Processamento e Representação de conhecimento

Trabalho Prático

Relatório de Desenvolvimento

Jaime Leite
(A80757)

3 de Julho de 2020

Resumo

O relatório apresentado, no âmbito da unidade curricular de *Processamento e Representação de Conhecimento*, tem como principal foco a implementação de uma aplicação para percorrer uma ontologia em *GraphDB*. Esta ontologia tem informação acerca de jogadores, equipas, torneios e partidas de ténis desde o início da *Era Open*, ou seja, desde 1968. A aplicação está desenvolvida em *JavaScript* e *VueJs* e percorre a informação da ontologia mencionada anteriormente.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

Nos dias de hoje, existe muita informação nos mais variados locais da *Web* que não está assente sobre uma estrutura suficientemente sódida e sobre a qual não são usadas as melhores ferramentas *Web* para a mostrar aos utilizadores. É necessário que se usem as ferramentas mais adequadas para que, não só os diferentes constituintes a representar se interliguem muito bem uns com os outros, como também as consultas à informação devem ser o mais eficientes possíveis. Tendo isto por base, o uso de ontologias para representar informação visa corresponder a estes requisitos. O uso de triplos para representar as diferentes entidades e respetivos atributos permite representar os dados de forma simples, sólida e sem ambiguidades. Por forma a estruturar ontologias e armazenar a informação, existem sistemas, como por exemplo, o *GraphDB*. Este repositório de dados permite guardar, não só a estrutura de várias ontologias, como para cada uma delas criar os mais variados indivíduos. Para além disto, disponibliza uma *API* para que sistemas externos interoperem com este sistema. Relativamente a ferramentas para mostrar a informação aos utilizadores, é relevante referir o *VueJs* que, inserindo-se no conjunto de frameworks reativas, permite estruturar os acessos à informação sem grandes problemas.

1.1 Descrição do problema

Foi proposto o desnvolvimento de uma aplicação *Web* que percorresse informação representada numa ontologia. Por forma a estruturar a ontologia, foi dada o máximo de liberdade, desde as ferramentas a utilizar, como também o local de onde esta informação se encontra.

1.2 Objetivos

Para ser possível implementar a solução que corresponda ao problema proposto, os objetivos a cumprir para este trabalho são os seguintes:

- Escolher um tema que a ontologia vai representar;
- Definir a ontologia;
- Procurar informação na *Web* que permita povoar a ontologia definida;
- Implementar uma interface *Web* que percorra a informação(entidades e respetivos atributos) que está contida na ontologia.

Capítulo 2

Desenvolvimento da solução

Neste trabalho, foi tido em conta o que foi feito nas aulas práticas da unidade curricular, tendo então sido estruturado o trabalho em três grandes partes: servidor de bases de dados(*GraphDB*); servidor de *API*(em *NodeJs*, apoiado por *Express*) e servidor de interface(usando a ferramenta *VueJs*). De seguida apresenta-se mais detalhadamente o trabalho desenvolvido, por forma a ser possível percorrer a informação contida numa ontologia.

2.1 Estrutura e povoamento da ontologia

Para este trabalho, foi escolhido o tema *Ténis*. Para abordar os pontos mais importantes acerca desta modalidade, decidiu-se representar informação acerca de jogadores, equipas, torneios e partidas(individuais) desde o início da *Era Open*, ou seja, desde 1968. Na figura seguinte apresenta-se a estrutura da ontologia definida. Existem as classes *Torneio*, *Fase*, *Partida*, *Jogador*, *Equipa* e *País*. Também foram definidas relações que representam informações como: um jogador pode jogar numa partida (temVencedor/vencedorEm ou temPerdedor/perdedorEm); um jogador faz parte de uma equipa (éElementoDe/temElemento); uma equipa representa um país (éEquipaDe/temEquipa); uma partida faz parte de uma dada fase/ronda (éPartidaDe/temPartida); uma ronda faz parte de um torneio (éFaseDe/temFase).

