https://eventos.utfpr.edu.br//sicite/sicite2018

Avaliação experimental de Teste Baseado em Modelo para aplicações Android

Experimental Model-Based Testing evaluation for Android applications

Henrique Neves da Silva
hen123neneves@gmail.com
Universidade Tecnológica Federal do Paraná, Cornélio Procópio, Paraná, Brasil
André Takeshi Endo
andreendo@utfpr.edu.br
Universidade Tecnológica Federal do Paraná, Cornélio Procópio, Paraná, Brasil

Recebido: 04 set. 2018.
Aprovado: 04 out. 2018.

RESUMO

Contexto: Teste Baseado em Modelo (TBM) é uma abordagem que permite testadores representarem o comportamento do sistema testado como modelos, especificando entradas e suas saídas esperadas. A partir destes modelos, ferramentas existentes podem ser empregadas para gerar casos de teste automaticamente. Enquanto o TBM representa um passo promissor para a automação da geração de casos de teste, a qualidade do modelo projetado pelo testador pode ter impacto, positivo ou negativamente, na sua capacidade de revelar falhas (ou seja, a eficácia do teste). *Objetivo:* Neste contexto, conduzimos um experimento preliminar para avaliar o impacto causado por diferentes testadores ao projetar um modelo de teste para a mesma funcionalidade. *Método*: no experimento, os participantes usaram Grafos de Sequência de Eventos e a ferramenta de suporte `FourMA` para criar modelos de teste para duas aplicações móveis: `arXiv-mobile` e `WhoHasMyStuff`. Dos modelos de teste, casos de teste foram gerados usando `FourMA` e concretizados por meio do framework `Robotium`. A fim de medir o impacto de diferentes testadores, utilizamos cobertura de código (nomeadas, *instruction* e *branch coverage*) como estimativa da eficácia do teste. *Resultados:* Baseado nos resultados obtidos, observamos alta variação de cobertura de código entre os testadores. Nenhum testador foi capaz de produzir um modelo de teste que inclui todos os modelos de outros testadores em relação a cobertura de código. Além disso, o fator de aprendizagem parece não reduzir a variação de cobertura de código. A relação entre tamanho do modelo, tempo de modelagem e cobertura de código foram inconclusivos. *Conclusão:* Concluímos que mais esforços de pesquisa sobre a etapa de modelagem do TBM é necessária para não apenas reduzir a variação entre os testadores, mas também melhorar sua eficácia.

PALAVRAS-CHAVE: Teste Baseado em Modelo. Testes Automatizados. Grafo de Sequência de Eventos. Android. Aplicações Móveis. Estudo experimental.

ABSTRACT

Context: Model-Based Testing (MBT) is an approach that allows testers to represent the behavior of the system under test as models, specifying inputs and their expected outputs. From such models, existing tools might be employed to generate test cases automatically. While MBT represents a promising step towards the automation of test case generation, the quality of the model designed by the tester may impact, either positively or negatively, on its ability to reveal faults (i.e., the test effectiveness). *Objective:* In this context, we conducted a preliminary experiment to evaluate the impact caused by different testers when designing a test model for the same functionality. *Method*: In the experiment, the participants used Event Sequence Graphs and its supporting tool `FourMA` to create test models for two mobile apps: `arXiv-mobile` and `WhoHasMyStuff`. From the test models, test cases were generated using `FourMA` and concretized by means of the `Robotium` framework. In order to measure the impact of different testers, we employed code coverage (namely, instruction and *branch* coverage) as an estimation of test effectiveness. *Results:* Based on the results obtained, we observe high variation of code coverage among the testers. No tester was capable of producing a test model that subsumes all other testers' models with respect to code coverage. Moreover, factor

learning seems not to reduce the code coverage variation. The relation between model size, modeling time, and code coverage were inconclusive. *Conclusion:* We conclude that further research effort on the MBT's modeling step is required to not only reduce the variation between testers, but also improving its effectiveness.

KEYWORDS: Model-Based Testing. Automated Tests. Event Sequence Graph. Android. Mobile Apps. Empirical Study.

INTRODUÇÃO

O teste de software é uma prática muito utilizada para avaliar a qualidade de um software, sendo responsável por mais de 50% no custo total do desenvolvimento de um software (ORSO e ROTHERMEL, 2014). Segundo Garousi e Elberzhager (2017), 79 bilhões de euros foram gastos com atividades de teste, em todo o mundo, no ano de 2010. Para o ano de 2014 foi esperado um aumento para 100 bilhões. Assim, é visível a importância de aprimorar as técnicas e abordagens de teste para reduzir os custos envolvidos nas atividades de teste (ANAND et al., 2013).

Uma possível abordagem para se testar aplicações móveis é o Teste Baseado em Modelo (TBM). O TBM consiste na elaboração de modelos que definem o comportamento esperado do System Under Test (SUT) e a partir destes modelos casos de teste são gerados e, em seguida, executados no SUT (UTTING et al., 2006). Dada a presença do fator humano no processo de modelagem da técnica TBM, fica evidente a oportunidade de se investigar quais aspectos no escopo de modelagem podem influenciar na efetividade da execução dos testes.

