Universidade de Coimbra

Mestrado em Engenharia Informática

Web Semântica

TV Series

Projeto

Autores:
Inês Coelho
Joaquim Leitão

Número de Estudante:
2004104311
2011150072

3 de Janeiro de 2016

Resumo

Para agregar informação sobre séries de televisão, atores e criadores, desenvolveu-se um sistema de Web Semântica. Baseado em web, o pilar do sistema é uma ontologia construída sobre as *frameworks Protégé e Apache Jena*, descrita pela linguagem *OWL DL* e populada com informação recolhida a partir do site *IMDB* através da técnica de *screen scapping*. Através da interface web, os utilizadores podem interagir com o sistema, que oferece funcionalidade de consulta, pesquisa e recomendação sobre a ontologia, permitindo usufruir em pleno das vantagens do armazenamento de informação num sistema de Web Semântica.

Palavras Chave: Séries; Ontologia; Web Semântica; Apache Jena; Protégé.

Conteúdo

1 Introdução

1.1 Objetivo

O presente trabalho prático tem como objetivo desenvolver um sistema de Web Semântica que agregue informação sobre séries de televisão, permitindo ao utilizador consultar e realizar pesquisas semânticas sobre essa informação através de uma página *web*, que oferece também recomendações de informação a visualizar com base no histórico de visualização dos dados pelo utilizador.

Este trabalho foi desenvolvido no âmbito da disciplina de Web Semântica, lecionada para o mestrado de Engenharia Informática da Universidade de Coimbra no ano letivo de 2015/2016.

1.2 Âmbito

O sistema de Web Semântica desenvolvido procura relacionar dados de diferentes séries com os seus criadores e com os atores que nelas participam utilizando uma ontologia. Para isso, recorre a uma *triple Store* (Apache Jena), utiliza RDFs para especificar o modelo de dados e utiliza OWL como linguagem formal de especificação da ontologia.

Para popular a ontologia, o sistema utiliza um *web crawler* capaz de recolher e de manipular informação de *websites* que não expõe a sua API para o público em geral.

O sistema funciona num ambiente de rede baseado em internet e utiliza SPARQL como linguagem de *query* para recolher e manipular a informação da ontologia, de forma a interagir com a plataforma web.

A sua *interface web* permite ao utilizador interagir e realizar operações de pesquisa sobre a informação armazenada na ontologia, e apresenta recomendações de páginas a visualizar ainda não visitadas, com base no histórico de visualização do utilizador.

1.3 Estrutura do Relatório

O relatório deste projeto encontra-se estruturado em 6 secções.

A arquitetura do sistema de Web Semântica desenvolvido é apresentado na seção 2. A seção 3 debruça-se sobre a ontologia utilizada no sistema, o seu desenho e construção.

A seção 4 aborda os módulos do sistema, referindo os algoritmos e tecnologias utilizadas na sua construção. De seguida, a seção 5 refere-se ao código do sistema, fazendo uma breve descrição de alto nível da sua composição.

Por fim, são apresentadas as principais conclusões obtidas com este projeto.

2 Arquitetura do Sistema

A presente secção é reservada para, numa fase inicial, se proceder à apresentação da arquitetura do sistema, procedendo-se posteriormente à especificação dos principais componentes que o constituem.

2.1 Vista da Arquitetura

O sistema desenvolvido apresenta uma arquitetura do tipo *Model-View-Presenter* (MVP). O modelo de dados é constituído por 3 componentes: o sistema de gestão de conteúdos, o editor da ontologia e a ontologia. O servidor REST constitui a camada de apresentação. A interface gráfica do sistema constitui a camada de apresentação.

A arquitetura do sistema encontra-se representada na figura 1.

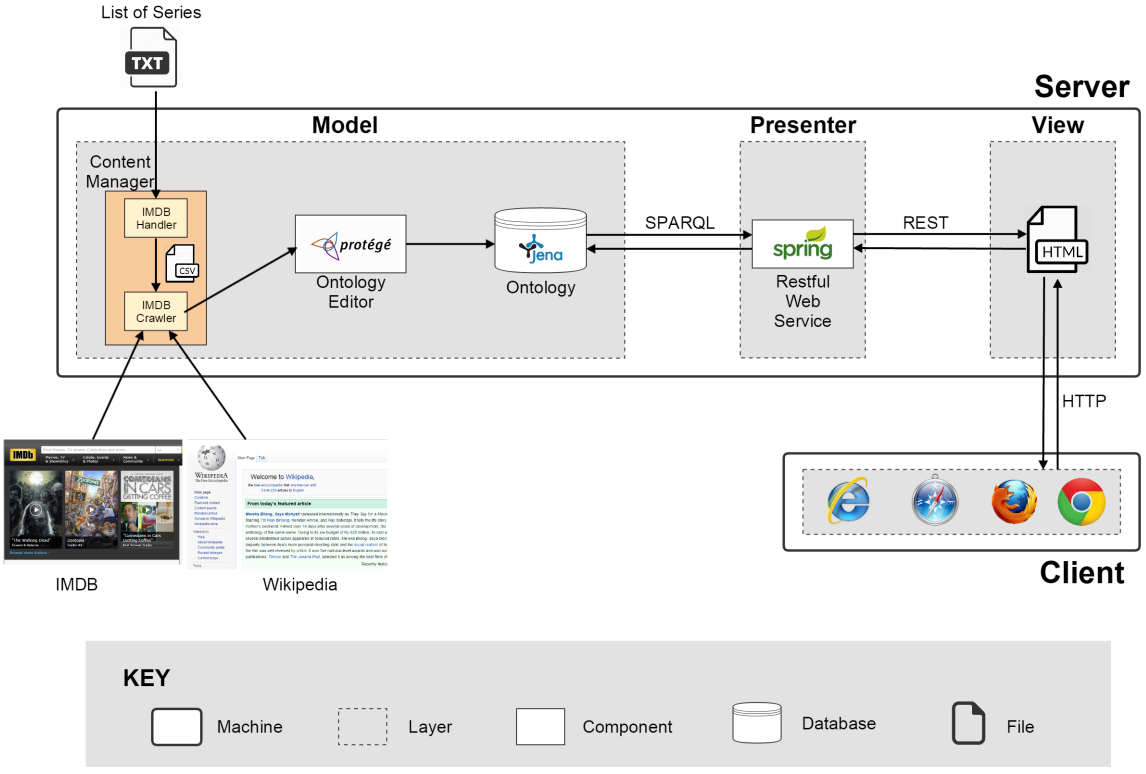

Figura 1: Arquitetura do Sistema.

2.2 Sistema de Gestão de Conteúdos

Os dados contidos no sistema de Web Semântica desenvolvido provêm de informação disponibilizada na internet e são recolhidos através de um sistema de gestão de conteúdos.

A aplicação de gestão de conteúdos é constituída por 2 componentes principais, fornecidos em conjunto com este relatório: *IMDB Handler* e *IMDB Crawler*.

O primeiro componente (presente na pasta *IMDBHandler*) tem como função percorrer a lista de séries presente no ficheiro de texto acima referido e, para cada série, obter o seu identificador único no site IMDB, procedendo ao armazenamento do título da série em questão e do seu identificador num ficheiro *CSV*.

O segundo componente (presente na pasta *TVSeries*) tem como principais funções obter a informação relativa a cada série listada a partir do site do IMDB e persistir essa mesma informação numa ontologia previamente criada e detalhada na secção 3. Este sistema funciona como um *web crawler*, utilizando a técnica de *screen scrapping* para recolher informação da internet. O processo de população da ontologia será abordado com maior detalhe na subsecção que se segue.

O nome das séries das quais se pretende recolher informação é fornecido à aplicação de gestão de conteúdos através de um ficheiro de texto.

2.3 Ontologia

De acordo com o detalhado na secção 3.2, a informação recolhida para cada série não se restringe apenas a dados exclusivos da série (como título, classificação, ano de estreia, etc) englobando também a sua lista de atores e criadores.