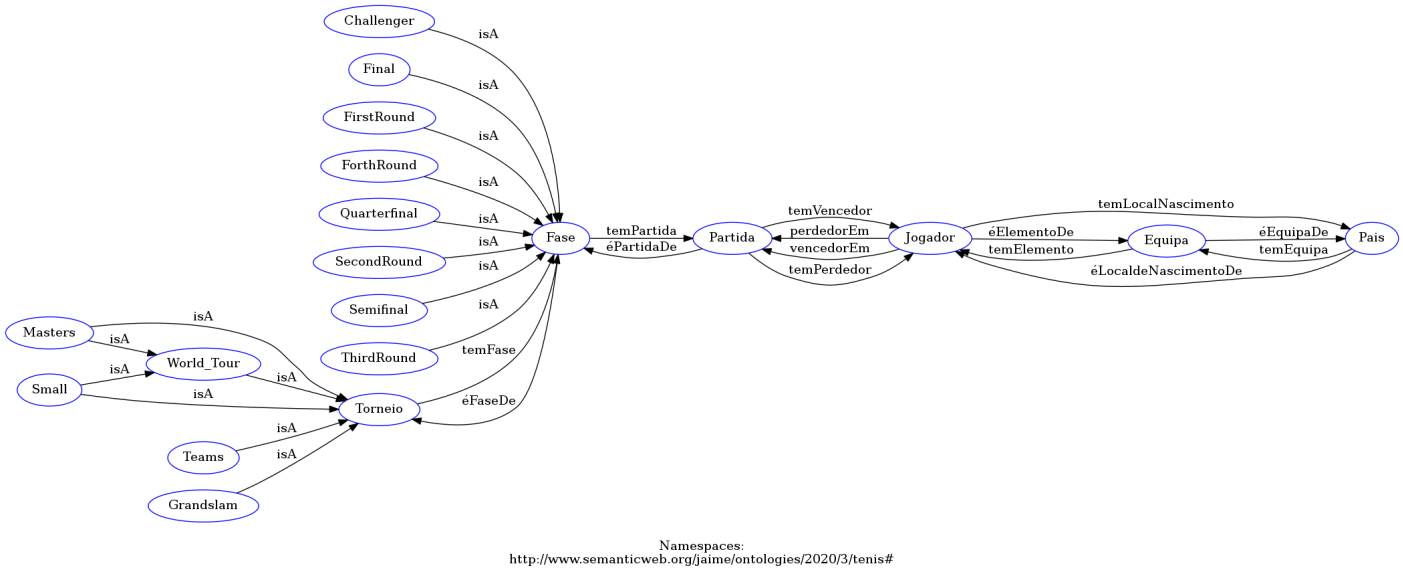

Figura 2.1: Estrutura da ontologia

Também é possível visualizar que podem existir vários tipos de fase, nomeadamente *FirstRound*, *SecondRound*, *ThirdRound*, *ForthRound*, *Quarterfinal*, *Semifinal*, *Final* e *Challenger*. Já um torneio pode ser *GrandSlam*, *Teams*, *World_Tour*, *Masters* ou *Small*. De notar que estes dois últimos pertencem à classe *World_Tour*, sendo representados todos os torneios que, não sendo de equipas, não pertencem à classe *GrandSlam*.

No que toca à procura de informação e povoamento da ontologia, foi filtrada informação que se encontra no repositório do *GitHub*, `https://github.com/JeffSackmann/tennis_atp`, nomeadamente a que se encontra desde o ficheiro *atp_matches_1968.csv* até ao ficheiro *atp_matches_2020.csv*, que contém os jogos individuais desde o início da era open. Foi também aproveitado o ficheiro *atp_players.csv*, que contém informações acerca dos jogadores de ténis. A informação contida nestes ficheiros, que estão no formato *CSV* foi convertida para *JSON* usando um conversor online, por forma a ser mais simples a escrita da informação em *Turtle*. Para visualizar qual a estrutura de que cada indivíduo de cada classe teria que ter, foram criados indivíduos à mão no *Protege*, para cada classe, e depois feita a exportação da ontologia para um ficheiro. Tendo esta estrutura da ontologia formada, foram usados vários scripts em *Python* para converter para *Turtle* a informação contida nos ficheiros *JSON*. Esta informação relativa às classes, *object properties* e *data properties* foi inserida num repositório do *GraphDB*.

