https://eventos.utfpr.edu.br//sicite/sicite2018

Refinamento do modelo FAQLP

Refinement of the model FAQLP

Andressa Oliveira Souza
andressasouza@alunos.utfpr.edu.br
Universidade Tecnológica Federal do Paraná, Ponta Grossa, Paraná, Brasil
Simone Nasser Matos
snasser@utfpr.edu.br
Universidade Tecnológica Federal do Paraná, Ponta Grossa, Paraná, Brasil

RESUMO

O FAQLP é um aplicativo voltado para perguntas e respostas do FrameMK (*Framework* de domínio para formação de preço de venda) que está sendo desenvolvido pelo Grupo de Pesquisa em Sistema de Informação no câmpus Ponta Grossa da Universidade Tecnológica Federal do Paraná desde 2008. O objetivo deste trabalho é o refinamento do modelo da arquitetura do FAQLP tornando-o mais flexível e manutível. Para isto, o uso de Padrões de Projeto apoiado por uma abordagem adaptada foi utilizado.
PALAVRAS-CHAVE: Refinamento. Padrões de Projeto. FAQLP.

ABSTRACT

The FAQLP is an question-and-answer application of FrameMK (Domain framework for sales price formatio) wich has been developed by the Information System Research group at Ponta Grossa campus in the Technological Federal University of Paraná since 2008. The Objective of this work is the refinement of the architecture model of the FAQLP by making it more flexible and maintainable. For this, the ose of Design Patterns supported by an adapted approach was used.
KEYWORDS: Refinement. Design patterns. FAQLP.

Recebido: 31 ago. 2019.
Aprovado: 04 out. 2018

INTRODUÇÃO

A refatoração de um sistema pode ser aplicada de forma independente ou com o auxílio de ferramentas já existentes. Para isto, um desenvolvedor ou pesquisador deve possuir conhecimento e experiências das técnicas existentes na literatura para o processo de refatoração ou refinamento de um projeto.

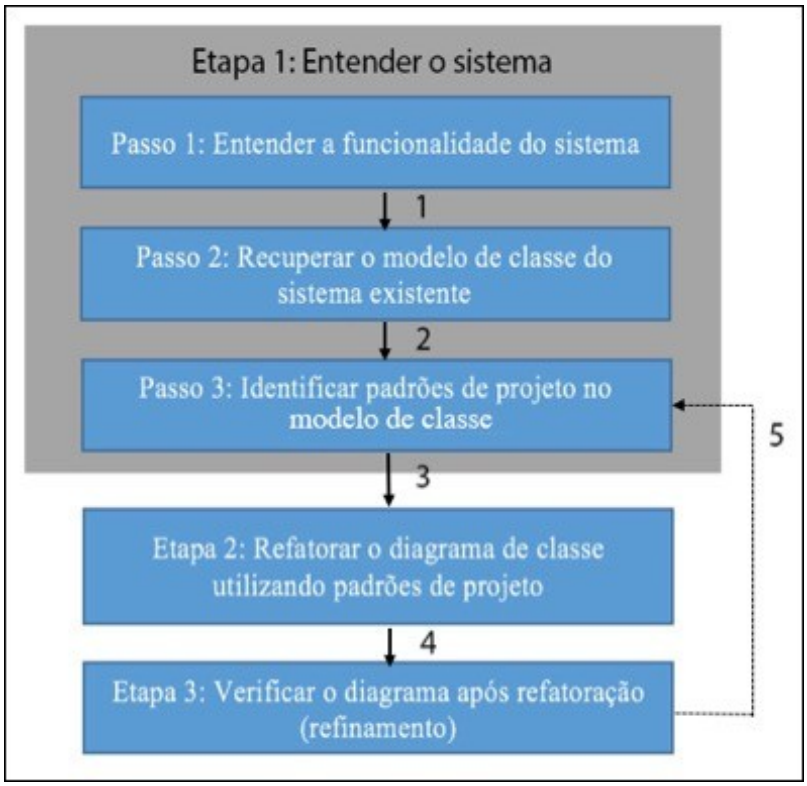
O projeto que será refinado é o FAQLP do FrameMK (*Framework* para Formação de Preço de Venda), que se trata de um *framework* de domínio que tem o objetivo de criar um modelo arquitetural para o domínio na área de formação de preço de venda, por meio de métodos já estabelecidos, identificando aspectos comuns e específicos para sua modelagem de domínio (MENDES, 2015).

O FAQLP se trata de um aplicativo de perguntas e respostas do FrameMK e foi criado por Mendes (2015) após observar o funcionamento do *framework* e constatar que o mesmo não possuía nenhum módulo responsável por ajudar usuários em caso de dúvidas na utilização do aplicativo. Este módulo faltante é denominado FAQ. Com isto, esta pesquisa utilizou os conceitos de refatoração para sugerir um refinamento no modelo do FAQLP.