Assim, para obter a informação de cada série o segundo componente apresentado na subsecção anterior carrega a lista de séries e respetivos identificadores para memória e, em seguida, para cada série em questão, recorre à técnica de *screen scrapping* para obter as informações pretendidas a partir da sua página no IMDB.

À medida que a informação de cada série vai sendo recolhida esta é persistida em memória[1] para que, assim que o processo de *web crawling* esteja concluído, esta possa ser inserida numa ontologia previamente criada, utilizando a *framework Apache Jena*[2] e tirando partido da sua implementação de uma *triple store*. Embora esta *framework* também suporte a criação de ontologias, neste trabalho recorremos ao editor *Protégé*[3] para esta tarefa, usufruindo assim da interface gráfica desta ferramenta.

Para persistir toda a informação na ontologia, este componente obtém, para cada série, as suas informações persistidas em memória, procedendo à criação do indíviduo

[1] Isto é, numa lista onde cada elemento contém as informações de uma série

[2] `https://jena.apache.org/`

[3] `http://protege.stanford.edu/`

correspondente na ontologia. Este passo não se resume apenas à inserção de informações base da série em questão (como título, classificação, entre outros), incluíndo também a inserção dos seus atores e criadores e das respectivas ligações entre ambos ao nível da ontologia.

Findo este passo, uma última tarefa é ainda realizada por este componente. Esta tarefa consiste na inserção de *hyperlinks* relativos a páginas web com informação de cada pessoa (ator ou criador) inserida na ontologia. Após a inserção de toda a informação recolhida para as séries e os seus atores e criadores constatou-se que a informação presente no IMDB para muitos atores e criadores era escassa, tendo sido tomada a decisão de adicionar uma referência para uma página pessoal de cada ator ou criador, de forma a extender a informação armazenada para cada entidade.

Assim, para esta tarefa, a aplicação limita-se a percorrer todas as pessoas presentes na ontologia e, para cada uma, verificar se esta possui uma entrada na wikipédia. Para isso recorre-se ao primeiro e último nomes da pessoa em questão para gerar uma *hiperligação* da wikipédia, cuja existência é de seguida verificada e, caso seja confirmada, a *hiperligação* gerada é adicionada à informação da pessoa em questão.

2.4 Servidor REST

Para suportar a interacção com a ontologia em todas as operações realizadas na interface web foi desenvolvido um pequeno servidor *REST*, utilizando a *framework Spring*[4]. Este servidor, que se encontra na pasta *RESTServer* enviada em conjunto com o presente relatório, disponibiliza uma série de serviços que permitem obter informações relativas às séries, atores e criadores armazenados na ontologia, de forma a poderem ser disponibilizadas na interface web. A lista de serviços disponibilizados por este servidor REST inclui:

- Listagem dos géneros das séries armazenadas na ontologia
- Listagem das séries pertencentes a um dado género
- Listagem das informações de uma dada série armazenada na ontologia
- Listagem das informações de uma dada pessoa, seja ator ou criador, armazenada na ontologia
- Processamento de operações de pesquisa semântica na ontologia mediante a introdução, pela parte do utilizador, de uma pequena frase contendo as palavras-chave da pesquisa a ser efectuada
- Processamento de operações de recomendação de séries e pessoas cuja informação ainda não foi consultada, tendo em conta as séries e pessoas consultadas pelo utilizador

Recorrendo a este servidor obtemos uma camada de abstração e de interface no acesso à ontologia, tornando o sistema mais resistente a alterações realizadas na

[4] `https://spring.io/`

ontologia (por exemplo ao nível dos campos de cada classe).

Mais ainda, caso se pretenda portar o sistema para um nova configuração onde uma nova ontologia, ou outra solução (como por exemplo a adopção de uma base de dados em detrimento da ontologia) é utilizada, apenas necessitamos de proceder às devidas alterações neste servidor, podendo manter os serviços por ele disponibilizados intactos. Isto implica que, caso se procedam a estas alterações, toda a implementação da interface web não necessita de sofrer qualquer alteração para se manter compatível com as novas configurações.

De facto, esta interacção com a ontologia poderia ter sido implementada do lado da interface web (necessitando de ser realizada do lado do servidor e não no browser do cliente). No entanto, dada a simplicidade e facilidade na criação de servidores REST, e a popularidade deste tipo de soluções, optámos por desenvolver um cliente web mais simples e leve, colocando todas as operações e lógica de negócio envoltas por um servidor. Utilizando um serviço REST asseguramos ainda compatibilidade com várias linguagens de desenvolvimento, não só para clientes web mas também para outro tipo de aplicações, o que é por nós visto como uma mais-valia.

2.5 Interface Web

A interface web da aplicação utiliza como linguagens de programação Java, Javascript e HTML. A customização do *website* foi feita recorrendo à *framework front-end Bootstrap* [5].

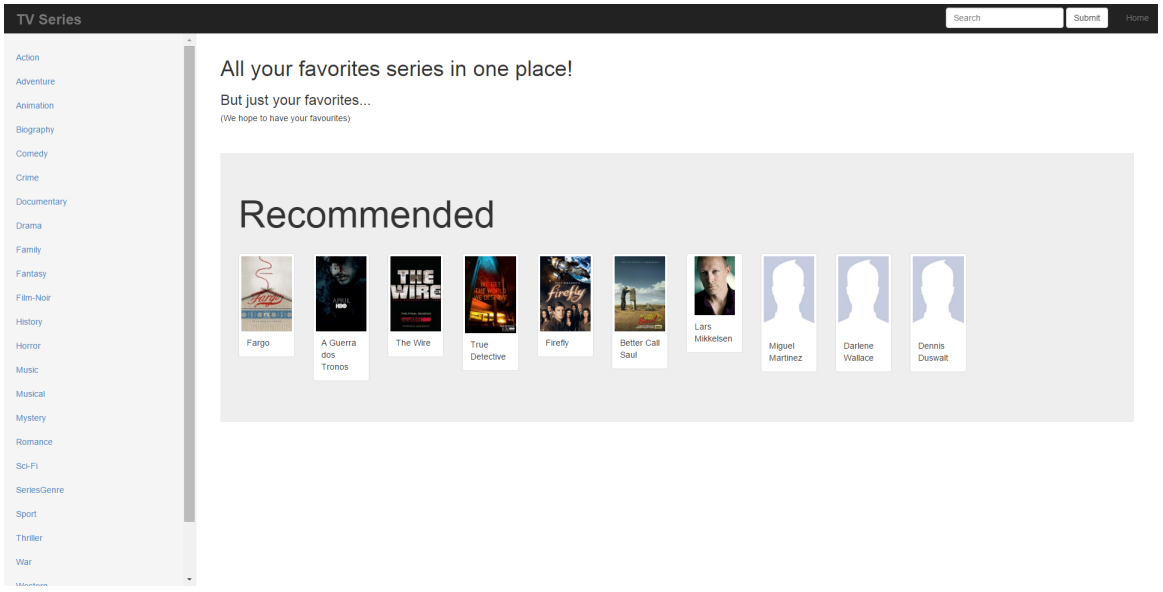

Figura 2: Página Principal da Aplicação TV series.

[5] `http://getbootstrap.com/`

Esta aplicação recorre a *web servlets* para estender as funcionalidades do servidor, de forma a apresentar conteúdo dinâmico. Os Java Servlets são objetos que recebem pedidos da página web e geram uma resposta com base nesse pedido.

Através da interface web, o utilizador pode interagir com o sistema, acedendo às funcionalidades de: navegação (*browsing*), pesquisa e recomendação.

A página principal da aplicação (figura 2) contém informação descritiva sobre a aplicação e uma seção de páginas recomendadas. À esquerda encontra-se o menu de navegação, que lista os géneros de séries disponiveis. A barra de navegação contém uma caixa de pesquisa e um botão que permite o retorno a esta página.

Para navegar na aplicação deve-se começar por selecionar um dos géneros disponíveis no menu de navegação. Desta forma, são listadas todas as séries disponíveis nessa categoria na janela principal da aplicação (figura 3).

