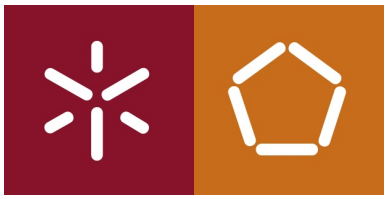

Universidade do Minho

Mestrado Integrado em Engenharia Informática

Processamento e Representação de Conhecimento

Hearthstone

Raul Vilas Boas, a79617

13 de Junho de 2019

Resumo

O presente trabalho foi desenvolvido no âmbito da unidade curricular Processamento e Representação de Conhecimento do 4º ano do Mestrado Integrado em Engenharia Informática da Universidade do Minho, sendo que esta tem como objetivo a aprendizagem prática de conceitos dos desenvolvidos associados ao perfil de Processamento de Linguagens e Conhecimento.

O trabalho consiste na escolha de um tema, na procura de um *dataset* relativo ao mesmo e no seu desenvolvimento de forma a ser possível disponibilizar a sua informação de forma fácil e acessível.

Conteúdo

Capítulo 1

Introdução

O projeto desenvolvido consiste na utilização das abordagens desenvolvidas na unidade curricular de PRC de modo a que com base num determinado *dataset* criar uma *web app* que reflita o conteúdo desse *dataset*. Para este trabalho, o tema escolhido tem como base um jogo de cartas Hearthstone, de modo que o *dataset* vai incidir sobre a informação contida sobre as cartas. Sendo que o objetivo do trabalho é ser capaz de mostrar a informação contida sobre o tema de forma fácil e acessível com ajuda da criação de uma ontologia para auxiliar na representação dos conceitos e as relações dentro deste domínio.

Deste modo, podemos distinguir alguns componentes de relevância cujo desenvolvimento consideramos fundamental como o uso da ferramenta *protégé* que auxiliou na criação da ontologia, o GraphDB que serviu como um base de dados sendo que usa grafos como estrutura para a semântica das *queries*, o *node* para se conseguir extrair a informação disponibilizada pelo GraphDB e o vue para disponibilizar essa informação.

Assim sendo, tendo as várias componentes disponíveis é relevante entender brevemente a metodologia usada para a resolução do problema. Primeiro, vamos começar por fazer uma breve contextualização do tema escolhido, depois passamos à criação da ontologia e do tratamento do *dataset* e a sua importação para o GraphDB. Depois, por último, ao desenvolvimento do *backend* utilizando *node* para extrair a *query* do GraphDB e do *frontend* utilizando vue para mostrar a informação com base numa *app*.

Capítulo 2

Concepção da Resolução

2.1 Tema

O tema escolhido para o trabalho consiste num jogo de cartas desenvolvido pela empresa *Blizzard Entertainment* denominado *Hearthstone*. O tema foi escolhido devido à facilidade de acesso a informação relativa ao jogo e também devido à riqueza em termos de variedade. Numa versão simplificada do jogo, ele consiste num duelo entre dois jogadores em que cada um possuí 30 cartas e 20 pontos de vida e o objetivo é fazer descer os pontos de vida do outro jogador até 0. Ambos os jogadores podem escolher entre diferentes classes que lhes permitem utilizar diferentes cartas e as cartas também possuem diferentes raridades assim como outros atributos. Ao longo do tempo são lançadas mais cartas que tem associado a si uma expansão na qual foram lançadas. As cartas possuem também um tipo de forma a as distinguir, isto é, podem ser monstros ou magias ,entre outros. Algumas cartas tem associadas a si um valor de custo que representa o valor necessário que se gasta para se a poder usar assim como um valor de ataque, de vida e um texto para representar o que fazem. Visto haver um grande número de cartas diferentes o *dataset* contém um volume de dados aceitável para o desenvolvimento do projeto.

2.2 Ontologia

Com base no jogo foi desenvolvido uma ontologia preparada para ser possível responder a toda a informação contida no *dataset*. Para isto utilizou-se a ferramenta protégé.

