;
- Type;
- Country;
- Listed_in;

Também criamos as seguintes subclasses:

- Actor (subclasse de Person, se esta fizer parte do elenco) ;
- Director (subclasse de Person, se esta for o realizador de um movie/tv show);
- Celebrity (subclasse de Person, se esta fizer parte do elenco) ;
- Movie (subclasse de Type, se for um filme) ;
- TV Show (subclasse de Type, se for um tv show) ;

3.2 Inferências entre classes e subclasses

Na tabela seguinte podemos observar como foram inferidas as classes e subclasses. Devido à ineficiência do protégé realizamos as inferências manualmente, sendo estas descritas no ficheiro “inferencias.txt”.

Classe/Subclasse	Inferência
Person	Todas as pessoas que sejam actores, realizadores ou ambos.
Type	Todos os Movies e TV Shows presentes.
Country	Todos os objetos associados ao predicado recorded.
Listed In	Todos os objetos associados ao predicado listed_in.
Actor	Todas as pessoas que tenham participado num Movie ou TV Show.
Director	Todas as pessoas que tenham sido realizadores de um Movie ou TV Show.
Celebrity	Todas as pessoas (actor ou director) que participaram num Movie ou TV Show.
Movie	Todos os objetos que tenham o predicado starring, directed, recorded e listed_in.
TV Show	Todos os objetos que tenham o predicado starring_tvshow, directed_tvshow, recorded_tvshow e listed_in_tvshow.

3.3 Novas propriedades declaradas

Declaramos também as seguintes propriedades inversas:

- starredin - propriedade inversa de starring (movies);
- starredin tvshow - propriedade inversa de starring_tvswhow (tv show);
- directed - propriedade inversa de directed_by;
- workers - propriedade inversa de worked_in (movies);
- workers tvshow - propriedade inversa de worked_in_tvshow (tv show);

```
# create inverse property of worked_in_tv_show
pred:workers_tv_show rdf:type owl:ObjectProperty;
    owl:inverseOf pred:worked_in_tv_show;
    rdfs:comment "Person that worked on a tv show".
```

Fig. 1 - Propriedades inversas

Declaramos também as seguintes propriedades simétricas:

- similar to - Person que trabalhou no mesmo TV Show que outra Person;
- similar to TV Show - Person que trabalhou no mesmo TV Show que outra Person;
- is friend of - relação de amizade entre duas persons;

```
# Create symetric property
pred:similar_to_tvshow rdf:type owl:SymmetricProperty;
    rdfs:domain netflix_titles:TVShow;
    rdfs:range netflix_titles:TVShow;
    rdfs:comment "Person that worked on a TVShow".
```

Fig. 2 - Propriedades simétricas

3.4 Sub-Propriedades

Definimos as seguintes propriedades demonstradas na figura seguinte.

```
#Actor properties

pred:starred_in rdf:type owl:ObjectProperty;
    rdfs:subPropertyOf pred:worked_in;
    owl:inverseOf pred:starring;
    rdfs:comment "Inverse of the predicate starring".

pred:starred_in_tv_show rdf:type owl:ObjectProperty;
    rdfs:subPropertyOf pred:worked_in_tv_show;
    owl:inverseOf pred:starring_tvshow;
    rdfs:comment "Inverse of the predicate starring".


#Director properties

pred:directed rdf:type owl:ObjectProperty;
    rdfs:subPropertyOf pred:worked_in;
    owl:inverseOf pred:directed_by;
    rdfs:comment "Inverse of the predicate directed_by".

pred:directed_tv_show rdf:type owl:ObjectProperty;
    rdfs:subPropertyOf pred:worked_in_tv_show;
    owl:inverseOf pred:directed_by_tv_show;
    rdfs:comment "Inverse of the predicate directed_by".
```

Fig. 3 - Sub-propriedades

3.5 Grafismo da Ontologia

A ontologia desenvolvida pode ser visualizada na figura seguinte ou recorrendo à aplicação protegé, utilizando o ficheiro netflix_titles_ontology.n3 .

