UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

MATEUS VINICIO DUARTES FERNANDES

OTIMIZAÇÃO DO SGTCC POR MEIO DE REFATORAÇÃO E BOAS PRÁTICAS DE DESENVOLVIMENTO

GUARAPUAVA

2024

MATEUS VINICIO DUARTES FERNANDES

OTIMIZAÇÃO DO SGTCC POR MEIO DE REFATORAÇÃO E BOAS PRÁTICAS DE DESENVOLVIMENTO

Optimization of the SGTCC through Refactoring and Best Development Practices

Proposta de Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná.

Orientador: Profª. Drª. Renata Luiza Stange

Coorientador: Prof. Dr. Diego Marczal

GUARAPUAVA

2024

4.0 Internacional

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

SUMÁRIO

1 INTRODUÇÃO

O Trabalho de Conclusão de Curso (TCC) é uma parte essencial de muitos cursos de graduação, sendo uma ferramenta de avaliação que permite aos alunos aplicar na prática os conhecimentos adquiridos durante sua formação. No ambiente acadêmico, os estudantes têm a chance de desenvolver seus projetos ao longo do tempo, seguindo um cronograma que se alinha com o planejamento do curso e contando com a orientação de um professor (SEVERINO, 2013).

No curso de graduação em Sistemas para Internet (TSI) da Universidade Tecnológica Federal do Paraná (UTFPR), Campus Guarapuava, os alunos desenvolvem e implementam seus projetos na área da tecnologia. A gestão do processo dos TCCs no curso era realizada de forma manual, necessitando que o envio, revisão e correção dos projetos fossem feitas por meio de documentos impressos. Com a digitalização, todo esse processo foi modernizado, permitindo com que a gestão e organização da avaliação dos trabalhos dos acadêmicos fosse feita de forma digital, trazendo mais agilidade e facilidade para os professores e alunos.

O Sistema de Gestão de TCC (SGTCC) do Curso de Sistemas para Internet foi idealizado por Dias (2015) em 2015, marcando a primeira fase do projeto, onde foram identificados os requisitos e analisadas as necessidades dos usuários. Esta fase estabeleceu as bases para o desenvolvimento do SGTCC, focando no problema do aumento no número de TCCs, que demandava uma melhor organização e acompanhamento. Em 2019, na segunda fase do projeto, foi desenvolvido um novo sistema com base nas diretrizes e aprendizados do sistema idealizado anteriormente. Esse desenvolvimento, conduzido por Silva (2019), trouxe evolução ao projeto inicial, juntamente com a integração de novas funcionalidades como: assinatura eletrônica de documentos, otimização na usabilidade de filtros e buscas, melhorias nos cadastros de orientações e bancas de defesa. Em 2023, Lima (2023) iniciou a terceira etapa do projeto, que teve como foco a identificação de problemas de usabilidade na interface do SGTCC, com o objetivo de propor melhorias utilizando técnicas de *UX Design*. O objetivo foi garantir uma experiência de uso mais eficiente e agradável. Durante essa etapa, novos *layouts* de tela foram desenvolvidos com base em estudos sobre usabilidade, e testes práticos com os usuários foram conduzidos para coletar *feedbacks*. As sugestões recebidas possibilitaram ajustes importantes, aprimorando a interação e a eficácia do sistema. Atualmente, estão em andamento outros projetos que envolvem melhorias no SGTCC, um deles propõe a atualização de tecnologias, atualmente desatualizadas e reescrita de código e o outro melhorias na interface gráfica e funcionalidades.

Apesar do sistema atender aos requisitos principais para a gestão dos TCCs, frequentemente surgem novas necessidades, que podem ser relacionadas tanto na interface gráfica, como a adição de novas funcionalidades. Com a demanda desses novos requisitos e prezando pela evolução contínua do software, o projeto está avançando para a quarta fase, onde pretende-se realizar melhorias na estrutura interna do código. Nesta etapa, o foco está na re-

fatoração do código existente e na aplicação de boas práticas de desenvolvimento de software, com o objetivo de melhorar a legibilidade e a manutenção do sistema. Além disso, a aplicação de padrões de projeto, será fundamental para organizar o código de maneira mais eficiente, facilitando futuras manutenções.