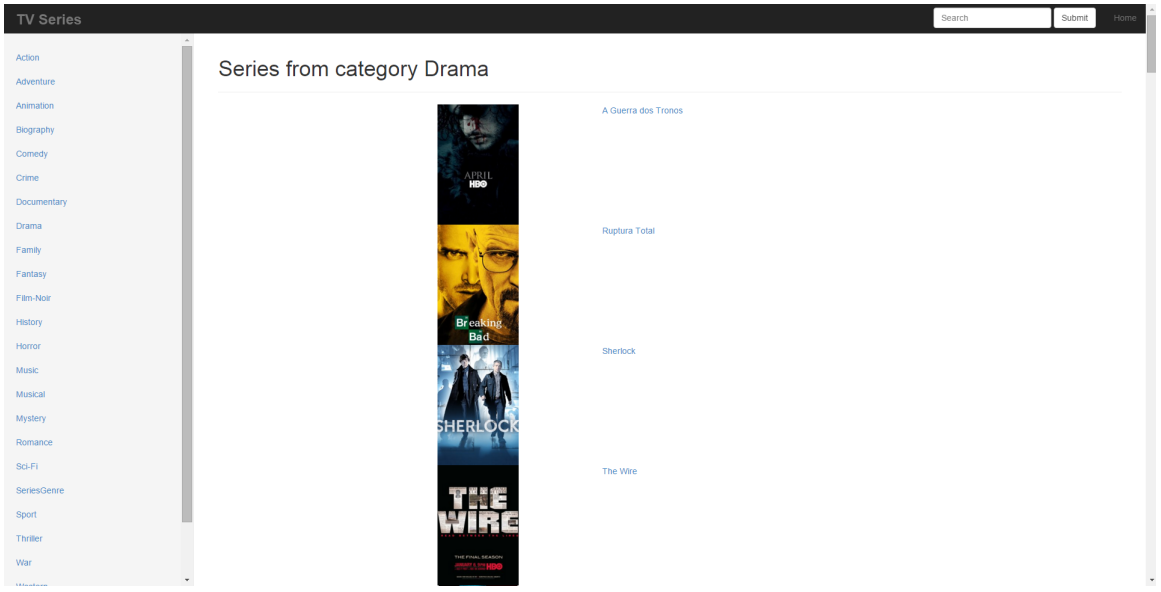

Figura 3: Listagem de séries da categoria Drama pela aplicação TV series.

Selecionando-se uma das séries disponíveis, é se redirecionado para a página da série (figura 4). Nesta página é possível consultar toda a informação disponível sobre a série, incluindo uma listagem dos criadores da série e dos atores, por ordem alfabética.

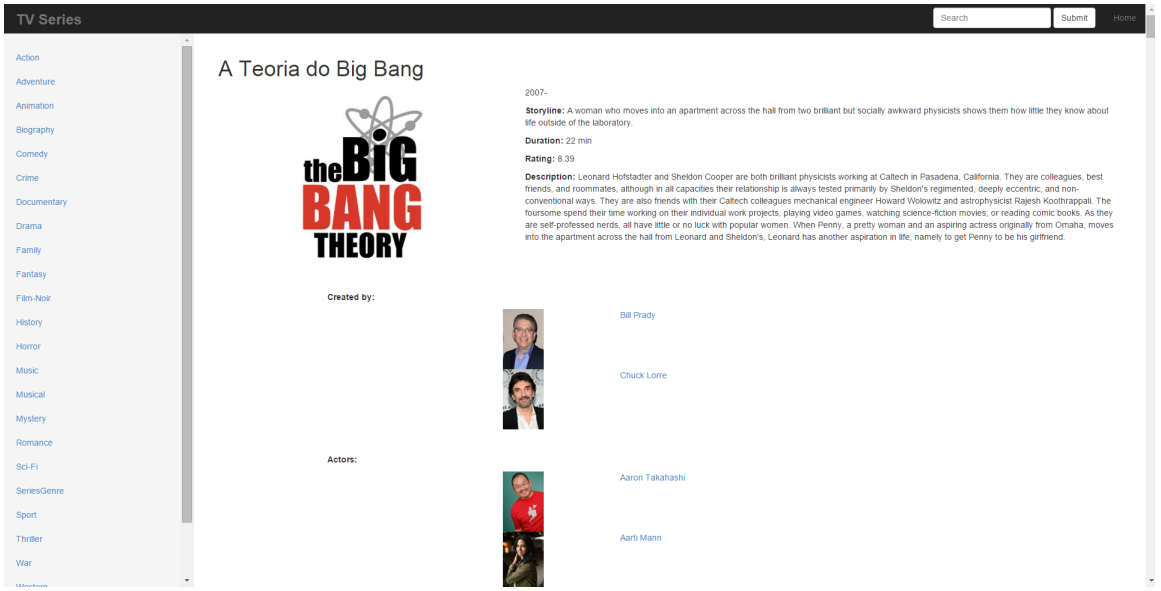

Figura 4: Página da série televisiva Big Bang Theory.

Selecionando-se um ator ou criador de uma série de televisão, é apresentada a página respetiva à pessoa em causa 5). Nesta página é possível consultar toda a informação disponível sobre a pessoa, incluindo um *link* para a respetiva página da wikipédia, caso ista, e uma listagem das séries que essa pessoa dirigou e/ou participou.

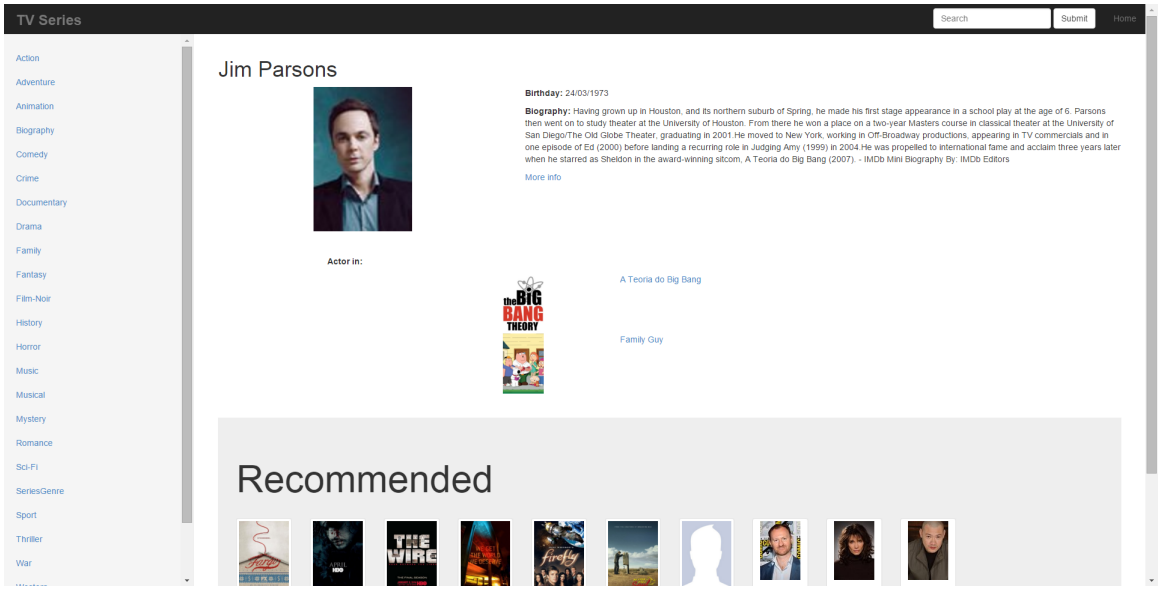

Figura 5: Página do ator Jim Parsons.

A caixa de pesquisa presente na barra de navegação da aplicação permite efetuar pesquisas semânticas sobre a informação armazenada na ontologia. O resultado da pesquisa pode ser do tipo série ou pessoa, sendo os resultados listados numa página semelhante à da figura 6.

