Refatoração de Aplicações Web: Um Estudo de Caso

Jean Carlos Dalcero, Bruno Batista Boniati

Tecnologia em Sistemas para Internet – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 54 – 98.400-000 – Frederico Westphalen – RS

{jeandalcero, brunoboniati}@gmail.com

Resumo. *Constantemente os sistemas de software estão em evolução, há quem diga que o código de uma aplicação é orgânico, ou seja, se deteriora naturalmente com o passar do tempo em função de novas técnicas e ferramentas. Refatoração é uma técnica que consiste em alterar a estrutura do código de uma aplicação sem que isso afete necessariamente seu funcionamento, trata-se de melhorias internas. Por meio deste trabalho objetiva-se estudar técnicas de refatoração para aplicações web. Através de um estudo de caso pretende-se identificar falhas relacionadas à segurança, leiaute, acessibilidade e desempenho de uma aplicação, realizando a reestruturação do código com técnicas de Refatoração de Aplicações Web.*

Abstract. *Constantly software systems are evolving, some say that the code of an application is organic, in other words, it naturally deteriorates over time due to new techniques and tools. Refactoring is a technique that consists in changing the structure of an application without necessarily affecting its functioning, just like internal improvements. This paper aims to study refactoring techniques for web applications. Through a case study it is intended to identify failures related to security, layout, accessibility and performance of an application, performing the restructuring of the code using refactoring techniques for Web Applications.*

1. Introdução

Com a evolução da Internet, a importância e demanda das aplicações Web aumentou muito, em especial nos últimos anos. Os sistemas defasados ou não planejados podem possuir inúmeras falhas e limitações. Porém essas características podem ser melhoradas através de técnicas e normas padronizadas que surgiram para o melhoramento dessas aplicações, fazendo com que os sistemas possam ser desenvolvidos com facilidade no entendimento e inclusão de novas funcionalidades por parte dos desenvolvedores.

De acordo com Flores (2011), quando citamos aplicações Web, uma das áreas mais abordadas é a camada de apresentação. Nestas, a boa formatação do código é essencial para maior qualidade de segurança, leiaute, acessibilidade e desempenho do sistema. Sabendo-se que os sistemas mais antigos ou pouco planejados podem apresentar problemas estruturais na criação dos códigos-fonte de suas páginas, o trabalho tem como proposta através de um estudo de caso, identificar falhas relacionadas à segurança, leiaute, acessibilidade e desempenho de uma aplicação Web, realizando a reestruturação do código utilizando técnicas de Refatoração de Aplicações Web citadas por Harold (2010).

2. Refatoração

O conceito de refatoração vem originalmente da comunidade de programação orientada a objetos, datando do ínicio dos anos 90, mas sendo popularizada por Martin Fowler em 1999 com o livro "Refactoring" (Addison-Wesley, 1999). Fowler (1999) define refatoração como o processo de aperfeiçoamento do código-fonte de um sistema de software, tornando-o mais bem entendido, menos custoso na modificação e sem mudanças no comportamento externo.

Outro conceito abordado por Harold (2010) cita que a refatoração é a melhoria gradual de uma base de código por meio de pequenas mudanças que não modificam o comportamento de um programa, normalmente com ajuda de ferramentas automatizadas. O objetivo da refatoração é remover problemas de código - muitas vezes legado, produzindo código mais claro, mais fácil de manter, depurar e adicionar funcionalidades.

Baseando-se nesses conceitos pode-se dizer que refatoração é o emprego de técnicas para melhorar o projeto do software auxiliando na reestruturação do código-fonte, visando promover atributos não funcionais de software, tais como extensibilidade, modularidade, reusabilidade, complexidade e eficiência.

3. Refatoração para Aplicações Web

Técnicas de refatoração são amplamente utilizadas em linguagens de programação como Java e C. Porém, não é só código orientado a objetos ou linguagens orientadas a objetos que podem produzir código ruim e com necessidade de refatoração. Pode-se dizer que não somente linguagens de programação, mas qualquer sistema suficientemente complexo e mantido ao longo do tempo pode se beneficiar de refatoração.

Em um sistema Web o lado servidor de uma grande aplicação distribuída, funciona geralmente sobre uma base de dados relacional, enquanto que o lado cliente é uma ou mais páginas web, quase sempre construídas com códigos HTML (HyperText Markup Language), CSS (Cascading Style Sheets) e JavaScript. O HTML tornou o desenvolvimento dessas aplicações mais rápidas, mas não as tornou mais fáceis, mais simples ou menos complexas. Vários desses sistemas, tal como qualquer outra aplicação de vida suficientemente longa podem acumular problemas de código. As novas tecnologias não se adaptam perfeitamente com as antigas, tornam os sistemas mais lentos, abrem brechas de segurança e deixam a alteração ou implementação de funcionalidades comprometida. Em muitos casos não é possível jogar a aplicação fora e criar uma nova, portanto a importância de aplicar as técnicas de refatoração é visível.

Harold (2010) aborda mais de 60 sugestões de refatoração para códigos HTML. As técnicas são subdivididas em Documentos bem formados, Validade, Leiaute, Acessibilidade, Aplicações Web e Conteúdo. As técnicas abordadas nas subseções seguintes foram escolhidas, pois se adequam ao que se foi proposto para a melhoria das aplicações estudadas.

3.1. Validade

A validade garante que somente elementos e atributos especificados no HTML apareçam, mostrando o mesmo conteúdo para usuários de diversos navegadores. Para Harold (2010), adicionar atributos *alt* em imagens é uma técnica de refatoração

essencial, uma vez que dá assistência para usuários com deficiências de visão à medida que navegadores de áudio vierem embarcados em celulares, carros e outros dispositivos direcionados a esse público.