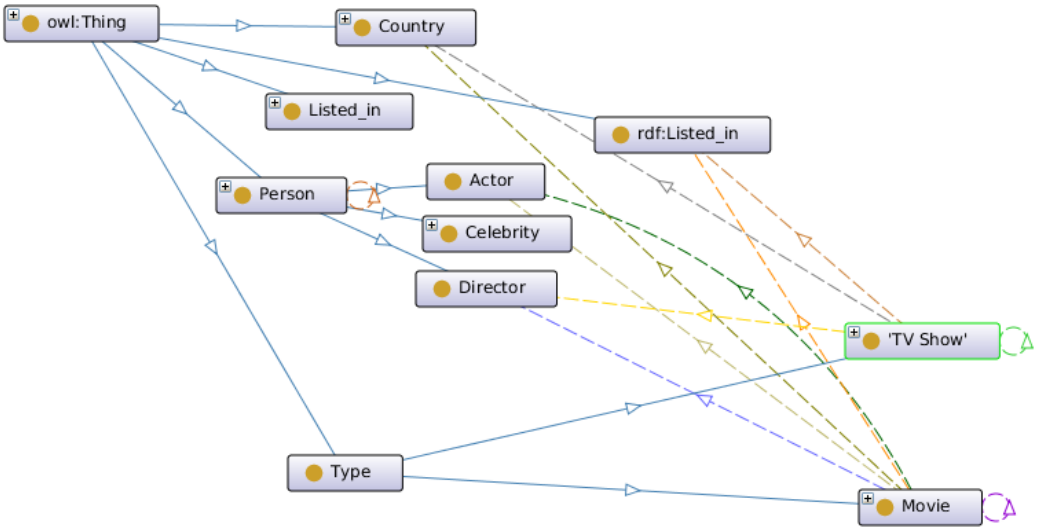

Fig. 4 - Ontologia

4. Wikidata

Outra das funcionalidades pedida era a melhoria da aplicação com bases de conhecimento externas, como a wikidata. O wikidata é uma base de conhecimento consideravelmente extensa e permite aumentar a base de conhecimento, fornecendo dados extra.

4.1 Obtenção dos dados

De modo a obtermos os dados da wikidata utilizamos a biblioteca SPARQLWrapper, permitindo-nos estabelecer a conexão com a wikidata. Na figura seguinte podemos observar as queries desenvolvidas (data de nascimento, descrição e prémios).

```
SELECT DISTINCT ?director ?birthDate
WHERE  {
       ?director wdt:P1559 ?actorLabel; wdt:P569 ?birthDate . FILTER(STRSTARTS(?actorLabel,
'Steven Spielberg')) .
}

Select ?o
where{
  wd:8877  schema:description ?o .
  filter(lang(?o) = "en")
}


SELECT ?awardsReceivedLabel
WHERE {
  wd:8877 wdt:P166 ?awardsReceived .

  SERVICE wikibase:label {bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en".}
}
```

Fig. 4-Queries Wikidata

Algumas queries, para além de demorarem consideravelmente, resultam, por vezes, em timeouts (retornando um array vazio). Destes timeout, resulta um erro que impede que a aplicação funcione corretamente.

5. DBPedia

Outra das funcionalidades pedida era a melhoria da aplicação com bases de conhecimento externas, como a DBPedia. A DBPedia, como a wikidata, é uma base de conhecimento consideravelmente extensa e permite aumentar a base de conhecimento, fornecendo dados extra.

5.1 Obtenção dos dados

De modo a obtermos os dados da DBPedia utilizamos a biblioteca SPARQLWrapper, permitindo-nos estabelecer a conexão com a DBPedia. Na figura seguinte podemos observar a querie desenvolvida (se um filme é baseado num livro).

```
select distinct *
{
    ?movie dbo:basedOn ?book .
    ?movie foaf:name 'Catch Me If You Can'@en .
    ?book a dbo:Book .
}
```

Fig. 5-Exemplo de query DBPedia para o filme ‘Catch Me If You Can’

6. Publicação de dados

Para a publicação dos dados utilizamos o formato RDFa e Micro Formatos.

6.1 Micro Formatos

Aplicamos o formato Micro Formatos nas páginas Movies e TV Show. Nestas páginas consideramos o tipo (Movie ou TV Show) e como propriedades o título, o realizador, o ano de publicação e as categorias.

JSON


```
{
    "items": [
        {
            "type": [
                "h-tvshow"
            ],
            "properties": {
                "tvshow-title": [
                    "3%"
                ],
                "tvshow-director": [
                    "unknown"
                ],
                "tvshow-release-year": [
                    "2020"
                ],
                "tvshow-category": [
                    "International TV Shows, TV Dramas, TV Sci-Fi & Fantasy"
                ]
            }
        },
        {
            "type": [
                "h-tvshow"
            ],
            "properties": {
```


Fig. 6 - Micro Formatos (TV Show)

JSON


```
{
    "items": [
        {
            "type": [
                "h-movie"
            ],
            "properties": {
                "movie-title": [
                    "7:19"
                ],
                "movie-director": [
                    "Jorge Michel Grau"
                ],
                "movie-release-year": [
                    "2016"
                ],
                "movie-category": [
                    "Dramas, International Movies"
                ]
            }
        },
        {
            "type": [
                "h-movie"
            ],
            "properties": {
```


Fig. 7 - Micro Formatos (Movies)

6.2 RDFa

A publicação via RDFa é mais completa, sendo possível exibir as relações entre as entidades e também todas as propriedades.