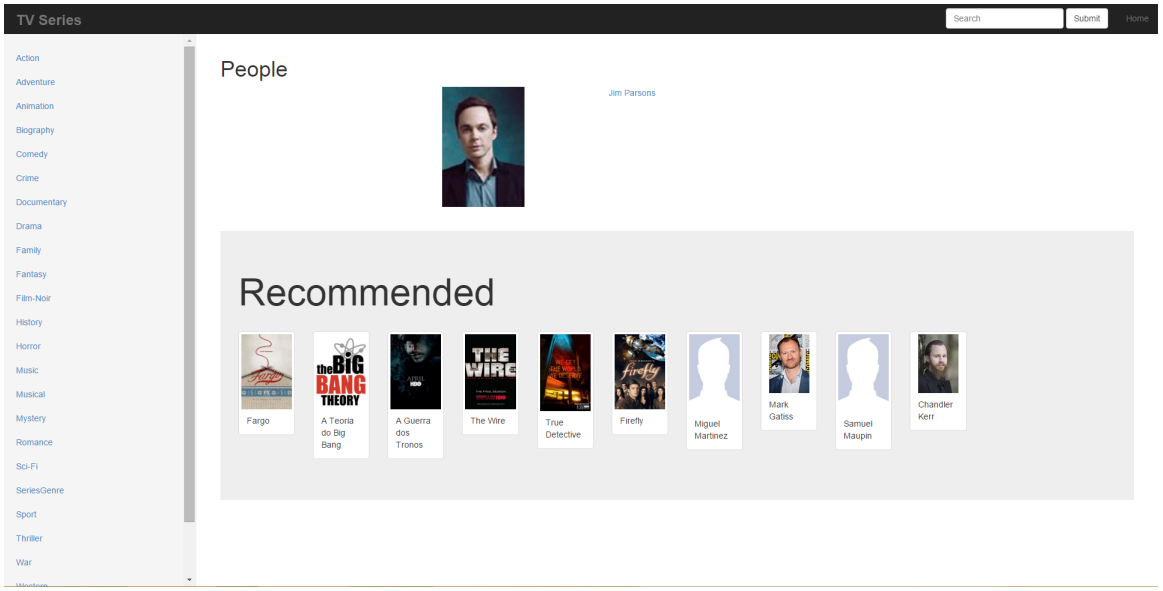

Figura 6: Resultados da pesquisa pelo nome "Jim Parson"

Em todas as páginas da aplicação é possível aceder à caixa de Recomendações (consultar figura 2 e 6), que apresenta recomendações de séries e atores/criadores ainda não visualizados pelo utilizador, com base no seu histórico de visualização.

3 Ontologia

3.1 Introdução

Tal como referido em secções anteriores, o presente sistema procura relacionar dados de diferentes séries com os seus criadores e atores que nelas participam recorrendo para isso, entre outras soluções, a uma ontologia.

Assim, a ontologia a apresentar nesta secção pretende cobrir o domínio das séries televisivas, armazenando informações relativas aos seus géneros, classificação, duração de cada episódio, ano de estreia, atores e criadores, entre outras informações.

Com esta ontologia pretendemos que o sistema construído seja capaz de, para além de disponibilizar as informações recolhidas e armazenadas para as diferentes séries, utilizá-las de forma a dar a conhecer novas séries aos utilizadores, nas quais estes possam estar interessados em conhecer. Também pretendemos que o sistema possa ser utilizado como uma forma simples e prática de obter informações sobre séries e pessoas (neste caso atores ou criadores) partindo de um pequeno conjunto de dados sobre essas entidades (utilizando para isso as funcionalidades de pesquisa, mas também de recomendação, que o sistema oferece).

Esta ontologia será então utilizada, em primeiro lugar e de forma mais directa, pelos desenvolvedores do sistema, estando estes também responsáveis pela sua manutenção. De uma forma mais indirecta, a ontologia criada também será utilizada pelos utilizadores do sistema, uma vez que este sistema necessita da ontologia para realizar as suas funções.

3.2 Definição da Ontologia

Esta subsecção pretende apresentar em maior detalhe a ontologia criada para o sistema, focando a sua taxonomia, os seus componentes e as relações entre si.

3.2.1 Taxonomia

Na ontologia desenvolvida foram criadas duas classes principais, que pretendem abstrair os diferentes tipos de instâncias a considerar: A classe *Person* e a classe *SeriesGenre*. A primeira é utilizada para abstrair os dois tipos de pessoas consideradas (actores e criadores), sendo que a segunda abstrai os diferentes géneros de séries considerados.

Nas figuras 7 e 8 podemos ver, respectivamente para cada classe principal, os seus descendentes hierárquicos.

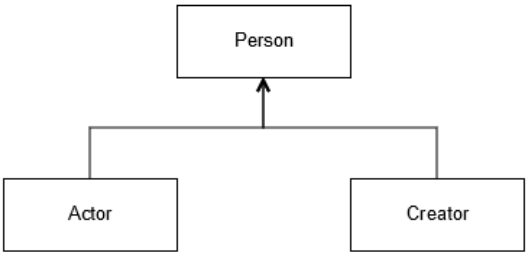

Figura 7: Taxonomia da classe *Person*

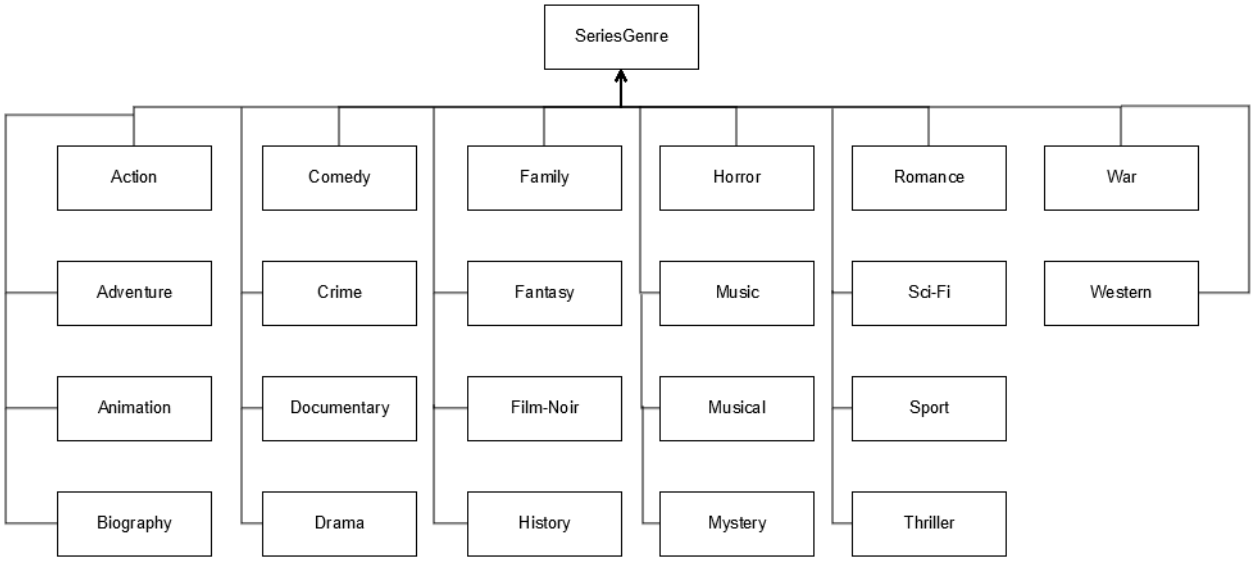

Figura 8: Taxonomia da classe *SeriesGenre*

Relativamente à classe *Person*, podemos identificar duas sub-classes: As classes Actor e Creator, representando respectivamente os Actores e Criadores de séries. Já a classe *SeriesGenre* tem como sub-classes os diferentes géneros de séries considerados. São eles: *Acção*, *Aventura*, *Animação*, *Biografia*, *Comédia*, *Crime*, *Documentário*, *Drama*, *Desporto*, *Família*, *Fantasia*, *Film-Noir*, *Guerra*, *História*, *Horror*, *Música*, *Musical* , *Mistério*, *Romance*, *Sci-Fi*, *Thriller* e *Western*.

