# SMARTYCHECKTOOL: UMA FERRAMENTA PARA DETECÇÃO E REMOÇÃO DE DEFEITOS EM MODELOS SMARTY DE LINHAS DE PRODUTO DE SOFTWARE

Mateus Hideaki Taroda (PIBIC/CNPq/FA/UEM), Edson A. Oliveira Junior (Orientador), Ricardo T. Geraldi
e-mail: mateustaroda@gmail.com, edson@din.uem.br, ricardogeraldi@gmail.com
Universidade Estadual de Maringá / Centro de Tecnologia / Maringá, PR.

**Área de Conhecimento:** Ciência da Computação
**Subárea do Conhecimento:** Metodologia e Técnicas da Computação
**Especialidade:** Engenharia de Software

**Palavras-chave:** Linha de Produto de Software, Inspeção por *Checklist*, Automatização.

**Resumo:**

O objetivo deste projeto é desenvolver uma ferramenta para automatizar a técnica SMartyCheck. Tal técnica foi criada com o intuito de detectar erros nas fases iniciais de uma Linha de Produto de Software, reduzindo possíveis custos relacionados a problemas em fases mais avançadas e aumentando a qualidade dos produtos de software. Para o desenvolvimento da ferramenta, primeiro, foram estudados conceitos de Linha de Produto de Software, a abordagem SMarty e a técnica SMartyCheck. E então, foram escolhidas as tecnologias para a implementação. A ferramenta foi implementada utilizando a linguagem Java em conjunto com a ferramenta XPath. Com a ferramenta é possível inspecionar diagramas de classes e casos de uso SMarty, alcançando o objetivo de automatizar a técnica SMartyCheck. Além disso, foi realizado um estudo empírico qualitativo com o intuito de avaliar a ferramenta. Com o estudo foi possível levantar pontos positivos e negativos da ferramenta. Também foram coletadas sugestões de melhoria da ferramenta, que podem ser implementadas em trabalhos futuros.

## Introdução

Linha de Produto de Software é uma abordagem para reutilização de artefatos de software, com o auxílio do gerenciamento de variabilidades, permitindo a customização desses produtos de acordo com as necessidades dos clientes. Segundo Linden et al. (2007) “Pode-se definir LPS como um conjunto de sistemas de software que compartilham características comuns, as quais podem ser gerenciadas, satisfazendo as necessidades de um segmento em particular de mercado ou missão”.

Os principais benefícios da LPS são: melhoria da qualidade, redução de custos de desenvolvimento, redução do tempo de produção e diminuição de riscos. Esses benefícios vêm do reuso de artefatos, pois como estes artefatos já foram validados, há garantia da qualidade dos mesmos, reduzindo assim os custos de desenvolvimento e o tempo de produção (Geraldi and OliveiraJr, 2017).

Uma família de produtos no contexto de LPS, é um conjunto de sistemas com características comuns, onde cada membro da família é desenvolvido a partir de um núcleo de artefatos, denominado *Core Assets*. Este núcleo de artefatos possui similaridades e variabilidades, onde a variabilidade possui fundamental importância, pois é a forma de distinguir membros da família entre si.

O gerenciamento de variabilidades é uma tarefa essencial em LPS, permitindo a identificação, representação e o rastreamento das variabilidades de uma LPS. Neste contexto está inserida a abordagem *Stereotype-based Management of Variability* (SMarty), baseada em modelos *Unified Modeling Language* (UML) (Oliveira et al. 2013). SMarty é composta por um perfil UML, denominado SMartyProfile, e um processo, denominado SMartyProcess, formado por atividades e diretrizes bem definidas para identificar e rastrear variabilidades UML de uma LPS, utilizando estereótipos propostos em seu perfil. Dessa forma, os artefatos de uma LPS podem ser inspecionados para prevenir defeitos de ocorrerem durante o ciclo de vida dos produtos de software.

Existem diversas técnicas de inspeção de software, entre elas está a técnica de Leitura baseada em Checklist (*Checklist-Based Reading* (CBR)), a CBR é a base da SMartyCheck, que é a técnica que se deseja automatizar no presente trabalho. A CBR não é uma técnica sistemática, ou seja, não fornece instruções sobre “como” realizar a inspeção, mas define “o que” deve ser inspecionado, isto é feito por meio de um documento no formato de lista de verificação, denominado Checklist. Os inspetores recebem a lista com os itens a serem inspecionados, e devem responder com um check para cada defeito encontrado.

Dados os conceitos de Linha de Produtos de Software, Gerenciamento de variabilidades, SMarty, Inspeção de Software e Checklist, pode-se introduzir a técnica SMartyCheck, que está inserida neste contexto. A SMartyCheck é uma técnica de inspeção de software baseada em checklist para diagramas SMarty, fazendo uso da abordagem SMarty para inspecionar diagramas UML de variabilidade de LPS por meio de checklist, contendo uma taxonomia de tipos de defeitos selecionados da literatura. Atualmente, a técnica SMartyCheck suporta diagramas UML de casos de uso e de classes.