Aplicamos o formato RDFa nas páginas Movies, TV Show e Search. Nestas páginas consideramos o tipo (Movie ou TV Show) e como propriedades o título, o realizador, o ano de publicação, elenco, país, data de publicação na netflix, duração e as categorias.

Na figura seguinte vemos um exemplo (da página Movies).

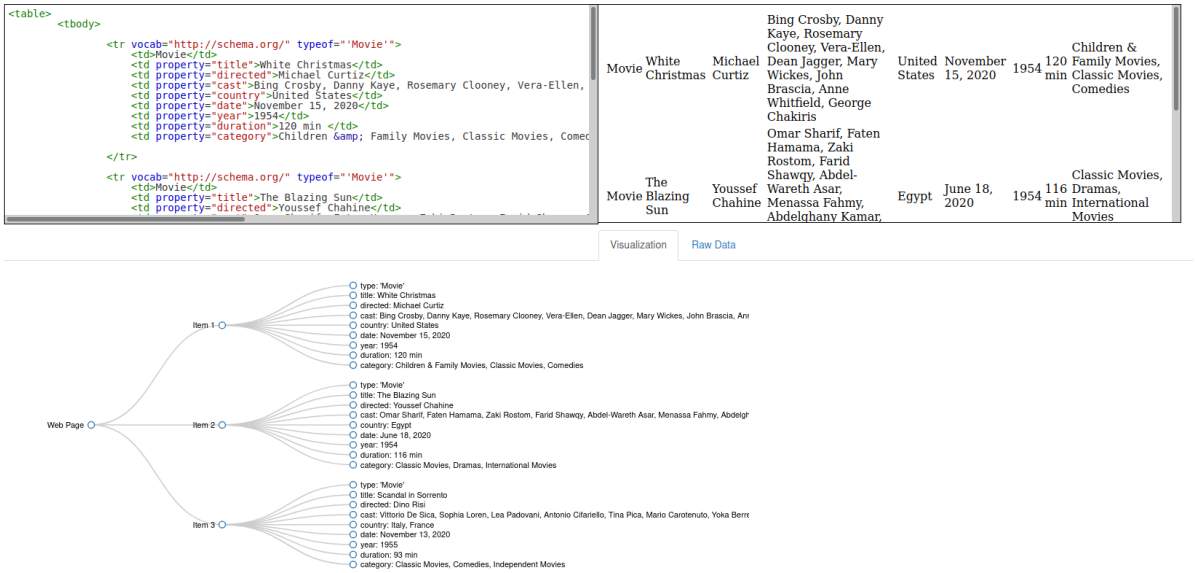

Fig. 8 - RDFa (Movies)

7. Conclusão

Com a criação da ontologia, concluímos que esta nos permitiu extrair mais informação em relação à obtida no primeiro trabalho, especialmente quando conjugada com as inferências.

As fontes externas mostraram-se bastante acessíveis, permitindo aumentar a qualidade dos dados e enriquecer a aplicação.

Finalmente, com a publicação de dados conseguir extrair muito mais informação da página do que apenas recorrendo a elementos HTML. Consideramos que a informação publicada através de RDFa é mais útil do que por Micro Formatos, de acordo com a nossa aplicação.

8. Configuração

Para executar a aplicação é necessário:

1. Executar o graphDb na porta 7200 e abrir a interface do graphDB no browser.
2. Criar um novo repositório com o ID “netflix”.
3. Importar o ficheiro netflix_titles.nt para o novo repositório.
4. Importar o ficheiro netflix_titles_ontology.n3 para o novo repositório.
5. Executar o projeto Django.
6. Aceder a http://localhost: 8000 .

É necessário instalar a biblioteca python SPARQLWrapper e s4api.

No directório vídeos é possível ver as funcionalidades da aplicação, incluindo a aplicação desenvolvida para o trabalho 1, a opção de editar dados, as inferências e acessos à wikidata e dbpedia.