2.2 Servidor de API e servidor de interface

Para ser possível aceder à informação contida no repositório mencionado anteriormente, foi necessário definir queries *SPARQL* para cada tipo de informação a consultar. Para tal, foi desenvolvido o servidor de *API*, em *NodeJs*, apoiado por *Express*, que faz a ligação entre os dados da ontologia e a interface. Este servidor é definido por um conjunto de rotas principais, que posteriormente se dividem em subrotas para ser possível aceder a informação mais concreta. Assim, o conjunto das rotas principais caracteriza-se por '/jogadores', '/torneios', '/fases', '/partidas', '/equipas', '/curiosidades', que agrupam as queries relativas jogadores, torneios, fases, partidas, equipas e curiosidades, respetivamente. De seguida apresenta-se um excerto de uma função que se encontra no controller dos torneios e que permite obter todos os tipos de superfície dos torneios.

```
Torneios.getSuperficies = async function(){
    var query = ‘select distinct ?superficie where {
        ?torneio a :Torneio .
        ?torneio :superfície ?superficie .
    }‘
    ...
}
```

Relativamente ao servidor de interface, este foi desenvolvido em *VueJs* e com o intuito de percorrer e mostrar a informação contida na ontologia. A informação que é apresentada na interface *Web* é agrupada em quatro grandes partes: *Jogadores*, *Equipas*, *Torneios* e *Curiosidades*, sendo estas também as partes que constituem o menu lateral da interface e o conjunto das rotas. Apresenta-se de seguida o que é possível visualizar na interface dependendo da rota em que se está, usando o endereço *http://localhost:8080*:

- /equipas/listagem: permite visualizar a listagem das equipas que participaram na *Davis Cup*;
- /equipas/:nomeEquipa: permite visualizar os anos em que uma equipa participou na *Davis Cup*;
- /jogadores/lista: permite visualizar a lista dos jogadores de ténis;

- /jogadores/:idJogador: permite visualizar o perfil individual de um jogador de ténis;
- /torneios/menuTorneios: permite escolher os parâmetros para filtrar os torneios;
- /torneios/listaTorneios: permite visualizar a listagem de torneios com uma filtragem na pesquisa;
- /torneios/escolherAno: permite escolher um ano em que se pretende visualizar um dado torneio;
- /torneios/fases/:idFase: permite visualizar as partidas de uma fase de um torneio, num dado ano;
- /torneios/partidas/:idPartida: permite visualizar informação acerca de uma partida de ténis;
- /torneios/:idTorneio: permite visualizar informações gerais de um torneio de ténis, para um dado ano;
- /curiosidades/menuCuriosidades: permite escolher a curiosidade que se pretende visualizar;

Na interface desenvolvida, foi tido sempre o cuidado de colocar uma ligação entre os vários tipos de informação que estão representados na ontologia. Por exemplo, partindo do perfil individual de um jogador, é possível redirecionar para a página de um torneio, num dado ano. É também possível visualizar o perfil individual de cada jogador, partindo do perfil de uma equipa. Para um dado torneio, em cada uma das fases/rondas realizadas em cada ano, são apresentadas as partidas, com as respetivas durações, resultados, vendedores e perdedores(nomes dos jogadores aparecem como elementos clicáveis que redirecionam para o perfil individual do mesmo).

Relativamente às curiosidades, representam informações globais acerca da relação entre as entidades da ontologia. De entre as curiosidades estão, por exemplo, o top dez dos jogadores com mais vitórias, o top 10 jogadores com mais títulos ou o top 10 equipas com mais títulos. Na figura seguinte, apresenta-se esta última curiosidade mencionada.