Deste modo, cada série persistida na ontologia corresponderá a um indivíduo dos tipos correspondentes aos seus géneros: Por exemplo, uma série de comédia e de animação será dos tipos *Animation* e *Comedy*. Já uma pessoa será representada por um indivíduo do tipo *Actor* ou *Creator*, consoante os seus papeis desempenhados nas séries consideradas, podendo também ser dos dois tipos (caso acumule os cargos de actor e criador, não necessariamente na mesma série).

3.2.2 Caracterização dos Componentes

A caracterização dos componentes apresentados na sub-secção anterior pode ser consultada na figura 9.

Relativamente à representação das séries na ontologia, para esta caracterização, apenas foi considerada a classe hierarquicamente superior (*SeriesGenre*), uma vez que os seus descendentes hierárquicos não possuem qualquer atributo específico, tendo como propósito caracterizar o género das séries. Desta forma evitamos tornar o diagrama da figura 9 excessivamente extenso.

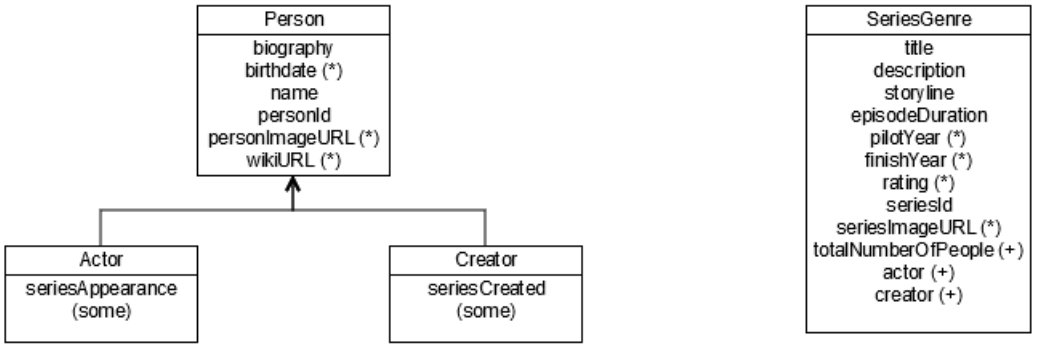

Figura 9: Caracterização dos componentes da ontologia

Assim, para cada pessoa representada na ontologia foi armazenada informação relativamente ao seu nome, identificador único no IMDB, data de nascimento, biografia e *hiperligação* para uma página *web* contendo mais informações sobre a pessoa. Para além destas informações, cada actor armazena ainda as séries em que participa, sendo feito o mesmo com os criadores mas para as séries por eles criadas.

No que respeita às séries, foi armazenada informação relativa ao seu título, descrição, enredo, duração de cada episódio, ano de estreia e de término, identificador único no IMDB, classificação, número total de atores e criadores, logótipo, bem como todos os seus criadores e actores.

Na figura 9, a utilização de * após um atributo indica que cada instância só pode, no máximo, ter um valor para esse atributo. A utilização de + após um atributo indica a obrigatoriedade de pelo menos um valor para o atributo em questão. A utilização de **some** permite um qualquer número de valores para o atributo (incluindo a ausência de valores). Por fim, quando nenhum símbol é utilizado após um atributo apenas é permitido um valor para o referido atributo.

3.2.3 Relações

Na figura 10 podemos ver a forma como as principais classes da ontologia se encontram ligadas, e quais as propriedades associadas a cada ligação.

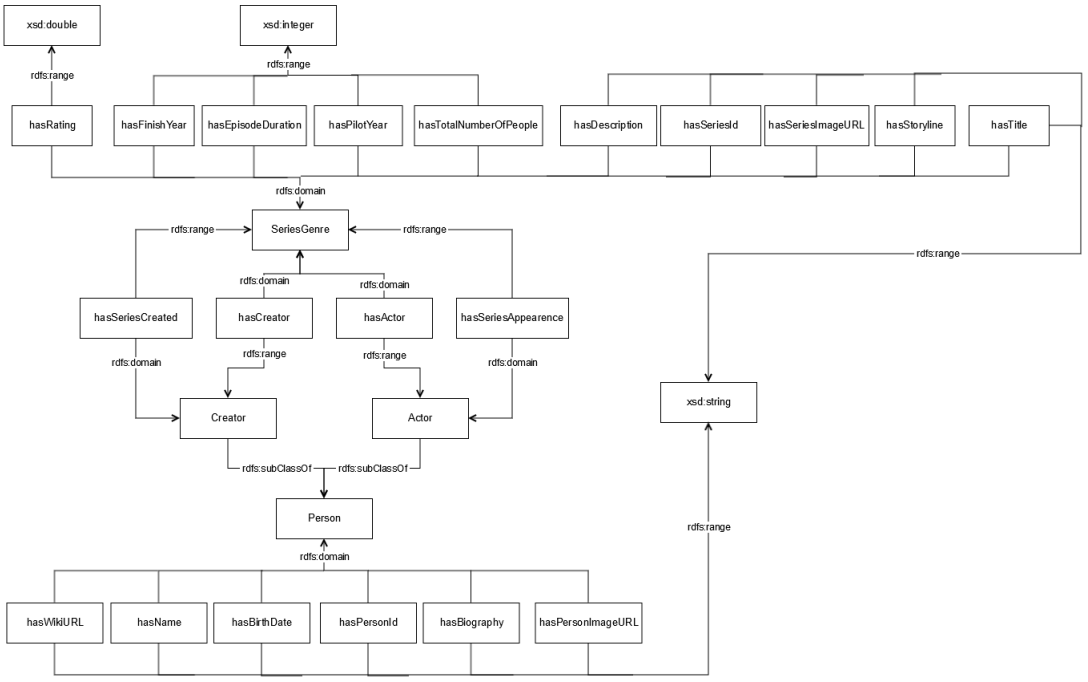

Figura 10: Relações entre as diferentes classes definidas na ontologia

Tal como nas sub-secções anteriores, as sub-classes de *SeriesGenre* não foram incluídas, uma vez que todas estas propriedades são herdadas da classe *SeriesGenre*.

3.2.4 Linguagem

Tendo em conta o referido nas sub-secções anteriores relativamente à ontologia criada, podemos afirmar que esta foi definida utilizando *OWL DL*.

A utilização desta linguagem deve-se à necessidade de permitir a existência de atributos de classes sem número mínimo ou máximo de valores possíveis por instância. Efectivamente, tal situação não pode ser modelada utilizando *RDF* ou *RDFS*, uma vez que estas linguagens não suportam definição de restrições de cardinalidade.

Recorrer a *OWL LITE* também não é suficiente, pois devido à sua limitada noção de cardinalidade os únicos valores que podem ser explicitamente definidos são 0 ou 1. Por fim, embora *OWL Full* permita exprimir a ontologia referida a sua utilização também não se revela adequada, uma vez que adiciona um nível de complexidade bastante superior ao necessário. Para além disso, nenhuma das funcionalidades que esta linguagem acrescenta ao *OWL DL* são necessárias na ontologia criada.

Assim, *OWL DL* surge como a única escolha possível para a criação da ontologia, fornecendo todas as ferramentes necessárias para a modelação dos conceitos e regras pretendidos.

4 Módulos

Na presente secção pretendemos abordar com algum detalhe alguns aspectos relevantes ao nível da implementação dos diferentes componentes do sistema, e que ainda não foram abordados no presente documento.

Desta forma, esta secção começa com a apresentação dos principais algoritmos implementados nos serviços disponibilizados pelo sistema, nomeadamente a pesquisa e recomendação dentro das informações armazenadas na ontologia. De seguida serão tecidas algumas considerações relativamente à utilização das linguagens *SPARQL* e *SWRL* no âmbito do trabalho desenvolvido. Por fim, a presente secção termina com algumas considerações relativamente ao motor de *reasoning* utilizado.

