Plataforma Richard Burton: utilizando Reflexão Computacional para criar filtros genéricos de dados

Andrés Vidal Berriel[1], Silvia de Castro Bertagnolli[1], Cimara Valim de Melo[1]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS) – Câmpus Canoas
CEP: 92412-240 – Canoas – RS – Brasil

andresvidal19976@gmail.com, silvia.bertagnolli@canoas.ifrs.edu.br, cimara.melo@canoas.ifrs.edu.br

Abstract. *This paper describes the study focused on the creation of generic data filters. Those tools constitute a solution to the necessity of filtering information stored in the Richard Burton Platform, according to variable search criteria. Thus, this study intends to present the concepts and elements regarding a searching process and discuss the (computational) reflection techniques adopted on the development of that solution. The main result of this research is the creation of the Richard Burton Platform's data filtering mechanism.*

Resumo. *Este trabalho descreve o estudo envolvido na criação de filtros genéricos de dados. Essas ferramentas configuram uma solução à necessidade de filtrar informações acumuladas pela Plataforma Richard Burton, de acordo a critérios de pesquisa variáveis. Este estudo tem como objetos-foco os conceitos e elementos envolvidos durante uma pesquisa, bem como as técnicas de Reflexão Computacional adotadas no desenvolvimento dessa solução. O resultado principal deste trabalho é a criação do mecanismo de filtragem de dados da Plataforma Richard Burton.*

1. Introdução

A Reflexão Computacional (RC) é um recurso fornecido por algumas linguagens reflexivas para obter informações sobre a estrutura (classe) e o estado de um objeto. Conforme Guerra (2014), dentre as possibilidades das técnicas de RC estão (i) o desenvolvimento de programas com alto nível de abstração e modularidade; (ii) a criação de frameworks dinâmicos; e (iii) a injeção de dependências em frameworks.

O problema abordado neste trabalho compreende a necessidade de criar mecanismos dinâmicos de filtragem de dados, a qual se enquadra no caso da Plataforma Richard Burton, apresentada na próxima seção. De modo a atender essa necessidade, desenvolveu-se uma estrutura de classes e métodos capazes de analisar os elementos da plataforma e, dinamicamente, filtrar os dados constantes num determinado modelo de dados (neste caso, uma lista). A construção desses filtros leva em consideração a complexidade dos relacionamentos entre os objetos que constituem esse sistema.

2. A Plataforma Richard Burton

Este trabalho está vinculado ao projeto de pesquisa "Literatura brasileira e transnacionalidades: deslocamentos, identidades e experimentações tecnológicas"[1] que possui, entre os seus objetivos, o desenvolvimento de um repositório *web,* denominado Richard Burton, de dados sobre literatura brasileira traduzida. Essa solução é desenvolvida com o objetivo de publicar esses dados realizando um controle rígido de consistência, de modo a constituir uma ferramenta de fácil uso e alta fiabilidade para pesquisadores da área de Literatura. A motivação para o desenvolvimento dessa plataforma surge da dificuldade do acesso a dados sobre obras literárias brasileiras traduzidas.

Conforme as pesquisas de Barbosa (1994), o sistema literário brasileiro encontrou-se, historicamente, numa situação de despreocupação com a universalização de seu conteúdo. Entretanto, nas últimas décadas, escritores nacionais aumentaram sua visibilidade, estimulando o interesse internacional em traduzir obras de literatura brasileira [Moser, 2011; Gurria-Quintana, 2014]. Isso decorreu na intensificação das pesquisas relacionadas a esses processos tradutórios, que esbarram, frequentemente, na falta de acesso a dados sobre o histórico da tradução de literatura brasileira – fato reportado pela Fundação Biblioteca Nacional (informação verbal)[2] e suportado pelo escasso número de publicações científicas que levantem esses dados.

Partindo para a perspectiva técnica, a plataforma Richard Burton adota o paradigma Orientado a Objetos e é desenvolvida na linguagem Java, utilizando as tecnologias Java Server Faces (JSF) e Java Persistence API (*Application Programming Interface*) (JPA), cuja implementação é o *framework* ORM (*Object-Relational Mapping*) Hibernate. Naturalmente, é adotado o *design-pattern* MVC (*Model-View-Controller*) para organizar as camadas da aplicação.