## Materiais e métodos

No desenvolvimento do trabalho foi utilizado um computador pessoal com processador Intel Core i5 de 2.5GHz, 6GB de memória RAM, sistema operacional Windows 10, IDE Netbeans 8.2 e Java na Versão 8, update 131.

Inicialmente, foram revisados os conceitos de Linha de Produto de Software, Inspeção, Checklists, SMarty e SMartyCheck. Foram estudados os tipos de defeitos abordados na SMartyCheck e selecionados os que poderiam ser automatizados. Durante a implementação, foram automatizadas as inspeções de diagramas de classes e casos de uso com base na SMartyCheck.

**Resultados e Discussão**

Com o objetivo de avaliar a viabilidade da ferramenta, foi preparado um estudo empírico quantitativo e qualitativo, com base no método TAM (*Technology Acceptance Model*) versão 3, proposto por Venkatesh e Bala (2008). Tal método propõe avaliar técnicas e ferramentas com base em três princípios: 1) Facilidade de uso, 2) Utilidade e 3) Intenção de uso.

As questões qualitativas foram analisadas utilizando procedimentos da teoria fundamentada (*Grounded Theory*) (Corbin and Strauss, 2008). Codificação (Coding) permite designar códigos para trechos de textos (Open Coding), podendo ser agrupados e categorizados (Axial Coding).

A análise do estudo empírico foi baseada nos dados coletados de cinco participantes. O questionário preparado possui questões de caráter quantitativo relacionadas as principais interfaces do sistema, relacionadas a facilidade de uso e utilidade, além disso, conta com questões de intenção de uso futuro da ferramenta no geral. O questionário também possui questões de caráter qualitativo, possibilitando que o participante contribua com sugestões e críticas sobre a ferramenta.

Ao analisar os resultados, percebeu-se que há muitas melhorias que podem ser realizadas na ferramenta no quesito facilidade de uso e intuitividade, entretanto, em relação a utilidade, os resultados foram bons, o que nos leva a entender que a ferramenta é útil. Quanto a intenção de uso, os participantes do estudo empírico concordaram de maneira unânime que utilizariam, caso estivessem trabalhando com projetos cujas variabilidades são gerenciadas via SMarty, e que recomendariam a ferramenta para projetistas que trabalham com LPS.

**Conclusões**

Os objetivos foram alcançados, de maneira que a ferramenta foi, de fato, implementada. A ferramenta inspeciona diagramas de casos de uso e de classes, encontrando os defeitos propostos pela técnica SMartyCheck, e então, os defeitos encontrados são apresentados ao usuário em forma de tabela, possibilitando ainda a filtragem por nomes de classes/casos de uso e

por tipos de defeitos. Além disso, foi implementada uma funcionalidade de rastreabilidade, para visualização de quais classes/casos de uso podem impactar outros (as), quando possuem defeitos. A ferramenta também possibilita a exportação de relatórios em .xls e em pdf.

Além do objetivo de automatizar a técnica SMartyCheck, estava previsto um estudo para avaliar a ferramenta, o que foi alcançado com um estudo empírico quantitativo e qualitativo. Tal estudo envolveu um documento explicando sucintamente a técnica SMartyCheck, e em sequência a utilização da ferramenta. Os participantes então responderam a um questionário sobre a ferramenta.

Como trabalhos futuros, podem ser realizadas melhorias na usabilidade, conforme os resultados levantados no estudo empírico, e também a expansão da ferramenta para outros tipos de diagramas.

## Agradecimentos

Agradecimentos à Fundação Araucária pelo apoio financeiro recebido para viabilizar este projeto.

## Referências

CORBIN, J. M.; STRAUSS, A. L. (2008). **Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory**. Sage Publications, 3 edition.

GERALDI, R. T.; OLIVEIRAJR, E. **Towards Initial Evidence of SMartyCheck for Defect Detection on Product-Line Use Case and Class Diagrams**. JOURNAL OF SOFTWARE, v. 12, p. 379-392, 2017.

LINDEN, F. J. van der, SCHMID, K., ROMMES, E. **Software Product Lines in Action**. Springer-Verlag Berlin Heidelberg. 2007.

OLIVEIRAJR, E.; GIEES, I. M. S.; MALDONADO, J. C.; MASIERO, P. C.; BARROCA, L. **Systematic Evaluation of Software Product Line Architectures.** Journal of Universal Computer Science, 2013, 19(1): 25-52.

VENKATESH, Viswanath; BALA, Hillol. **Technology Acceptance Model 3 and a Research Agenda on Interventions**. Decision Sciences**.** Decision Sciences Institute, p. 273-312. 2008.