TTEA+ COM RECURSOS DE CAPTURA DE MOVIMENTOS PARA JOGOS SÉRIOS

Luis Eduardo Bet, Marcelo da Silva Hounsell

INTRODUÇÃO

O T-TEA é um console de jogos sérios ativos (*exergames*) baseado em projeção para chão interativo, desenvolvido especialmente para o público com Transtorno do Espectro Autista (TEA) (Trindade, 2022). A plataforma dispõe de uma interface que viabiliza o cadastro e a gestão de usuários, a configuração dos componentes de hardware, a definição de parâmetros de execução e a inicialização dos *exergames* que compõem o sistema.

Por tratar-se de um software científico, cuja evolução envolve a contribuição de diversos colaboradores ao longo do tempo, observa-se, de forma semelhante a outros sistemas acadêmicos, uma ênfase maior nas funcionalidades do que em aspectos como modularidade, clareza, manutenibilidade e boas práticas de engenharia de software (Arvanitou *et al*., 2022). Segundo Lehman (1980), "À medida que um programa em evolução é continuamente modificado, sua complexidade tende a aumentar, a menos que sejam realizados esforços para mantê-la ou reduzi-la". Com o objetivo de reduzir a complexidade e ampliar a manutenibilidade do código do T-TEA, foram conduzidas análises sobre a base de código original, seguidas da aplicação de um processo de refatoração, bem como da implementação de melhorias e da correção de erros identificados.

DESENVOLVIMENTO

A ferramenta de análise *SonarQube* (SonarSource, 2025) foi empregada para analisar a base de código do console T-TEA e seus *exergames*, visando identificar problemas de complexidade e manutenibilidade. As métricas consideradas foram o Débito Técnico (DT), que avalia a eficiência da manutenção (Avgeriou *et al*., 2016), e a Complexidade Ciclomática de McCabe (CCM), que mensura decisões lógicas em módulos de software (Mccabe, 1976) e relaciona-se à densidade de erros (Yu; Zhou, 2010). A Tabela 1 apresenta os resultados da análise de toda a base de código.

Uma das partes do código que apresentou maiores problemas foi a interface do console, também denominada "interface pré-jogo", composta por três arquivos: um destinado ao menu, outro ao gerenciamento dos arquivos utilizados e um terceiro responsável pela calibração do jogo. O processo de refatoração concentrou-se inicialmente nos dois primeiros arquivos, deixando a calibração de lado devido a restrições de tempo.

A refatoração, com o objetivo de aprimorar a qualidade do código e atender à correção de erros e novos requisitos, iniciou-se com a adoção da convenção de estilo PEP8, guia de estilo para a linguagem python, e do padrão arquitetural MVC (*Model-View-Controller*), buscando padronizar o estilo de codificação e estruturar adequadamente o sistema, que anteriormente possuía organização indefinida. Além disso, optou-se pelo paradigma de programação orientada a objetos, visando a reutilização de código e compatibilidade com a arquitetura definida.

Durante o processo de refatoração, uma parte de código necessitou ser reescrita para atender os novos padrões definidos e várias técnicas de refatoração foram aplicadas como renomeação, extração de variáveis e remoção de código morto. Simultaneamente outras técnicas de engenharia de software foram adotas durante o processo como padrões de projeto

e princípios de design como SOLID. Além disso, novas funcionalidades foram adicionadas à interface, incluindo manuais do usuário para o console e para os *exergames*, seção de ajuda, tela de abertura (*splash screen)*, registros do sistema (*logs*) e suporte aos idiomas português e inglês.

RESULTADOS

O *SonarQube* foi utilizado para analisar a base de código antes e após a refatoração, permitindo a comparação das métricas. Conforme a Tabela 1, problemas de manutenibilidade foram resolvidos, refletindo na redução do Débito Técnico (DT), embora a CCM total tenha aumentado 248%.

O aumento da CCM é devido à implementação de cinco novas funcionalidades. A complexidade porém, está distribuída uniformemente, com CCM médio de 12 por arquivo, sendo o módulo principal o mais complexo e candidato a futuras melhorias. Durante o processo, um bug crítico de persistência de dados foi corrigido e a legibilidade do código aprimorada, beneficiando os colaboradores que trabalham recorrentemente sobre a base de código.