Os objetos que comportam a Plataforma podem ser classificados de acordo com seu papel em uma pesquisa. Por um lado, existem os objetos passíveis de serem pesquisados – pesquisáveis, como as Obras Traduzidas –, isto é, aqueles que constituem grandes conjuntos de informações úteis para o usuário. Por outro lado, estão os objetos passíveis de serem critérios de pesquisa, isto é, que representam informações referenciais para a seleção de objetos (pesquisáveis), como o ano da obra.

Os relacionamentos entre as várias entidades da plataforma constituem agregações, composições e associações bilaterais e de múltiplas cardinalidades – principalmente do tipo muitos para muitos (n:n) –, o que deriva na utilização recorrente de modelos de dados, como as listas. Esse fato resulta na grande variabilidade dos caminhos dos critérios de pesquisa (abordados na seguinte seção) e dificulta a realização de pesquisas genéricas entre os dados da Plataforma.

---

[1] Site do projeto disponível para acesso em: http://frombraziltotheworld.canoas.ifrs.edu.br/

[2] Colóquio *Brazilian Literature: Challenges for Translation, King's College London* (K2.31 *Nash Lecture Theatre Kings Building Strand Campus*), Londres, Reino Unido. 18 Ago. 2015. Informação retirada da apresentação de Salgado: "*The Brazilian government and the presence of works of literature from Brazil in the international publishing market.*" Mais informações em: <http://www.kcl.ac.uk/artshums/depts/splas/eventrecords/2013-14/Brazilianlit.aspx> (programação acessível no *link*)

3. Filtros Genéricos de Informação

Neste trabalho, entende-se como "filtro" qualquer método que execute um processo de filtragem sobre determinado modelo de dados – como uma lista. Esse tipo de processo tem a função de selecionar, dentre esse modelo, aqueles objetos que sejam relevantes de acordo a um critério de pesquisa, ao qual é atribuído um valor, e um tipo de filtragem determinados.

O conceito de critério de pesquisa remete à característica de um determinado objeto, pela qual este é pesquisado. Alguns exemplos podem ser o ano de publicação e o título de uma determinada obra traduzida. Esse critério é alcançado através de um caminho de objetos e propriedades, através do qual pode-se navegar. Nos filtros desenvolvidos neste trabalho, esse caminho é estruturado como ponteiros, conforme o exemplo "*obraTraduzida.tradutores.nome*", que define o critério "nome de algum dos tradutores de uma obra traduzida". O tamanho desse caminho define a profundidade do critério.

Cada critério de pesquisa possui um valor de pesquisa, com o qual deverá coincidir dentro de cada objeto selecionado pelo filtro. São exemplos de valores de pesquisa "1997" (para o ano de publicação) e "*The Heritage of Quincas Borba*" (para o título). A coincidência entre o valor e o critério de pesquisa verifica-se através de um tipo de filtragem, isto é, um subprocesso que define a forma em que o filtro será aplicado. O tipo de filtragem é a ação que relaciona o valor de pesquisa ao critério de pesquisa. Alguns exemplos, para uma pesquisa de critério "a" e valor "b", seriam os seguintes: "a contêm b"; "a igual a b"; "a maior que b"; entre outros.

Quanto ao filtro desenvolvido para a Plataforma Richard Burton, estabeleceu-se, como principal requisito, a sua generalidade, isto é, a capacidade de utilizar o mesmo método em qualquer situação provável dentro do seu contexto de uso. As situações prováveis definiram-se de modo a reduzir o escopo e agilizar o desenvolvimento. Um exemplo de situação provável é o usuário pesquisar uma obra traduzida tendo como critério/valor um nome constante na lista de autores da sua obra original – *obraTraduzida.obraOriginal.autores.nome*. Em contraponto, é uma situação improvável que o usuário pesquise obras traduzidas através do título de uma outra obra traduzida por algum de seus tradutores – *obraTraduzida.tradutores.traduções.título*.

