Ferramenta Computacional SAATS – Aprimorando sua Eficácia

LUCAS PEREIRA DA SILVA[1], LUIZ CAVAMURA JÚNIOR[2]

[1] Graduando em Tecnologia de Análise e Desenvolvimento de Sistemas, Bolsista PIBIFSP, IFSP, Câmpus Piracicaba, l.silva1998@outlook.com
[2] Professor Mestre e doutorando em Ciência da Computação, IFSP, Câmpus Piracicaba, cavamura@ifsp.edu.br
Área de conhecimento (Tabela CNPq): 1.03.03.02-2 Engenharia de Software

Apresentado no
8° Congresso de Inovação, Ciência e Tecnologia do IFSP
06 a 09 de novembro de 2017 - Cubatão-SP, Brasil

RESUMO: As atividades de teste de software são essenciais devido aos prejuízos causados por falhas no software. O teste de software é uma atividade complexa, demanda tempo e recursos da equipe de desenvolvimento, além de exigir, da equipe, o domínio dos conceitos de teste de software. Assim, ferramentas computacionais são necessárias para aumentar a eficácia e a eficiência das atividades de teste de software. Nesse contexto, a SAATS é uma ferramenta computacional que auxilia as atividades de teste de software estabelecendo, por meio de métricas, extraídas dos arquivos executáveis do software sob teste, uma prioridade na execução dos casos de teste projetados. Este trabalho relata o desenvolvimento de novas funcionalidades na ferramenta SAATS, as quais visam aprimorar a eficácia da ferramenta no auxílio às atividades de teste de software. Esse aprimoramento é baseado na capacidade da ferramenta em considerar, na sistemática adotada pela ferramenta, além das métricas já consideradas, o perfil operacional do software testado, perfil esse a ser obtido por meio de uma análise dinâmica do software testado, ou seja, por meio da sua operação pelos seus usuários.

PALAVRAS-CHAVE: teste de software; perfil operacional do software; qualidade de software.

SAATS computing tool - improving its effectiveness

ABSTRACT: Software testing activities are essential because of the damage caused by software failures. The software test is a complex activity, it demands time and resources of the development team, besides requiring, from the team, the mastery of the concepts of software testing. Thus, computational tools are necessary to increase the effectiveness and efficiency of software testing activities. In this context, SAATS is a computational tool that assists software testing activities by establishing, through metrics, extracted from the executable files of the software under test, a priority in the execution of the designed test cases. This paper reports the development of new functionalities in the SAATS tool, which aim to improve the tool's effectiveness in supporting software testing activities. This improvement is based on the tool's ability to consider, in the systematic adopted by the tool, the metrics already considered, the operational profile of the software tested, a profile to be obtained through a dynamic analysis of the software tested, that is, through of its operation by its users.).

KEYWORDS: Software testing; Software operational profile; Software quality.

INTRODUÇÃO

O software é o principal elemento de sistemas baseados em computador, sistemas esses cuja responsabilidade e complexidade variam (MYERS; SANDLER; BADGETT, 2011). Tais sistemas podem controlar, por exemplo, uma simples agenda telefônica ou gerir e controlar sistemas aeroviários, metroviários e biomédicos. Falhas em tais sistemas, decorrentes de falhas no software, podem causar prejuízos cujo valor pode ser imensurável. Assim, atividades de teste de software, realizadas durante o processo de desenvolvimento do software, são fundamentais para garantir a qualidade do software. O teste de software é uma atividade complexa, demanda tempo e recursos da equipe de desenvolvimento (MYERS; SANDLER; BADGETT, 2011) e, embora sejam essenciais, por não impactarem diretamente na implementação e entrega do software à seu solicitante, deixam de ser realizadas quando surgem problemas, durante o processo de desenvolvimento de software, relacionados a prazos e custo. Assim, o uso de ferramentas computacionais é recomendado para minimizar o esforço referente a realização das atividades de teste. Nesse contexto, a SAATS é uma ferramenta computacional cuja contribuição às atividades de teste é estabelecer, com base na implementação do software a ser testado, uma prioridade na execução dos casos de teste projetados para o referido software. Assim, o objetivo geral deste trabalho é aprimorar a eficácia da ferramenta SAATS agregando funcionalidades que permitam analisar não apenas a maneira com a qual o software, a ser testado, foi implementado, mas, também, como o software é operado por seus usuários, ou seja, analisar o perfil operacional do software.