4.1 Algoritmos

4.1.1 Screen Scrapping

A técnica de *screen scrapping* consiste no processo de coletar informação de sites que não expõe a sua API para o público em geral, de forma a disponibilizá-la a outra aplicação. Neste projeto, esta técnica foi utilizada para recolher informação do site IMDB através de um *web crawler*.

Para isso, a aplicação *web crawler* faz uso do *HTML parser JSoup* [6]. Esta biblioteca de Java providencia uma API acessível para extrair e manipular dados de ficheiros HTML através de métodos DOM, seletores CSS e jquery. Os dados são armazenados para objetos em memória.

4.1.2 Browsing

O *browsing* da informação disponível na ontologia é feito através da interação do utilizador com a interface web da aplicação.

Em qualquer página da aplicação encontra-se disponível num painel lateral, à esquerda da página, uma listagem dos géneros de séries disponíveis na ontologia. Selecionando um dos géneros, é apresentado a lista de séries de televisão contidas nesse género. O utilizador pode, então, selecionar uma das séries e consultar a sua informação, que inclui uma listagem de criadores e de atores relativos à série. Selecionando uma dessas pessoas, a navegação é rediricionada para a respetiva página da presenta, que contém a lista de séries que este criou e/ou participou.

[6] `http://jsoup.org/`

4.1.3 Pesquisa

De uma forma resumida, o processo de pesquisa implementado e disponibilizado na interface web consiste na interpretação de uma frase escrita pelo utilizador na qual este indica quais as informações que pretende consultar, extracção dos conteúdos solicitados pelo utilizador e posterior disponibilização dos mesmos na interface. O idioma suportado para esta operação é o inglês.

Para facilitar a interpretação da frase escrita pelo utilizador o sistema possui ainda o seguinte conjunto de palavras reservadas, cada uma associada a um contexto de pesquisa diferente:

- *Termos específicos da ontologia*, nos quais se incluem os termos *series*, *series-genre*, *genre* ou *genres*.
- *Nomes de Géneros presentes na ontologia*.
- *Nomes de classes presentes na ontologia* e seus sinónimos. Este tipo de palavras inclui expressões como *actor*, *actors*, *starring*, *featuring*, *creator*, *creators*, *created*, *person*, *people* e *with*.
- *Nomes de propriedades ou atributos presentes na ontologia* e seus sinónimos, incluindo termos como *score*, *rating*, *born*, *starting*, *since*, *from*, *beginning*, *start*, *pilot*, *started*, *running*, *airing*, *ended*, *finished*, *until*, *end*, *to*, *finale*, *terminated* e *canceled*.
- *Quantificadores*, utilizados para estabelecerem relações . Este tipo de palavras inclui expressões como *equal*, *same*, *lower*, *shorter*, *smaller*, *higher* ou *bigger*.
- *Conetores*, utilizados para interligarem palavras numa frase. Este tipo de palavras reservadas incluem expressões como *and* ou *between*.

Assim, após receber a frase escrita pelo utilizador o sistema analisa-a palavra a palavra, identificando ocorrências das palavras reservadas previamente mencionadas e armazenando num *buffer* todas as palavras que não sejam identificadas como reservadas.

Este *buffer* será posteriormente analisado quando for detectada uma nova palavra reservada, ou quando já não existirem mais palavras para processar. Ao ser processado o conteúdo do *buffer* são consideradas, no máximo, as duas últimas palavras reservadas detectadas.

Por exemplo, se o utilizador escrever *"series comedy simpsons"* o sistema detecta as palavras reservadas *"series"* e *"comedy"* (nome de uma classe da ontologia) e armazena *"simpsons"* no *buffer* para processamento. Ao processar este *buffer*, uma vez que as duas últimas palavras são referentes a séries, o sistema irá em primeiro lugar procurar séries. Mais ainda, uma vez que o utilizador especificou um género esta pesquisa é limitada a todas as séries do género especificado.

Assim, serão procuradas todas as séries de comédia que contenham a expressão *"simpsons"* no seu título. Todos os resultados obtidos são adicionados a uma lista de

resultados para posterior disponibilização na interface web. Após concluir esta pesquisa o sistema procura ainda pessoas na ontologia cujo nome contenha a expressão presente no *buffer* (caso exista mais do que uma expressão o processo repete-se para cada uma).

À semelhança do caso anterior, todos os resultados obtidos são adicionados à lista anteriormente referida, para que sejam fornecidos e disponibilizados na interface web. Para além disso, são também procuradas as séries que contam com a participação dos indivíduos encontrados, sujeitas a eventuais restrições que o utilizador possa ter realizado (como o caso do género já mostrado). Muito embora no exemplo fornecido este último passo seja inútil, a sua utilização permite suportar pesquisas menos organizadas, como por exemplo *series comedy jim parsons*.

De uma forma semelhante ao exemplo descrito, outras combinações de expressões e palavras reservadas podem ser utilizadas para efectuar pesquisas na aplicação.

No entanto, dada a elevada complexidade das tarefas de processamento de linguagens naturais, existe uma vasta gama de situações que não são devidamente suportadas e interpretadas pela aplicação, sendo que o seu comportamento final não se encontra definido (mas que na maioria dos casos o sistema não é capaz de produzir qualquer resultado com base na informação interpretada).

Um destes casos, e que cuja implementação se encontra bastante limitada, envolve a utilização de quantificadores e conectores. Por exemplo, pesquisas envolvendo classificações de séries apenas suportam quantificadores de igualdade, maioridade ou inferioridade (por exemplo *series score bigger 1*), não sendo possivel utilizar a palavra reservada *between* para estas pesquisas.

Pesquisas de atores envolvendo o seu ano de nascimento são ainda mais condicionadas, uma vez que apenas é suportada a utilização da palavra reservada *born* seguida da data de nascimento do ator (no formato dd/mm/aaaa) ou, em alternativa, o seu ano. De notar que estes valores necessitam de corresponder aos valores exactos da data de nascimento do ator. Por exempl para o ator *Jim Parsons*, nascido a 24/03/1973 apenas são suportadas pesquisas na forma *actor jim parsons born 24/03/1973* ou *actor jim parsons born 1973*.

Já nas pesquisas por série envolvendo uma data existe um pouco mais de flexibilidade. O sistem aceita pesquisas por ano de estreia, ano de término, ambas suportando ainda a utilização do conector *between*. De salientar ainda que, nas pesquisas por ano de estreia se especifica o ano a partir do qual a série estreou: *series comedy started 2005*. Da mesma forma, nas pesquisas por ano de término o ano especificado é o ano a partir do qual a série terminou: *series comedy finished 2010*. Utilizando o conector *between*, estes intervalos podem ser restringidos: *series comedy started between 2005 2010*.

Por fim, caso o utilizador pretenda pesquisar séries onde mais do que um actor ou criador participa pode fzê-lo utilizando o conector *and*: *series comedy with jim parsons and johnny galecki*. Nestas pesquisas outros tipos de restrições mais simples, como por exemplo restrições ao nível do género da série, também são suportadas.

4.1.4 Recomendação

De uma forma resumida, o processo de recomendação tem como objectivo analisar o histórico de séries e pessoas consultadas pelo utilizador e sugerir novas séries e pessoas, que o utilizador ainda não consultou, mas que de alguma forma lhe podem interessar, por estarem relacionadas com os items por ele consultados.

Assim, para implementarmos a nossa versão do algoritmo de recomendação, procedemos em primeiro lugar ao registo e armazenamento dos items consultados pelo utilizador. Este registo é persistido através de dois ficheiros de texto, sendo armazenadas as séries e pessoas consultadas pelo utilizador em ficheiros separados, cujo conteúdo é carregado pela aplicação aquando do seu arranque.

O processo de recomendação visa então recomendar 10 items diferentes, dos quais 6 correspondem a novas séries e 4 a novas pessoas (atores ou criadores). Este processo é dividido em 6 fases, que passamos a apresentar:

1. Neste primeiro passo da recomendação são analisados os últimos 20 items consultados pelo utilizador, dos quais se extraem as séries e pessoas consultadas. Esta lista será utilizada nos passos seguintes.