CONSIDERAÇÕES FINAIS

As análises e modificações aplicadas ao T-TEA, utilizando técnicas de engenharia de software, evidenciam melhorias significativas em manutenibilidade e CCM por arquivo.

Conclui-se que práticas como refatoração devem ser incorporadas ao longo de todo o ciclo de vida do software para aumentar a clareza e apoiar a manutenção corretiva e evolutiva.

Para trabalhos futuros, espera-se que todas as partes do código sejam refatoradas, testes e integração contínua sejam aplicados durante todo o processo, bem como a documentação do código seja elaborada.

Palavras-chave: Engenharia de Software; Jogos Sérios Ativos; Refatoração; *Exergames*; Complexidade ciclomática; Débito Técnico.

ILUSTRAÇÕES

Tabela 1. *Complexidade antes e após refatoração da interface do console.*

Métricas	**Toda a Base de Código**	**Arquivos de Menu e de Gerenciamento de Arquivos**	
		Antes	**Depois**
Linhas de Código (LOC)	7130	591	1231
Número de Arquivos	35	2	18
Número de Classes	25	0	20
Número de Funções	312	77	108
Débito Técnico (minutos)	9060	1738	5
Código Duplicado (%)	27	0	0

Total CCM	1447	85	211
CCM Médio (por arquivo)	41	42	12
Arquivo com Maior CCM	485	66	42
Arquivo com Menor CCM	0	19	3

REFERÊNCIAS

TRINDADE, André Bonetto. **Plataforma de projeção interativa para jogos sérios relacionados ao transtorno do espectro autista.** 2025. Dissertação (Curso de Engenharia Elétrica) - Udesc, 2022. Disponível em: https://repositorio.udesc.br/handle/UDESC/17286.

ARVANITOU, Elvira-Maria; NIKOLAIDIS, Nikolaos; AMPATZOGLOU, Apostolos; CHATZIGEORGIOU, Alexander. Practitioners' Perspective on Practices for Preventing Technical Debt Accumulation in Scientific Software Development. **Proceedings Of The 17TH INTERNATIONAL CONFERENCE ON EVALUATION OF NOVEL APPROACHES TO SOFTWARE ENGINEERING**, [S.L.], p. 282-291, 2022. SCITEPRESS - Science and Technology Publications. DOI:10.5220/0010995000003176.

LEHMAN, M.M.. Programs, life cycles, and laws of software evolution. **Proceedings Of The IEEE**, [S.L.], v. 68, n. 9, p. 1060-1076, 1980. Institute of Electrical and Electronics Engineers (IEEE). DOI:10.1109/proc.1980.11805.

SONARSOURCE. **SonarQube**. Versão 10.5. Genebra: SonarSource, 2025. Disponível em: https://www.sonarsource.com/products/sonarqube/.

AVGERIOU, Paris; KRUCHTEN, Philippe; OZKAYA, Ipek; SEAMAN, Carolyn. Managing Technical Debt in Software Engineering (Dagstuhl Seminar 16162). **Dagstuhl Reports**, [S.L.], v. 6, n. 4, p. 110-138, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. DOI: 10.4230/DAGREP.6.4.110.

MCCABE, T.J.. A Complexity Measure. **IEEE Transactions On Software Engineering**, [S.L.], v. 2, n. 4, p. 308-320, dez. 1976. Institute of Electrical and Electronics Engineers (IEEE). DOI:10.1109/tse.1976.233837.

YU, Sheng; ZHOU, Shijie. A survey on metric of software complexity. **2010 2Nd IEEE INTERNATIONAL CONFERENCE ON INFORMATION MANAGEMENT AND ENGINEERING**, China, p. 352-356, 2010. IEEE. DOI:10.1109/icime.2010.5477581.

DADOS CADASTRAIS

BOLSISTA: Luis Eduardo Bet
MODALIDADE DE BOLSA: PIBITI/CNPq (IT)
VIGÊNCIA: 09/2024 a 08/2025 – Total: 12 meses
ORIENTADOR(A): Marcelo da Silva Hounsell
CENTRO DE ENSINO: CCT
DEPARTAMENTO: Departamento de Ciência da Computação
ÁREAS DE CONHECIMENTO: Ciências Exatas e da Terra / Ciência da Computação
TÍTULO DO PROJETO DE PESQUISA: Recursos de Captura de Movimentos para Jogos Sérios Ativos
Nº PROTOCOLO DO PROJETO DE PESQUISA: NPP4107-2023