MATERIAL E MÉTODOS

A SAATS é uma ferramenta computacional que auxilia a realização das atividades de teste de software estabelecendo uma prioridade na execução dos casos de teste. Essa prioridade é estabelecida por meio da criticidade dos caminhos executáveis que compõe o software testado, o qual deve ter sido desenvolvido usando a linguagem de programação Java sob o paradigma orientado a objetos. A SAATS obtém a criticidade dos caminhos executáveis analisando os arquivos executáveis do software testado. Essa análise gera um grafo que representa o software testado e extrai as métricas que estabelecem a criticidade de cada caminho. A Figura 1 mostra uma interface de operação da SAATS.

Para que o objetivo deste trabalho seja alcançado, o software testado será instrumentado para que, enquanto o software é operado por seus usuários, durante os processos de teste alfa e beta, os dados, necessários para identificar o perfil operacional, sejam coletados e registrados em arquivos (logs de operação). O teste alfa é um processo de teste realizado no site da equipe de desenvolvimento e é conduzido por um grupo de usuários do software. O teste beta é realizado no site dos usuários, no qual os desenvolvedores não estão presentes (PRESSMAN, 2014).

Com base no perfil operacional do software testado, as novas funcionalidades serão desenvolvidas, capacitando a ferramenta a: **a)** Identificar as partes do software mais relevantes às operações dos usuários; **b)** Fornecer, por meio do grafo gerado pela ferramenta, uma visualização das partes do software mais relevantes às operações dos usuários, auxiliando a determinar, juntamente com a criticidade estabelecida para os caminhos executáveis, a prioridade de execução dos casos de teste. Essa visualização também auxiliará a avaliar a eficácia dos casos de teste existentes, verificando se a cobertura provida por eles contempla as partes do software mais utilizadas pelos usuários.

Atividades já realizadas: A sistemática a ser adotada para a instrumentação do software testado foi definida e implementada. A instrumentação é feita por meio do paradigma de programação orientado a aspectos (PPOA), o qual tem por princípio a separação de interesses (LADDAD, 2009). Um aspecto, unidade de código escrita sob o PPOA (Figura 2), foi criado para coletar dados durante a operação do software por seus usuários. Ressalta-se que os códigos-fontes do software sob teste não sofrem alterações em decorrência da instrumentação.

Atividades em desenvolvimento: As funcionalidades que permitirão a leitura dos logs de operação e a visualização das partes do software mais relevantes às operações dos usuários estão sendo implementadas. A Figura 3 mostra, de maneira simplificada e conceitual, o resultado esperado ao término da implementação dessas funcionalidades. As funcionalidades que estão em desenvolvimento fornecerão, a partir da representação gráfica do software (Fig. 3.a), uma visualização das partes do software mais relevantes às operações dos usuários (Fig. 3.c), as quais serão identificadas a partir logs de operação obtidos pela instrumentação do software (Fig. 3.b), capacitando, assim, a ferramenta aos propósitos descritos.

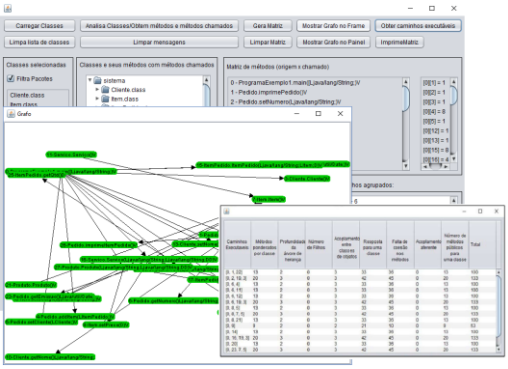

Figura 1 – Interface SAATS

```
public aspect Aspect1 {
    pointcut traceMethods() : (execution(* *(..)));
    before(): traceMethods(){
        FileWriter arquivo;
        Signature sig = thisJoinPointStaticPart.getSignature();
        String line =""+ thisJoinPointStaticPart.getSourceLocation().getLine();
        String sourceName = thisJoinPointStaticPart.getSourceLocation().getWithinType().getCanonicalName();
        SimpleDateFormat df = new SimpleDateFormat("yyyyy-mm-dd hh:mm:ss");
        String momento = df.format(new java.util.Date());
        String metodo = momento + "," +sourceName+","+line+","+sig.getDeclaringTypeName()+","+sig.getName() + "\n";
        try {
            String nomearquivo = "perfil.txt"; arquivo = new FileWriter(new File(nomearquivo),true);
            arquivo.write(metodo); arquivo.close();
        } catch (IOException e) { e.printStackTrace(); } catch (Exception e) { e.printStackTrace(); }
    }
}
```