2. A partir da lista de pessoas extraída no primeiro passo são obtidas as séries nas quais essas pessoas participaram, sendo feita uma contabilização da frequência com que cada série é obtida. A partir das séries obtidas mais do que uma vez (portanto, comuns a duas ou mais pessoas) são aleatoriamente seleccionadas até 3 séries que ainda não foram consultadas pelo utilizador para serem adicionadas à lista de séries a recomendar. Caso não seja possível adicionar as 3 séries, é adicionado o número máximo de séries possível, passando o restante para o passo seguinte.

3. A partir da lista de pessoas obtida no primeiro passo são seleccionadas as duas séries com maior classificação que ainda não tenham sido consultadas pelo utilizador para serem adicionadas à lista de séries a recomendar. Caso não seja possível adicionar as 2 séries, é adicionado o número máximo de séries possível, passando o restante para o passo seguinte.

4. A partir da lista de séries extraída no primeiro passo são obtidos os atores e criadores comuns a essas séries e que ainda não foram consultados pelo utilizador, registando a sua frequência. Para as pessoas cuja frequência é superior a 1 (comuns a mais do que uma série) obter aleatoriamente até duas pessoas e adicioná-las à lista de pessoas a recomendar. Das restantes pessoas (isto é, com frequência igual a um) escolher aleatoriamente uma pessoa para ser adicionada à lista de pessoas a recomendar. Caso não existam pelo menos duas pessoas com frequência superior a 1 estas podem ser seleccionadas a partir do conjunto de pessoas com frequência igual a 1. Caso não seja possível adicionar todas as pessoas pretendidas, é adicionado o número máximo de pessoas possível, passando o restante para o passo seguinte.

5. A partir da lista de séries extraída no primeiro passo é feita uma contagem das ocorrências dos diferentes géneros, sendo que para cada um dos dois géneros

mais populares é seleccionada a série com maior classificação que ainda não tenha sido consultada pelo utilizador. Caso não seja possível adicionar as 2 séries, é adicionado o número máximo de séries possível, passando o restante para o passo seguinte.

6. Neste último passo são obtidas as 10 séries mais populares, das quais até 6 são aleatoriamente escolhidas e adicionadas à lista de séries a recomendar. Das restantes são escolhidos aleatoriamente até 4 pesoas para serem adicionadosà lista de pessoas a recomendar.

4.2 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) é uma linguagem de consulta[7] bastante utilizada em pesquisas semântias. O *SPARQL* permite a formulação de *queries* declarativas semânticas que visam extrair e manipular dados armazenados no formato *RDF*, como por exemplo ontologias.

No presente projecto, recorremos à utilização de *SPARQL* em todas as acções que envolviam obter informação previamente armazenada na ontologia, como por exemplo para listar as informações de uma série ou de um ator aquando da sua consulta na interface web.

Assim, em todos os serviços implementados e disponibilizados pelo servidor *REST* foram criadas e executadas *queries SPARQL* para, com base no serviço em questão e na informação fornecida pelo utilizador, extrair a informação por ele pretendida de forma a que esta fosse posteriormente disponibilizada na interface web para sua consulta.

4.3 SWRL

SWRL (Semantic Web Rule Language) é uma linguagem para a *Web Semântica* que pode ser utilizada para expressar lógica e as suas regras. Uma regra lógica consiste numa implicação entre um antecedente e um consequente, sendo que ambos consistem num conjunto (possivelmente vazio) de átomos.

A utilização destas regras pode ser vista como uma forma de expressar novas informações e propriedades que podem ser inferidas a partir das informações já recolhidas e armazenadas. Estas regras permitem também exprimir condições e relações não suportadas pelas linguagens utilizadas no desenvolvimento de ontologias, como por exemplo o *OWL*.

Um dos exemplos mais comum de utilização deste tipo de regras envolve a identificação de relações de parentesco entre dois ou mais indivíduos, como por exemplo, a identificação de uma relação *tio-sobrinho*: O tio de um dado indivíduo é o irmão de um dos pais desse indivíduo. O problema desta relação, e que a torna impossível

[7] Ou utilizando o termo inglês, *query language*

de expressar em *OWL*, prende-se com o facto desta relação só poder ser identificada ao nível dos indivíduos de uma classe (por exemplo, classe Pessoa). Mais ainda, esta relação pode não se verificar para todos os indivíduos da mesma classe (na verdade só se verifica para aqueles cujo antecedente da regra é verdadeiro).

Assim, a utilização de *SWRL* surge como uma vantajosa alternativa para modelar e representar estas situações, que de outra forma não seria possível realizar.

Na aplicação desenvolvida, fruto do tipo de informação armazenada, serviços disponibilizados e ainda de alguma simplicidade abordagem definida, não foram utilizadas regras lógicas envolvendo *SWRL*. De facto, atendendo às propriedades e entidades consideradas a utilização de *SWRL* não permite modelar nenhuma situação ou comportamento que não seja suportado por *OWL*.

4.4 Reasoner

No âmbito deste projecto foi utilizado um motor de *reasoning* por parte do servidor *REST* nas suas queries feitas sobre a ontologia, sempre que a implementação dos diferentes serviços obrigava à obtenção de informações armazenadas na ontologia.

Este motor de reasoning foi usado maioritariamente para inferir ligações de uma pessoa a uma ou mais séries. De acordo com o especificado na secção 3.2, aquando da definição da ontologia, foram criados dois pares de propriedades inversas, que ligam uma série a um actor ou criador. São elas: *hasActor* e *hasSeriesAppearance*, bem como *hasCreator* e *hasSeriesCreated*. Efectivamente, aquando da população da ontologia, em cada um destes pares apenas uma propriedade foi preenchida (*hasActor* e *hasCreator*). Preencher as duas propriedades em cada par revelar-se-ia uma tarefa desnecessária, adicionando redundância à informação armazenada na ontologia.

Assim se quisermos obter, para uma dada série, a sua lista de atores ou criadores podemos simplesmente obter todos os seus *statements* e ver quais é que se referem a propriedades do tipo *hasActor* ou *hasCreator*. No entanto, se quisermos obter a lista de séries criadas por uma pessoa, ou a lista de séries que contam com a participação de um dado ator, não podemos simplesmente procurar *statements* dessa pessoa e procurar ocorrências da propriedade *hasSeriesCreated* ou *hasSeriesAppearance*, pois estas não estão preenchidas.

Ora, dado que *hasSeriesCreated* é uma propriedade inversa de *hasCreator*, se uma dada série está ligada a uma pessoa através da propriedade *hasCreator* então podemos inferir que essa pessoa também está ligada a essa série pela propriedade *hasSeriesCreated*.

Assim, neste projecto, foi utilizado um motor de *reasoning* para fazer este tipo de inferências, que procuram descobrir a existência de propriedades e de outro tipo de relações entre classes, instâncias e até propriedades de uma ontologia, de forma a extrair o máximo de conhecimento possível das mesmas.

Caso não fossem utilizados estes motores de inferência e não se fosse mantida redundância na informação armazenada, o processo de obtenção as séries criadas por

um indivíduo (ou nas quais este participa) seria um processo bastante penouso e ineficiente, envolvendo analisar todos os atores ou criadores de todas as séries armazenadas na ontologia.

Por fim, a utilização de *reasoner* permite ainda detectar inconsistências e erros na definição da ontologia de uma forma automática, o que é, obviamente, bastante vantajoso.

5 Estrutura do Código

Na presente secção pretende-se apresentar de uma forma sucinta e estruturada o código produzido durante o processo de desenvolvimento do sistema. Esta secção não pretende ser um manual de programador para o sistema desenvolvido, mas sim oferecer uma visão geral e de alto-nível sobre o código produzido, tendo como foco principal a sua estrutura e organização.

Assim, na presente secção serão abordados os componentes previamente apresentados aquando da especificação da arquitectura do sistema, na secção 2.