Neste contexto, é essencial considerar as Leis de Lehman, que estabelecem que sistemas de software em uso contínuo precisam ser mantidos e evoluídos para manter sua utilidade e relevância (LEHMAN; BELADY, 1985). Assim, a otimização e a aplicação de boas práticas no desenvolvimento de software são ações que não apenas seguem as Leis de Lehman, mas também as suportam, ajudando a prolongar a vida útil do sistema, melhorar sua qualidade e facilitar sua adaptação a novos requisitos (SOMMERVILLE, 2011).

1.1 Objetivo geral

O objetivo geral deste Trabalho de Conclusão de Curso é analisar, identificar e implementar melhorias no código do Sistema de Gestão de TCC com o intuito de tornar o código mais fácil de manter e expandir, além de promover uma maior reutilização de componentes. Para isso, serão considerados quatro aspectos principais: refatoração, padrões de projeto, arquitetura limpa e princípios de código limpo.

1.2 Justificativa

Dado o crescimento e a evolução contínua do SGTCC, tornou-se necessário aprimorar sua estrutura interna, assegurando uma base sólida e escalável que facilite a manutenção e acompanhe a demanda de novos requisitos. Para isso, serão aplicados os princípios de Código limpo (do inglês, *Clean Code*), Arquitetura Limpa (do inglês, *Clean Architecture*) e técnicas de refatoração, abordagens amplamente reconhecidas por contribuírem com a clareza e a adaptabilidade de sistemas de software ao longo do tempo.

O conceito de Código Limpo destaca a importância de escrever código claro, objetivo e de fácil compreensão. De acordo com Martin (2011), um código limpo é marcado pela simplicidade e pela facilidade de interpretação, o que permite aos desenvolvedores entender e modificar o sistema de maneira mais eficiente. No contexto do SGTCC, esses princípios favorecem a integração de novos desenvolvedores ao projeto, minimizam a ocorrência de erros e contribuem para a durabilidade do sistema. Com um código bem estruturado, o tempo necessário para resolver problemas é reduzido, e os processos de manutenção são otimizados, tornando um sistema mais confiável e eficaz.

A Arquitetura Limpa oferece uma visão estrutural abrangente que favorece a organização modular e uma clara definição de responsabilidades entre os componentes do sistema. Segundo Martin (2019), essa arquitetura é projetada para separar as regras de negócio dos de-

talhes de implementação, o que proporciona uma maior flexibilidade para futuras alterações e atualizações. No SGTCC, sua implementação poderá aumentar significativamente a escalabilidade, permitindo adaptações sem comprometer funcionalidades existentes, mantendo a coesão do sistema e facilitando expansões e integrações com novas tecnologias.

A refatoração é essencial para reorganizar e simplificar a estrutura interna do código, eliminando redundâncias e aumentando a eficiência do sistema. Segundo Fowler (2019), essa prática facilita a comunicação entre desenvolvedores e contribui para a robustez e durabilidade do software. No SGTCC, será crucial para manter a eficiência e prepará-lo para a adição de novas funcionalidades, sem comprometer a estabilidade. Além disso, um código bem escrito e contextualizado é fundamental para a continuidade e colaboração no desenvolvimento, especialmente em um ambiente acadêmico onde novos desenvolvedores frequentemente ingressam no projeto. A clareza e consistência do código, também destacadas por Fowler (2019), facilitam a comunicação e agilizam melhorias e correções, reduzindo a curva de aprendizado e promovendo uma evolução contínua do sistema no contexto do SGTCC.

Com base nos princípios de Código Limpo, Arquitetura Limpa e Refatoração, este trabalho propõe um estudo do código atual do SGTCC com o objetivo de identificar problemas e implementar melhorias na estrutura interna. Esses aspectos de otimização são fundamentais para garantir que o software funcione adequadamente, mas também esteja preparado para futuras modificações.

2 PROPOSTA

Considerando o desenvolvimento e aprimoramento contínuo das funcionalidades do SGTCC em suas fases anteriores, esta etapa busca dar ênfase na otimização da arquitetura interna do SGTCC, com foco na refatoração do código, na aplicação de padrões de projeto e boas práticas de desenvolvimento de software. Assim, esta proposta visa aprimorar a legibilidade e facilitar a manutenção do SGTCC, além de permitir uma integração mais eficiente de novas funcionalidades, atendendo à crescente demanda de TCCs e às necessidades de aperfeiçoamento do processo de avaliação acadêmica.