De modo a atender essa generalidade, adotaram-se técnicas de RC, utilizando-se introspecção e outros recursos disponibilizados pela API de reflexão da linguagem Java [Guerra, 2014]. As técnicas adotadas neste trabalho foram implementadas, principalmente, com vistas à construção dinâmica do caminho do critério de pesquisa. Para isso, encadeiam-se os vários métodos *getters* dos atributos para chegar, desde o objeto inicial, até o atributo indicado pelo critério de pesquisa. Complementarmente, a RC é utilizada para chamar os métodos dos tipos de filtragem, de modo a manter o código otimizado.

Considerando o conceito de "grau" do filtro como o número de critérios de pesquisa aplicados a um filtro – por exemplo, "autor da obra original", de grau 1, e "autor da obra original e data de publicação", de grau 2, criou-se uma projeção do número de métodos que deveriam ser implementados em função do número de critérios de pesquisa disponíveis. Essa estimativa é calculada através da quantidade mínimo de algoritmos estáticos necessários para percorrer o caminho do critério de pesquisa, utilizando como

técnica o encadeamento de *getters*. No gráfico ilustrado na Figura 1, é possível observar as vantagens da solução com RC, uma vez que esta permite implementar um único método de filtro para qualquer número de critérios e para qualquer grau de filtro.

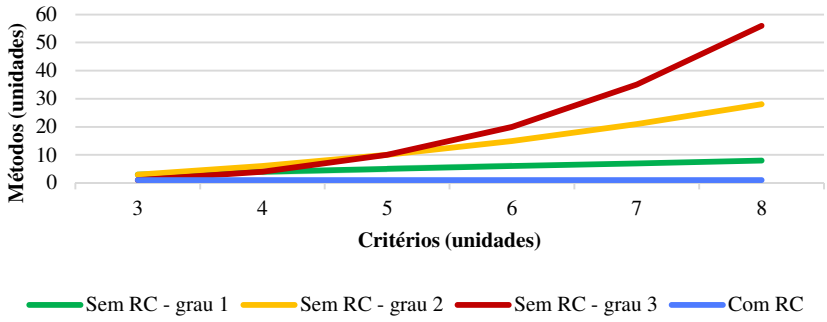

Figura 1. Projeção do número de métodos por critério de pesquisa entre graus de filtro

4. Conclusão

Como resultado principal deste trabalho está o desenvolvimento de um mecanismo que, ao receber como entrada um critério de pesquisa normalizado, um valor de pesquisa, uma lista e um tipo de filtragem, retorna a lista contendo os elementos filtrados. Para tanto, utiliza a geração dinâmica de filtros de forma otimizada, utilizando RC. Com o auxílio da API de reflexão da linguagem Java, foram criados recursos computacionais capazes de inferir tipos de filtragem a partir do critério e do valor de pesquisa inseridos – permitindo a utilização de valores de pesquisa interpretáveis pelo sistema, tais como os operadores lógicos para valores numéricos e as versões textuais dos mesmos, por exemplo, “maior que” em vez de “>”.

Como etapas futuras deste trabalho, pretende-se adicionar o aninhamento de listas; otimizar a inferência de tipos de filtragem; e ampliar e otimizar o processo de interpretação de valores de pesquisa estendidos. A partir deste trabalho, pode-se afirmar que a API de RC da linguagem Java permite explorar os detalhes do objeto sem conhecê-lo previamente, abrindo uma gama de possibilidades úteis para a criação de soluções de software mais otimizadas.

References

Barbosa, H. G. (1994) “The Virtual Image: Brazilian Literature in English Translation”. Centre for British and Comparative Cultural Studies: University of Warwick. http://wrap.warwick.ac.uk/56829/1/WRAP_thesis_Barbosa, Heloisa Goncalves.pdf

Gurría-Quintana, A. (2014) “Brazil does books as well as football”. The Guardian. http://www.theguardian.com/books/booksblog/2014/jun/05/brazil-does-books-as-well-as-football-world-cup

Moser, B. (2011) “Brazil’s Clarice Lispector Gets a Second Chance in English”. Publishing Perspectives. http://publishingperspectives.com/2011/12/brazil-claire-lispector-second-chance-in-english.

Guerra, E. Componentes Reutilizáveis em Java com Reflexão e Anotações. São Paulo: Casa do Código, 2014.