5.1 População Ontologia

No sistema desenvolvido, a população da ontologia é realizada pela aplicação de *web crawling*, após esta proceder à recolha dos dados relativos a cada série a ser armazenada na ontologia, a partir do site do IMDB.

Relativamente à recolha de dados o código desenvolvimento encontra-se dividido em dois projectos (e portanto, duas aplicações) independentes e desenvolvidos em linguagens de programação diferentes:

O primeiro consiste na aplicação presente na pasta *IMDBHandler*, fornecida em conjunto com o presente relatório. Esta é uma simples aplicação desenvolvida em *Python*, que carrega um conjunto de séries presentes num ficheiro de texto e, utilizando a biblioteca *IMDbPY*[8], obtém os identificadores únicos de cada série no site IMDB, persistindo os mesmos num ficheiro *CSV*.

O segundo projecto desenvolvido, presente na pasta *TVSeries*, consiste numa aplicação desenvolvida em *Java* que carrega a lista de séries e respetivos identificadores a partir do ficheiro *CSV* produzido pela aplicação anterior, analisando as páginas do IMDB relativas a cada série que se encontra no ficheiro, extraindo delas as informações pretendidas e persistindo-as na ontologia criada. O código desenvolvido para este projecto encontra-se organizado em três *packages*, de acordo com as funções nele implementadas:

- *IMDBCrawler*. Neste *package* estão implementadas todas as funções necessárias para carregar a lista de séries e identificadores, bem como a implementação das acções de *screen scrapping*.
- *ontology*. Neste *package* apenas se encontra implementada uma classe (*OntologyCreator)*, que tem como objectivo abstrair todas as interacções com a ontologia do resto da aplicação. Tal como referido na secção 3.2, todas as interacções com a ontologia são realizadas utilizando a *framework Apache Jena*. Assim, é nessa classe que são implementados todos os métodos utilizados na população da ontologia, dos quais se destacam a criação de séries e pessoas

[8] `http://imdbpy.sourceforge.net/`

(sejam atores ou criadores), ligação entre uma série e uma dada pessoa (novamente, seja ator ou criador dessa série) e ainda a atribuição de hiperligações para páginas com informação relativa a uma dada pessoa.

- *data*. Neste *package* encontram-se implementadas algumas estruturas de dados que pretendem representar algumas classes criadas na ontologia, como é o caso da classe *Pessoa* ou *Série*, bem como outras estruturas de dados úteis, tanto para o processo de *screen scrapping* como para a população da ontologia.

5.2 Interface Web

A interface web da aplicação constitui uma *Java EE Application*, que utiliza o servidor *open-source Glassfish*. O código da aplicação encontra-se disponibilizado na pasta *TVSeries_Website*.

O conteúdo web da aplicação está contido na pasta *web*. O projeto faz uso de páginas JavaServer (JSP) e encapsula conteúdo reutilizável em ficheiros *Tag*. O ficheiro *Tag* pode ser encontrado na pasta *tags* no interior da pasta *WEB-INF*. Adicionalmente a pasta *web* contém uma pasta *bootstrap* contendo os ficheiros CSS de costumização das páginas web.

A pasta *source* contém uma única *package*, designada de *actions*. Nesta, estão contidos os *Java Servlets* utilizados na aplicação.

5.3 Servidor REST

Na interface web disponibilizada, todo o conteúdo e funcionalidades implementados são conseguidos através da invocação de serviços de um servidor *REST* encarregue de gerir todas as interacções com a ontologia e dela extrair a informação que se pretende disponibilizar aos utilizadores, na interface web. Tal como previamente referido, este servidor foi implementado utilizando a *framework Spring*, sendo todas as interacções com a ontologia realizadas através da *framework Apache Jena*.

Relativamente à implementação deste servidor o código produzido foi dividido em duas *packages* principais. São elas:

- *ontology*. Nesta *package* encontra-se implementada a classe *OntologyHandler*, onde são implementadas todas as interacções com a ontologia. Esta *package* possuí ainda uma *sub-package* denominada *data*, onde se encontram implementações de uma série de estruturas de dados que pretendem representar algumas classes criadas na ontologia, bem como outras estruturas de dados úteis para as operações implementadas.

- *server*. Nesta *package* encontra-se toda a implementação do servidor *REST* bem como de classes e estruturas de dados necessárias para suportar os serviços implementados.

A *package server* foi ainda dividida nas seguintes *sub-packages*:

- *application*. Nesta *sub-package* encontram-se classes específicas de configuração do servidor, incluindo o seu ponto de partida, o controlador onde são declarados os serviços por ele disponibilizados e a configuração dos filtros utilizados (que no caso da aplicação em questão se resumem a um filtro de *Cross Origin Requests*).
- *services*. Nesta *sub-package* encontram-se implementadas uma série de classes utilizadas para representar a informação retornada pelo servidor em alguns serviços disponibilizados.
- *utils*. Nesta *sub-package* encontram-se classes e estruturas de dados de suporte aos serviços disponibilizados pelo servidor.

6 Conclusão e Trabalho Futuro

Neste relatório, apresentamos o projeto de construção de um sistema de Web Semântica que disponibiliza informação de séries de televisão através de um *website*, oferecendo serviços de *browsing*, pesquisa e recomendação sobre a ontologia.

O sistema desenvolvido consiste na integração de diferentes aplicações: gestão de conteúdos, ontologia para armazenamento de dados, servidor REST e interface gráfica. Estes componentes estão dispostos numa arquitetura MVP em que o sistema de gestão de conteúdos e a ontologia constituem o modelo de dados, o servidor REST é reponsável pela camada de apresentação e a interface gráfica constitui a camada de visualização do sistema.

Este projeto cumpre todos os restrições impostas: a ontologia é representada em linguagem de Web Semântica, *OWL DL*; é utilizada uma *triple store*, *Jena*, para armazenar a ontologia; a pesquisa sobre a ontologia é feita utilizando SPARQL; o sistema apresenta uma interface web para interação com o utilizado. Para construir a ontologia foi necessário desenvolver um *web crawler* que recolhe informação sobre séries de televisão do site IMDB através da técnica de *screen scrapping*.

O sistema oferece com sucesso as seguintes funcionalidades: *browsing* da informação através da navegação do *website*; pesquisa semântica sobre a informação contida na ontologia; recomendação de páginas ainda não visualizadas pelo utilizador com base no histórico de pesquisa do utilizador.

Não obstante, o sistema desenvolvido ainda possui alguns aspetos que podem ser trabalhados e melhorados, a realizar em iterações futuras do presente trabalho. Para além de eventuais erros que possam não ter sido detectados nos testes realizados ao sistema, entedemos que o processamento das frases introduzidas pelos utilizadores na realização de pesquisas pode ser melhorado, concedendo mais liberdade ao utilizador para construir as suas pesquisas.

Para além disso consideramos também necessário alterar o mecanismo de recomendação de modo a que o seu tempo de execução seja mais reduzido. Naturalmente que a utilização de um *reasoner* contribuiu para esta situação, no entanto consideramos que a definição da ontologia criada também contribuiu para esta demora. Assim, em iterações futuras deste trabalho, gostaríamos também de melhorar o esquema da ontologia a utilizar, de forma a reduzirmos o tempo de execução das *queries* a ela realizadas.

Também seria interessante expandir a população de séries armazenadas na ontologia, uma vez que o número médio de séries ao qual cada pessoa está associada na ontologiaé bastante reduzido. De facto, consideramos importante para este tipo de aplicações poder associar a cada pessoa vários trabalhos nos quais esta tenha participado e nesse ponto o sistema desenvolvido poderia ser mais rico. Nesse sentido, também consideramos a possibilidade de extender o sistema desenvolvido para o universo dos filmes, armazenando e disponibilizando informações tanto sobre séries de televisão, como sobre filmes.

Por fim, dado que o número de instâncias armazenadas na ontologia é consideravelmente elevado (nomeadamente o número de pessoas armazenadas) também achamos interessante explorar a utilização de indíces de forma a reduzir o tempo de execução das *queries* realizadas à ontologia.