Com o avanço tecnológico, intensificação de uso do sistema e o aumento do número tanto de usuários como de desenvolvedores do sistema, surgiram necessidades relacionadas à escalabilidade, desempenho e facilidade de manutenção do software. Nesse contexto, a otimização tornou-se uma necessidade essencial para garantir a qualidade, evolução e sustentabilidade do sistema a longo prazo.

O presente trabalho se classifica na Engenharia de Software como Manutenção de Software, que abrange três tipos principais (SOMMERVILLE, 2011):

1. Manutenção preventiva: envolve a atualização de tecnologias para garantir qualidade, segurança e disponibilidade, e a refatoração é uma forma dessa manutenção, pois busca melhorar a estrutura do software e reduzir sua complexidade.

2. Manutenção corretiva: foca na correção de erros que podem surgir após atualizações, garantindo o funcionamento adequado das aplicações.

3. Manutenção perfectiva: busca melhorias contínuas no código, aplicando princípios de design, para aumentar a legibilidade, escalabilidade e eficiência do sistema.

A seguir serão apresentados alguns exemplos de problemas que justificam esta necessidade. Dentre os problemas específicos que foram identificados no código atual, que comprometem a eficiência do sistema é o uso inadequado de Enums [1]. No estado atual, os Enums estão sendo implementados de forma incorreta, resultando em um código que é mais complexo do que o necessário. A correta implementação dos Enums trará uma organização mais clara das opções e estados no sistema, simplificando a manutenção e evitando erros futuros.

Outro problema identificado são o uso de Concerns [2], que deveriam servir para reaproveitar código de forma modular, estão sendo usados apenas como funções extras para algumas classes ou modelos específicos. Esse uso acaba dificultando a manutenção e vai contra a ideia original dos Concerns, que é tornar o código mais organizado e reutilizável. Reestruturar esses elementos ajudará a deixar o código mais coeso e fácil de manter.

[1] Um Enum é um atributo que permite mapear valores para inteiros no banco de dados, mas pode ser consultado pelo nome. Ruby on Rails (2024a)

[2] Um concern é um módulo que organiza funcionalidades específicas e comuns a vários modelos ou controladores, facilitando a manutenção e o reuso de código no sistema.Ruby on Rails (2024b)

Por fim, identificou-se que a geração manual do calendário acadêmico, que atualmente exige a inserção do semestre e ano, sem a definição de períodos com datas exatas de início e fim. Isso acaba limitando a precisão no controle dos TCCs e aumentando o risco de inconsistências, especialmente em situações como greves, onde o semestre pode se prolongar além do planejado. A implementação de um sistema que permita configurar períodos específicos com datas automatizadas de início e término contribuirá para uma gestão mais eficiente e organizada do processo.

Nesta quarta fase do desenvolvimento do SGTCC, o foco é garantir que o sistema continue funcionando bem e se prepare para futuras mudanças. Desta forma, este trabalho propõe identificar problemas no código do Sistema de Gestão de TCC que dificultam a manutenibilidade e reutilização; refatorar o código para eliminar duplicações e reduzir complexidade; implementar padrões de projeto adequados para melhorar a organização do código; aplicar os princípios de arquitetura limpa para modularizar o sistema; seguir os princípios de código para garantir legibilidade e manutenção; documentar o processo de refatoração e as melhorias realizadas.

Com isso, busca-se não apenas resolver os problemas atuais, mas também deixar o sistema pronto para futuras evoluções junto com as necessidades da universidade e da comunidade acadêmica, mantendo sua importância e facilitando sua manutenção ao longo do tempo.

É importante destacar que não faz parte do escopo deste trabalho avaliar os resultados das melhorias implementadas por meio de métricas de qualidade de código, como redução da complexidade ciclomática, cobertura de testes e modularidade, para verificar o impacto das mudanças realizadas na manutenibilidade e expansibilidade do sistema.