Figura 2 – Aspecto criado para instrumentação

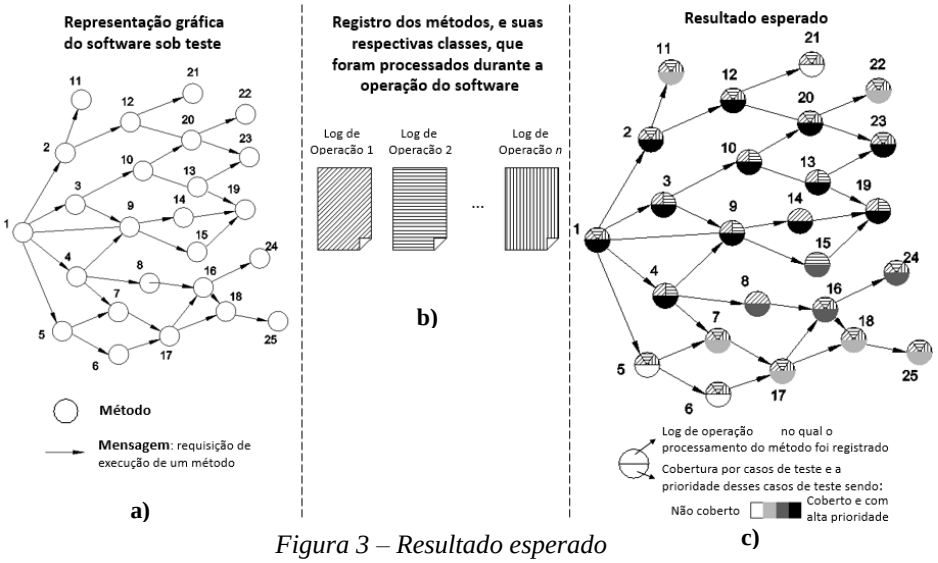

Figura 3 – Resultado esperado

Tendo a Figura 3.c como referência, são exemplos de informações a serem providas pela ferramenta: os métodos 1, 5, 6, dentre outros, são exemplos de métodos relevantes às operações dos usuários pois foram registrados em todos os arquivos de log de operação; os métodos 5 e 6, embora sejam relevantes às operações dos usuários, não são cobertos pelos casos de teste existentes; os métodos 7, 17 e 16, embora também sejam relevantes às operações dos usuários, seus casos de teste não estão classificados com alta prioridade; os casos de teste que cobrem os métodos 3, 10 e 8 têm maior prioridade que os casos de teste que cobrem os métodos 7, 17 e 16 embora os métodos 7, 17 e 16 tenham maior relevância às operações dos usuários que os métodos 3, 10 e 8.

RESULTADOS E CONCLUSÕES

Este trabalho relata as atividades referentes à implementação de novas funcionalidades na ferramenta computacional SAATS, cujo propósito é aumentar a eficácia da ferramenta ao auxiliar as atividades de teste de software. As atividades para identificação do perfil operacional do software, por meio dos logs de operação, no qual as novas funcionalidades estão baseadas, já foram realizadas e atenderam às expectativas do projeto. Com isso, acredita-se que as demais atividades, relacionadas às funcionalidades que usam o perfil operacional do software irão satisfazer aos objetivos para as quais foram idealizadas. Após o término da implementação das novas funcionalidades, estudos de caso serão realizados para constatar a melhoria na eficácia da ferramenta SAATS.

REFERÊNCIAS

LADDAD, R. **AspectJ in Action: Enterprise AOP with Spring Applications**. 2nd. ed. Greenwich, CT, USA: Manning Publications Co., 2009.

MYERS, G. J.; SANDLER, C.; BADGETT, T. **The Art of Software Testing**. 3rd. ed. [s.l.] Wiley Publishing, 2011.

PRESSMAN, R. S. **Engenharia de Software**. eighth Edi ed. New York, NY: McGraw-Hill, 2014.