METODOLOGIA UTILIZADA PARA REFATORAÇÃO

Para a realização deste trabalho foi utilizado a abordagem de Rapeli (2006) com algumas alterações para melhor aproveitamento. A abordagem adaptada de Rapeli é apresentada na Figura 1.

Figura 1 – Modelo Adaptado de Rapeli



As etapas e passos da abordagem adaptada estão descritas a seguir:

1. Entender o Sistema: esta etapa é composta por três passos.
    - Passo 1: Entender a funcionalidade do sistema: Este é o passo inicial do método e consiste em analisar o sistema e suas funcionalidades afim de entender o mesmo.
    - Passo 2: Recuperar o modelo de classe do sistema existente: Neste passo é realizada a recuperação do modelo de diagrama de classes com o auxílio do código fonte e materiais disponíveis do projeto.
    - Passo 3: Identificar padrões de projeto no código-fonte do sistema: Com base nas informações obtidas pela execução dos passos anteriores, torna-se possível a identificação de padrões de projetos que podem ser aplicados no projeto e assim identificar possíveis soluções de refinamento.
2. Refatorar o diagrama de classe utilizado padrões de projeto: Com o diagrama de classe recuperado (ou analisado) na etapa anterior, esta etapa é responsável pela refatoração do diagrama de classes utilizando padrões de projeto que possam ser aplicados no modelo.
3. Verificar o diagrama após refatoração (refinamento): Esta fase é responsável por verificar se os padrões inseridos no diagrama de classe condizem com a aplicação e se ainda existem alterações que possam ser realizadas no modelo.

A diferença deste modelo adaptado com a abordagem de Rapeli (2006) está na refatoração através do uso do diagrama de classe ao invés de linha de código. A visualização de quais padrões de projeto podem ser inseridos é dado pelo diagrama de classe recuperado com o auxílio do código-fonte na documentação da aplicação. Esta adaptação foi realizada pois este projeto não visa a refatoração por linha de código, apenas no modelo (diagrama de classe).

PADRÕES DE PROJETO

Segunda Gamma *et al* (2000) e Freeman *et al* (2009), os padrões de projeto podem ser descritos com uma solução para um problema dentro de um determinado contexto quando o assunto é projeto de software. Um padrão de projeto se trata de um modelo de como resolver um determinado problema. No geral, um padrão possui quatro elementos essenciais:

- Nome do padrão: Normalmente é uma referência que pode ser utilizada para descrever um problema de projeto, suas soluções e consequências em uma palavra.
- Problema: Descreve em que situação aplicar um determinado padrão. Explica o problema juntamente com seu contexto, podendo ser problemas específicos tal como representar algoritmos como objetos.
- Solução: Responsável por descrever os elementos que compõem o padrão de projeto, seus relacionamentos, suas responsabilidades e

colaborações. Basicamente se trata de um “gabarito” que pode ser aplicado em diversas situações diferentes e não uma implementação ou um projeto concreto.

- Consequências: Trata-se dos resultados e análises de vantagens e desvantagens da aplicação de um padrão.

Os padrões são classificados por dois critérios. Sua finalidade e seu escopo. Sobre as finalidades os padrões refletem o que um padrão faz, podendo ser do tipo criacionais, estruturais e comportamentais. Os padrões do tipo criacionais são responsáveis por abstrair o processo de criação de objetos, ou seja, sua intanciação, estruturais são responsáveis por lidar com a composição de classes ou de objetos, enquanto que os padrões comportamentais descrevem maneiras pelas quais as classes ou objetos interagem e distribuem responsabilidades.

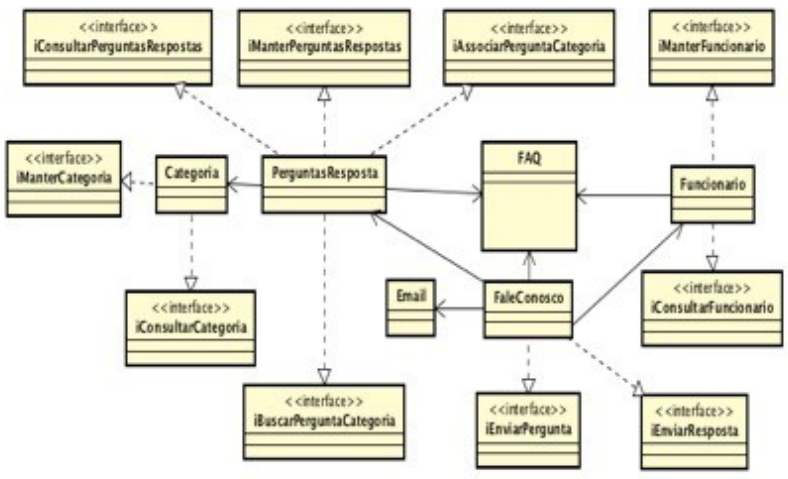
RESULTADOS E DISCUSSÕES

De acordo com o Método Adaptado de Rapeli criado, o primeiro passo do projeto consistia no entendimento do sistema como um todo. Para isto, foi realizado um estudo sobre o Trabalho de Conclusão de Curso de Mendes (2015). A documentação do projeto possibilitou o entendimento do mesmo enquanto que os diagramas dispostos no trabalho deram uma visão mais aprofundada de como o sistema foi modelado.