- Classes:

  Card - classe que representa as cartas;

  PlayerClass - classe que representa a classe do jogador;

  Set - classe que representa a expansão em que a carta saiu;

- Relações:

  hasPlayerClass - relação entre a carta e a classe do jogador;

  hasSet - relação entre a carta e a expansão em qual saiu;

  controls - relação inversa da hasPlayerClass;

  contains - relação inversa da hasSet;

- Atributos:

Name - Nome da carta;

Cost - Valor gasto ao usar a carta;

Attack - Valor de ataque;

Health - Valor de vida;

Rarity - Raridade associada;

Type - Tipo da Carta;

Text - Descrição da carta;

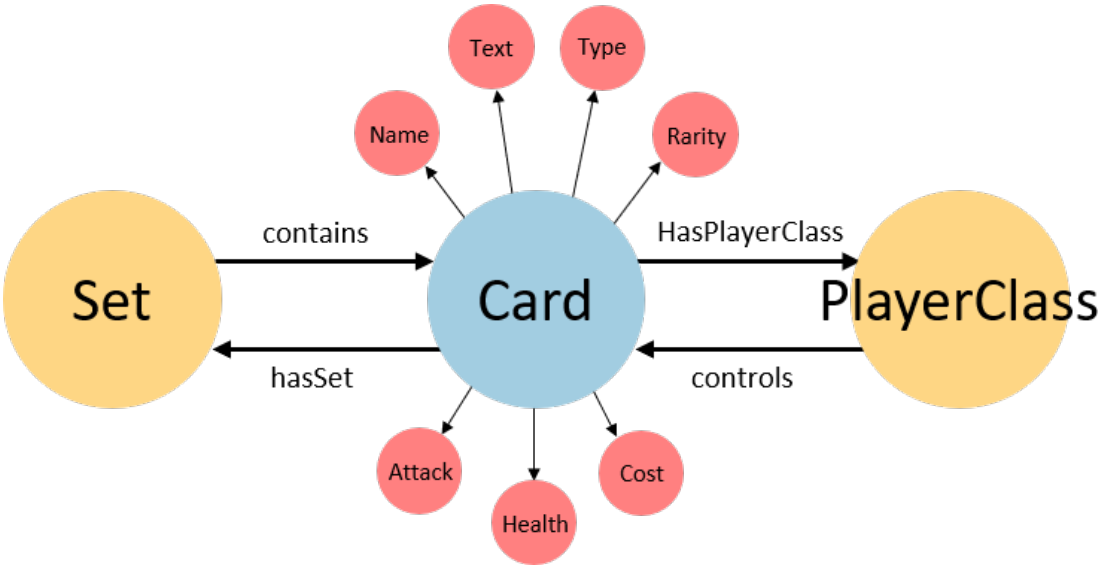

Figura 2.1: Grafo da ontologia criada.

Após ter a ontologia criada com tudo o que iria ser necessário foi possível então passar a fase seguinte que foi o tratamento dos dados. Isto é tratdar o *dataset* e retirar de lá a informação para em conjunto com a ontologia ser possível depois povoar o GraphDB.

2.3 Estrutura de Dados

Para tratar dos dados e os juntar à ontologia criada desenvolveu-se um *script* em python para tratar da informação. A metodologia usada foi percorrer o *dataset* pelas colunas procurando a coluna pretendida e depois ao retirar a sua informação atribuia-se a conotação correta da ontologia caso fosse uma classe, uma relação ou um atributo. Depois de obter tudo corretamente, concatenou-se o resultado com o ficheiro das ontologias e criou-se assim o ficheiro do povoamento que irá ser usado para importar no GraphDB.

No *GraphDB* verificou-se se tudo foi importado corretamente e depois testou-se algumas *queries* para se confirmar o acesso à informação. Na figura 2.2 temos um exemplo de um visão global de algumas cartas da *playerclass Warrior* assim como as suas relações.

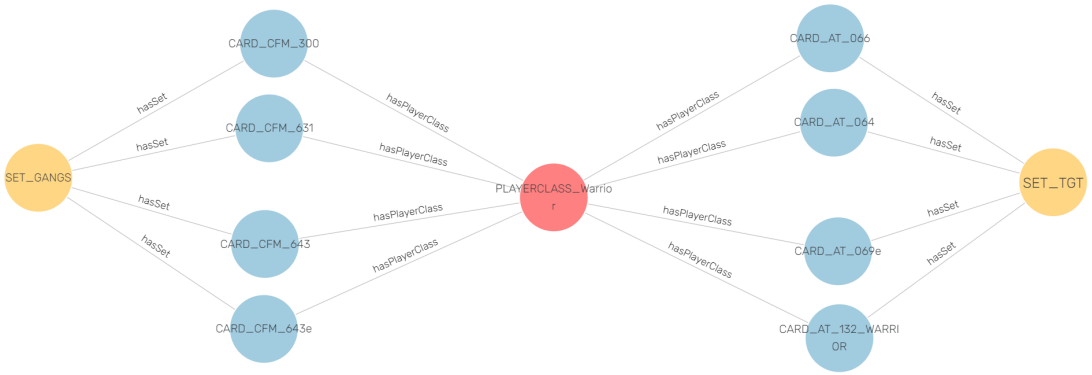

Figura 2.2: Exemplo no GraphDB.