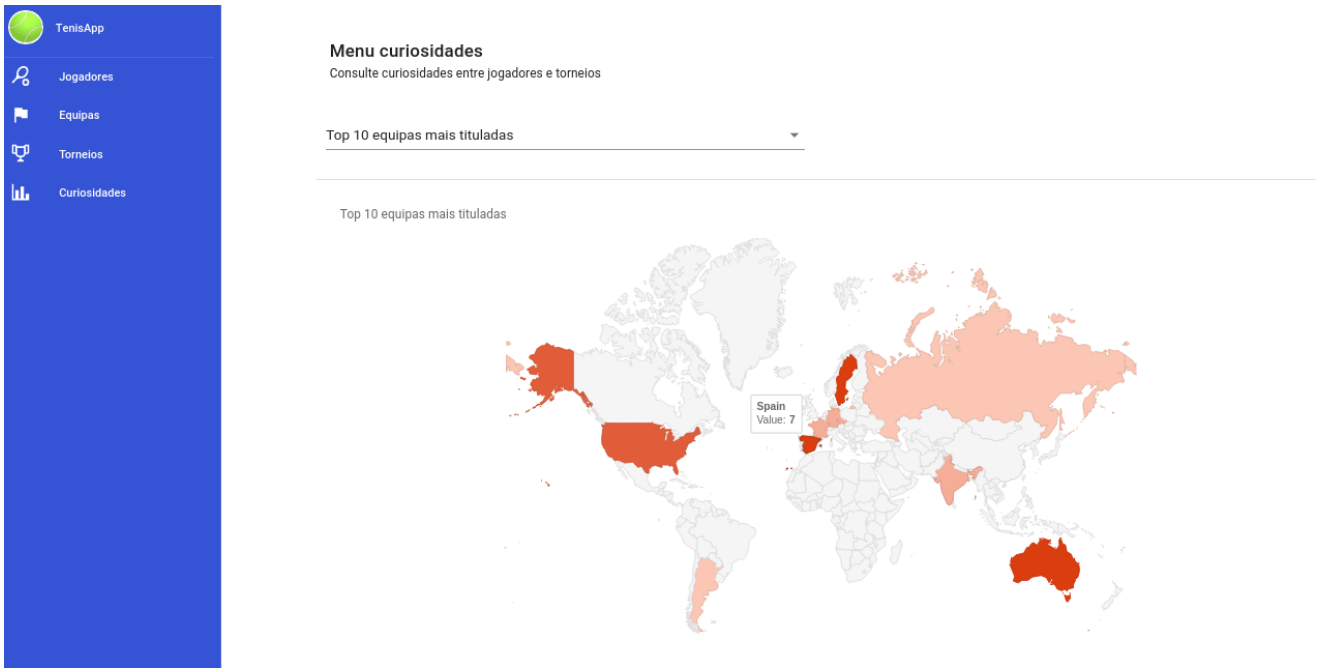

Figura 2.2: Estrutura da ontologia

Na figura, é possível observar as dez equipas com mais títulos na *Davis Cup*, que é um torneio anual para países disputarem um título a nível global. Quanto mais saliente for a cor do país no mapa mundo, significa que mais títulos a sua equipa possui.

Capítulo 3

Conclusão

As ontologias permitem permite representar os dados de forma bem estruturada e sem ambiguidades. As ferramentas como o *VueJs* permitem acessar e mostrar informação de forma simples e rápida, podendo interagir facilmente com *API's* de repositórios de dados, como é o caso do *GraphDB*. Estes foram os fundamentos para se implementar este projeto.

No desenvolvimento deste trabalho, foram considerados os pontos mencionados anteriormente. Posteriormente à escolha do tema para o trabalho, foi definida uma ontologia capaz de representar informação acerca do mesmo. Foram usados vários scripts em *Python* para converter para *Turtle* a informação que estava no repositório do *GitHub*(mencionado no Capítulo 2) em ficheiros *CSV* e que sofreu uma conversão para *JSON*. Posteriormente ao povoamento da ontologia, foi definido um servidor de *API* para efetuar a ligação entre o repositório que armazena informação da ontologia e o servidor de interface *web*. Este último, desenvolvido em *VueJs*, permite percorrer a informação da ontologia.

Os objetivos do projeto foram alcançados, apesar de haverem alterações que poderiam ser aplicadas, como por exemplo, apresentar na interface *web* informação ainda mais específica da ontologia, explorando em maior pormenor as relações entre as entidades das diferentes classes desta.