O estudo de exemplos disponíveis na literatura segundo Mendes (2015), auxiliou na geração das regras de negócio, permissões, funcionalidades do sistema. Através da descrição dos casos de uso do sistema tornou-se possível a compreensão dos cenários possíveis dentro do sistema.

No passo 2, responsável pela recuperação do modelo de classe do sistema, os diagramas de conceito de negócio das similaridades e de componentes das similaridades foram utilizados como base para a abstração de um modelo de classe geral da aplicação. O resultado obtido através da análise dos diagramas e documentação do FAQLP é apresentada na Figura 2.

Figura 2 – Diagrama de Classe Geral FAQLP



Após a abstração do diagrama de classe geral do FAQLP a pesquisa prosseguiu para o passo 3, responsável por identificar padrões de projeto já existentes no sistema. Após análise, os seguintes padrões *SessionFactory* e *DAO (Data Access Object)* foram encontrados.

Ao fim dos passos da Etapa 1 a pesquisa prosseguiu para a Etapa 2, responsável por refatorar o diagrama de classe retornando a utilização de Padrões de Projetos para o refinamento do modelo FAQLP. Onde os seguintes padrões de projeto foram sugeridos para a aplicação:

- Proxy: Este padrão possui o objetivo de fornecer um substituo ou marcador da localização de outro objeto para controlar o acesso a este objeto. Este padrão também e conhecido como *Surrogate*. (GAMMA et al., 2000).
- Decorator: Responsável por atribuir responsabilidades adicionais a um objeto dinamicamente. Este padrão fornece uma alternativa flexível a subclasses para extensão da funcionalidade (GAMMA et al., 2000).
- Observer: Define uma dependência de um-para-muitos entre objetos, de modo que, quando um objeto muda de estado, todos os seus dependentes são automaticamente notificados e atualizados. (GAMMA et al., 2000).

O padrão *Proxy* pode ser implementado com o modelo *CGLIB Proxy* nas classes de gerenciamento como a classe *Funcionary* por exemplo. Como a aplicação do FAQLP utilizou o *Hibernate* para realizar o mapeamento objeto-relacional e que por padrão utiliza o mapeamento do tipo *LazyLoading*, o *Proxy* pode ser utilizado para a criação de uma classe dinâmica quando o método *get* for invocado.

O padrão *Decorator* é sugerido pela existência de fatores comuns entre as interfaces gráficas utilizadas para o controle do FAQLP para padronizar as interfaces de usuário e administrador. O padrão *Observer* pode ser utilizado para melhor controle das perguntas e respostas enviadas por usuários e administradores da aplicação. Uma das maiores motivações para sugerir o uso do *Observer* é o efeito do particionamento de um sistema em uma coleção de classes operantes com a necessidade de manter a consistência.

Ainda sobre refatoração do modelo FAQLP sugere-se o uso de ferramentas para a detecção de "*Bad Smells*" no código fonte do sistema em trabalhos futuros.

Com esta pesquisa foi possível visualizar o quanto o uso de padrões de projeto pode implicar no aumento da reusabilidade, criação e entendimento dos modelos.

Os conceitos de padrões de projeto estudados e utilizados nesta pesquisa visam formalizar a documentação de experiência do conhecimento, garantindo aos próximos pesquisados e desenvolvedores do FrameMK o que pode se tornar no aumento da produtividade e diminuição do grau de complexidade do sistema.

REFERÊNCIAS

FREEMAN, E.; FREEMAN, E.; SIERRA, K.; BATES, B. **Use a Cabeça! Padrões de Projetos.** Rio de Janeiro: Alta Books, 2009.

GAMMA, E.; HELM, R.; JOHNSON R.; VLISSIDES, J. **Padrões de Projeto: Soluções reutilizáveis de software orientado a objetos.** São Paulo: Bookman, 2000.

MENDES, Renan R. **Linhas de Produto de Software: Um estudo de caso para o desenvolvimento de um sistema de FAQ**. 2015. 131 f. Trabalho de Conclusão de Curso (Graduação) – Bacharelado em Ciência da Computação em Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2015.

RAPELI, Leide R. **Refatoração de Sistema Java Utilizando Padrões de Projeto: Um Estudo de Caso.** 2006. 127 f. Dissertação (Mestrado) em Universidade Federal de São Carlos. São Paulo, 2006.

AGRADECIMENTOS

Agradecimento especial à Fundação Araucária pela bolsa concedida para a realização deste projeto.