2.4 Backend/Frontend

Para o *backend* utilizou-se um servidor *node* que vai receber os pedidos efetuados pelo *frontend*. Para responder as estes pedidos o servidores *node* vai buscar a informação ao GraphDB usando *queries*, que foram previamente feitas, capazes de satisfazer os pedidos necessários. Os pedidos enviados pelo servidor seguem a API do GraphDB que retornam a sua resposta em JSON que é passado para o *frontend* e por sua vez é lá tratada a informação.

De modo a facilitar a compreensão foram divididas as *queries* em diferentes ficheiros devidos às diferenças entre elas, pois algumas precisavam de receber argumentos enquanto que outras não. Sendo assim, foram criadas várias *routes* no servidor para ser possível aceder à informação fornecida por cada uma destas *queries*.

De seguida segue-se uns exemplos de algumas *queries* criadas.

```
PREFIX : <http://www.semanticweb.org/raulv/ontologies/2019/5/projeto#>
select ?id ?name ?set ?playerclass ?rarity ?type ?attack where {
    ?id ?p :Card.
    ?id :name ?name.
    ?id :hasSet ?set.
    ?id :hasPlayerClass ?playerclass.
    ?id :type ?type.
    OPTIONAL {?id :rarity ?rarity.}

tabela_filtros: function (set,player_class,type,rarity) {
    var query = `PREFIX : <http://www.semanticweb.org/raulv/ontologies/2019/5/projeto#>
    select ?id ?name ?set ?playerclass ?rarity ?type where {
        ?id ?p :Card.
        ?id :name ?name.
        ?id :hasSet ?set.
        ?id :hasPlayerClass ?playerclass.
        ?id :type ?type.
        ?id :rarity ?rarity.\n`
    if(set)
      query += '\t?id :hasSet :SET_'+set+' .\n'
    if (player_class)
      query += '\t?id :hasPlayerClass :PLAYERCLASS_' + player_class + ' .\n'
```

```
  if (type)
    query += '\t?id :type "' + type + '" .\n'
  if (rarity)
    query += '\t?id :rarity "' + rarity + '" .\n'
  query+='}'
  return query
}
```

A primeira *query* permite para todas as cartas retornar a informação relativa ao seu nome, set, classe tipo e raridade. Esta *query* foi utilizada para a construção de um tabela em que cada coluna é um destes campos. A segunda *query* foi usada para a criação de filtros para filtrar a informação, isto é, caso num dos filtros estivesse definido um valor para a classe do jogador, esse valor era adicionado a esta *query* tornando-a mais especifica.

Após ter a informação disponibilizada começou-se a tratar do *frontend*, no qual foi usado vue. De uma forma simplificada foram criadas 4 *routes*. Uma para a *home* e outras 3 para apresentar a informação relativa ao tema. Dessas mesmas páginas, criou-se uma para mostrar a informação relativa a cada carta individual, outra em que se colocou a informação de todas as cartas numa tabela e, por último, uma página que mostra todas as imagens de todas as cartas.

Para o caso da página que continha a tabela criou-se na seguinte *route* `http://localhost:8080/cartas` que consiste numa tabela que apresenta a informação recebida pela primeira *query* apresentada anteriormente. No entanto, existe também a opção de procurar por alguma carta em especifica utilizando o *search* mas também permite filtrar as informações das colunas com os filtros criados. Isto é, é possível fixar a *playerclass* em *Warrior* e depois na tabela apenas mostrará as cartas que cumpram este requisito. Para isto ser possível, é utilizada a função apresentada na 2 *query* mostrada anteriormente, que quando algum filtro se encontre ativo vai adicionar mais especificidade à *query*.