3 CONSIDERAÇÕES FINAIS

Desde a criação do sistema, quando foram identificados os requisitos e iniciada a estruturação do SGTCC, o sistema passou por constantes evoluções. Em 2019, com base na versão inicial do sistema, foi desenvolvido um novo sistema e novas funcionalidades como a assinatura eletrônica e melhorias nos cadastros e filtros foram adicionadas. Na terceira fase, em 2023, o foco foi aprimorar a usabilidade e a interação com o sistema, proporcionando uma experiência mais fluida para os usuários. Agora, nesta quarta fase, a atenção se volta à estrutura interna, com a refatoração do código e a adoção de boas práticas, visando otimizar a manutenção e facilitar futuras expansões.

Ao longo do desenvolvimento, com o aumento de usuários e TCCs levantou-se a necessidade de um sistema mais robusto, com escalabilidade e desempenho aprimorados. A refatoração do código não apenas tornará o sistema mais eficiente, mas também permitirá que novas funcionalidades sejam implementadas de forma mais rápida e fácil, mantendo a qualidade e organização do software. Além disso, a adoção de padrões de projeto e arquitetura limpa trará mais clareza e facilidade para futuros desenvolvedores que venham a colaborar no projeto.

Este projeto visa continuar e agregar valor às fases anteriores do SGTCC por meio da otimização de sua estrutura interna. Com a refatoração do código e a adoção de boas práticas de desenvolvimento, a manutenção do sistema será mais simples, permitindo a implementação de futuras melhorias com maior eficiência. Ao melhorar a qualidade do código, espera-se que o sistema acompanhe as mudanças tecnológicas e atenda às necessidades acadêmicas de forma mais eficiente.

REFERÊNCIAS

DIAS, É. Monografia de Trabalho de Conclusão de Curso, **Desenvolvimento de um sistema para o gerenciamento do processo de Trabalho de Conclusão de Curso do Curso de Tecnologia em Sistemas para Internet da UTFPR Câmpus Guarapuava**. Guarapuava: [*s.n.*], 2015. 53 p.

FOWLER, M. **Refatoração: Aperfeiçoando o Design de Códigos Existentes**. 2. ed. [*S.l.*]: Novatec Editora Ltda., 2019. Tradução autorizada da edição original em inglês, publicada pela Addison-Wesley Professional, Pearson Education, Inc. ISBN 978-85-7522-724-4.

LEHMAN, M. M.; BELADY, L. A. **Program Evolution: Processes of Software Change**. London: Academic Press, 1985.

LIMA, A. C. Monografia de Trabalho de Conclusão de Curso, **Projeto e implementação de interface baseada na experiência do usuário para um sistema de gerenciamento de Trabalho de Conclusão de Curso**. Guarapuava: [*s.n.*], 2023.

MARTIN, R. C. **Código Limpo: Habilidades Práticas do Agile Software**: A handbook of agile software craftsmanship. [*S.l.*]: Alta Books, 2011. 456 p. (Série de Robert C. Martin). Inclui bibliografia, anexos e índice. ISBN 978-85-508-1148-2.

MARTIN, R. C. **Arquitetura Limpa: O Guia do Artesão para Estrutura e Design de Software**. [*S.l.*]: Alta Books, 2019. Tradução autorizada da edição original em inglês, publicada pela Pearson Education, Inc. ISBN 978-85-508-1600-5.

Ruby on Rails. **ActiveRecord::Enum**. 2024. Acesso em: 1 nov. 2024. Disponível em: https://api.rubyonrails.org/classes/ActiveRecord/Enum.html.

Ruby on Rails. **Getting Started with Rails**. 2024. Acesso em: 1 nov. 2024. Disponível em: https://guides.rubyonrails.org/getting_started.html#using-concerns.

SEVERINO, A. J. **Metodologia do trabalho científico [livro eletrônico]**. [*S.l.*]: Cortez Editora, 2013. ISBN 9788524920813.

SILVA, R. Monografia de Trabalho de Conclusão de Curso, **Aperfeiçoamento do Sistema de Gestão de Processos de Trabalho de Conclusão de Curso de Tecnologia em Sistemas para Internet da UTFPR Câmpus Guarapuava**. Guarapuava: [*s.n.*], 2019. 49 p.

SOMMERVILLE, I. **Software Engineering**. 9th. ed. Boston, MA: Pearson, 2011.