3.2. Leiaute

Harold (2010) cita que usar a semântica adequada para cada elemento torna as páginas inteligíveis para leitores de tela e faz com que sejam mostradas apropriadamente para diferentes plataformas. Muitos elementos, como o *table* são usados abusivamente para tornar as páginas com aparência agradável. Atualmente, tem-se trabalhado bastante com o desenvolvimento de páginas Web utilizando o conceito "tableless", ou seja, sem tabelas. O padrão de desenvolvimento com o CSS e tags *div* permite a separação da camada de apresentação, tornando a manutenção das páginas mais fácil, dentre outros benefícios.

3.3. Acessibilidade

A web tem o poder de integrar à sociedade as pessoas com necessidades especiais. As páginas Web devem ser projetadas de modo que não precisem saber qual o tipo de dispositivo o usuário está utilizando, seja ele um monitor de vídeo ou um leitor de tela. Harold (2010) lembra que os usuários com limitações visuais que utilizam leitores de tela não podem usar o leiaute visual de uma página para determinar quais rótulos estão associados a quais campos. É preciso rotular, deixar explícito cada um dos campos não ocultos, de forma que eles sejam facilmente identificados pelos dispositivos. Para cada elemento visível do tipo *input, textarea ou select*, deve existir ao menos um elemento do tipo *label* associado.

3.4. Aplicações Web

Atualmente, muitos sites não são mais apenas estáticos, são aplicações completas para entrada de dados, processamento de texto, jogos e muito mais. Com a evolução das aplicações vem também a necessidade de tornar as páginas mais rápidas e seguras, de modo que o usuário passe a utilizar essas ferramentas sem medo de que seus dados sejam expostos e que o sistema suporte toda a demanda requerida.

Um exemplo de refatoração é a substituição de requisições GET inseguras por POST. As requisições GET utilizam a própria URL para enviar os dados para o servidor, essas requisições podem ser navegadas por robôs, pré-carregadas, armazenadas em cache, repetidas ou acessadas automaticamente. Operações inseguras como, por exemplo, cadastrar, alterar ou excluir um cliente, devem ser realizadas apenas via POST, evitando que os dados sejam manipulados sem o consentimento do usuário.

Outro exemplo de refatoração é adicionar tipos de formulários Web 2.0, eles trazem novos campos de entrada fortemente tipados (*email, date, time, datetime, datetime-local, month, week, number, tel e url*), novos atributos para restrições, novas interfaces e novos eventos DOM (Document Object Model) para validação e acompanhamento de dependências. Eles permitem que os navegadores forneçam componentes mais apropriados de interface gráfica para a entrada de dados.

Na questão segurança, usar sequência de escape para as entradas de usuário é fundamental. Atualmente, a fonte mais comum de falhas de segurança na Web é a injeção de código SQL (Structured Query Language). Harold (2010) cita que é

provavelmente mais fácil encontrar um site baseado em banco de dados com uma vulnerabilidade de injeção de código SQL que um site que não possua uma. Harold (2010) ainda diz que a injeção de código SQL tem levado ao roubo de dados confidenciais de clientes, fraudes de cartão de crédito, *phishing*, *spams* e a quase todos os outros tipos de crimes auxiliados por computador que possamos imaginar.

4. Trabalhos Relacionados

Nesta seção serão analisados alguns trabalhos relacionados com a proposta. O trabalho de graduação de Flores (2011) objetivou a busca de oportunidades de Refatoração em Aplicações Web através da criação de buscas XQuery em páginas codificadas XHTML, visando melhorar a estrutura do código fonte, bem como o desempenho de aplicações Web.

Na dissertação de mestrado Boniati (2009) propõe identificar, automatizar e aplicar técnicas de refatoração em aplicações de alto desempenho escritas em linguagem Fortran (não orientadas a objetos) com vistas ao ganho de desempenho em relação a suas construções originais. A tese de doutorado de Piveta (2009) trabalha com muitos conceitos relacionados à refatoração, em especial a busca de oportunidades para refatoração, de um contexto mais amplo, tendo como objetivo prover um processo detalhado para refatoração.

5. Resultados Esperados e Considerações Finais

O presente trabalho apresentou as motivações para a realização de um estudo de caso com o objetivo de aplicar técnicas de refatoração para aplicações Web. Trata-se de um trabalho em andamento do qual, por meio de sua continuidade, pretende-se aprofundar o estudo das principais técnicas para refatoração de aplicações Web citadas por Harold (2010) e por meio de um estudo de caso aplica-las em sistemas legados na tentativa de reestruturar seus códigos-fonte sem que isso interfira em seu funcionamento.

A refatoração vem sendo utilizada há anos e demonstra ser uma excelente prática quanto referimos a reestruturação do código de aplicações legadas e/ou mal planejadas ao longo do tempo. Espera-se por meio da continuidade deste trabalho documentar o processo de aplicação das técnicas escolhidas bem como demonstrar os resultados e benefícios alcançados (segurança, leiaute, acessibilidade, desempenho, etc.)

Referências

Boniati, B. B. (2009), Refatoração de Programas Fortran de Alto Desempenho. Universidade Federal de Santa Maria, Dissertação de Mestrado.

Flores, P. L (2011), Busca de Oportunidades de Refatoração em Aplicações Web. Universidade Federal de Santa Maria, Trabalho de Graduação.

Fowler, M (1999), Refatoração: Aperfeiçoamento e Projeto, Bookman.

Harold, E. R. (2010), Refatorando HTML, Bookman.

Piveta, E. K. (2009), Improving the Search for Refactoring Opportunities on Object-Oriented and Aspect-Oriented Software, Universidade Federal do Rio Grande do Sul. Tese de Doutorado.