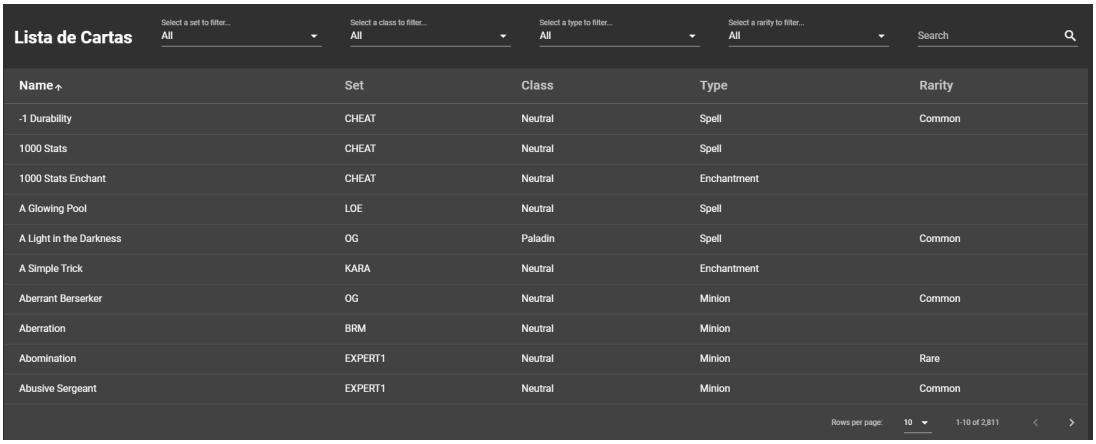

Figura 2.3: Tabela das cartas.

De modo a tornar a aplicação mais acessível, para cada entrada da tabela caso ele seja selecionada o utilizador e então rederecionado para a página individual dessa mesma carta, sendo o *route* dela `http://localhost:8080/` Nesta página é apenas apresentada toda a informação existente da respetiva carta, assim como uma imagem da carta, caso esta exista.

Figura 2.4: Página individual de um carta.

Por último, dispomos de outra página na *route* `http://localhost:8080/cartasview` que mostra todas as imagens das cartas com o seu título correspondente. Nesta página também é possível pesquisar pelo nome da carta assim como filtrar as cartas por *set, playerclass, type, rarity*. Caso um carta seja selecionada, o utilizador é redirecionado para a página individual dessa mesma carta.

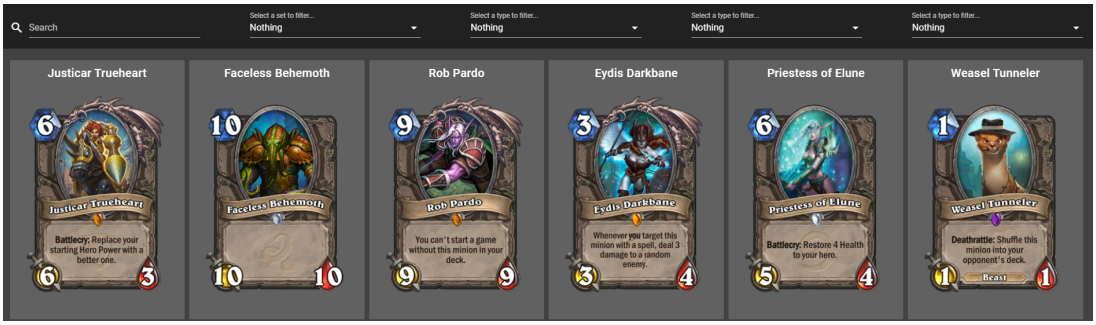

Figura 2.5: Página com imagens das cartas.

Capítulo 3

Conclusão

O projeto detalhado neste relatório consistiu da escolha, tratamento e exploração do conhecimento presente num *dataset* relativo ao conhecido jogo de cartas *Hearthstone*. Tal objetivo foi atingido através da criação de uma ontologia (modelo de dados que permite a representação de conhecimento) e, por fim, uma aplicação web para que um dado utilizador possa aceder à informação de forma simples, eficiente e intuitiva através das várias páginas, filtros e opções disponibilizadas na mesma.

A nível de ferramentas e tecnologias utilizadas recorreu-se maioritariamente a NodeJS, Vue e ao GraphDB o que permitiu aprimorar os conhecimentos adquiridos ao longo do semestre na unidade curricular na qual o projeto se insere, nomeadamente, Processamento e Representação de Conhecimento.

Por fim, faz-se uma apreciação positiva quanto ao trabalho desenvolvido e, a nível de trabalho futuro, sugere-se