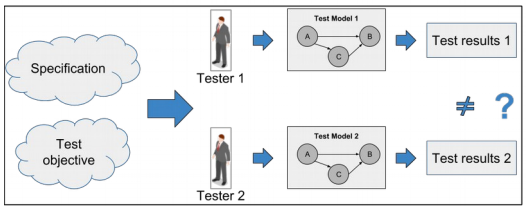
O objetivo deste estudo é avaliar o impacto na efetividade dos testes ao se criar diferentes modelos de teste no contexto do TBM. Especificamente, trabalhou-se com a ferramenta FourMA (ferramenta desenvolvida por FARTO e ENDO em 2017 que utiliza a técnica TBM em aplicações móveis desenvolvidas na plataforma Android) nas etapas de construção do modelo e geração dos casos de teste, observando principalmente a relação entre o modelo criado com a métrica de efetividade na estratégia de exploração: o quanto de código da aplicação é executado durante o teste. Com este objetivo, foi conduzido um experimento em que os participantes desenvolveram dois modelos de teste para duas aplicações diferentes. A partir dos modelos criados, foi analisado a cobertura de código dos casos de teste que cada modelo forneceu.

Este estudo foi divido da seguinte forma: a Seção Configuração do estudo define o experimento do trabalho, explicando o objetivo e como foi conduzido. A Seção de Análise de Resultados elenca os resultados obtidos de forma empírica. A Seção de Discussão e Conclusão sintetiza e discute os principais resultados do experimento, trazendo consigo novas diretrizes que podem ser exploradas em trabalhos futuros.

CONFIGURAÇÃO DO ESTUDO

Com base no conteúdo exposto o presente estudo analisa o impacto causado quando se tem modelos de teste projetados por diferentes testadores no contexto do TBM de aplicações móveis (Figura 1).

Figura 1 – Dois testadores diferentes projetando um modelo de teste



Fonte: Elaborado pelo autor

QUESTÃO DE PESQUISA

Para concentrar o estudo e orientar as etapas de investigação e análise definiu-se a seguinte questão de pesquisa: “Qual o impacto na cobertura de código considerando que diferentes testadores projetem modelos de teste para uma mesma funcionalidade? ”. O cenário que descreve a questão de pesquisa, encontra-se na Figura 1, em que dado a mesma especificação de teste, diferentes testadores trabalham no processo de modelagem, cada modelo construído representa a abstração do comportamento de alguma funcionalidade da aplicação testada. Como cada testador possui sua forma de interpretar, é natural de se pensar que o modelo construído por um testador será diferente do construído por outro. Assim, determinar o impacto na cobertura de código de cada modelo concebido foi a dúvida que fomentou o surgimento desta questão de pesquisa.

HIPÓTESES

Hipótese nula. A hipótese nula H_0 que representa uma afirmação que não existe relação entre os fenômenos estudados, é definida como: “Não há impacto significativo na cobertura de código quando diferentes testadores projetam modelos de teste para uma mesma funcionalidade”.

Hipótese alternativa. A hipótese alternativa (ou hipótese de pesquisa) confronta a hipótese nula H_0. Para este experimento a hipótese alternativa H_1, que representa a afirmação da qual a H_0 rejeita, é definida como: “Existe impacto significativo na cobertura de código a partir do momento que diferentes testadores projetam modelos de teste para uma mesma funcionalidade”.

SELEÇÃO DOS SUJEITOS

O experimento controlado foi realizado nas dependências da Universidade Tecnológica Federal do Paraná Campus Cornélio Procópio. Os 12 participantes foram selecionados por cursarem a disciplina Tópicos Avançados em Teste de Software. Dentro da população total dos sujeitos criou-se de maneira aleatório dois grupos com o objetivo de alternar a interação com as aplicações móveis de cada participante. Desta forma é possível evidenciar se existe alguma correlação no fator aprendizagem (Tabela 1).

Tabela 1 – Condução do experimento

Grupos	Primeira interação	Segunda interação
GRUPO 1	arXiv-mobile	WhoHasMyStuff
GRUPO 2	WhoHasMyStuff	arXiv-mobile

Fonte: Elaborado pelo autor

ROTEIRO

O roteiro como um todo utilizou 5 aulas de 50 minutos cada. As duas primeiras aulas foram focadas no treinamento do participante para a execução do experimento. Os minutos iniciais da aula foram alocados para uma breve explicação conceitual envolvendo o TBM, a técnica ESG e a ferramenta `FourMA`.

Para as três aulas restantes aplicou-se o experimento em questão, iniciado com o preenchimento do formulário de perfil pelos participantes. Os grupos tiveram no máximo duas aulas para a criação de dois modelos de teste, um para a aplicação arXiv-mobile e um para a aplicação `WhoHasMyStuff`. No momento da modelagem, delegou-se a cada participante a primeira aplicação com que fariam a interação, de maneira que metade dos participantes começassem com aplicações alternadas. Ao final da primeira interação o modelo criado foi validado e submetido na plataforma moodle.

Posteriormente, os participantes criaram o modelo de teste para a aplicação restante. No momento da finalização, os modelos tiveram o mesmo destino do primeiro modelo criado anteriormente. O final do tempo foi disponibilizado para o preenchimento do questionário pós-experimento.

Dado o fim da aplicação do experimento, todos os modelos construídos pelos participantes foram coletados e validados. A validação consistiu em verificar se o modelo era consistente a nível de grafo e também se condizia com o comportamento da aplicação testada. A validação do grafo foi feita antes da submissão de cada participante a partir da própria ferramenta `FourMA`. A consistência do modelo obedece às mesmas suposições sobre a validade do grafo apresentada na Seção de Referencial Teórico e ainda levanta a questão da representatividade: a sequência de eventos representada no modelo é ou não condizente com o comportamento da aplicação móvel.

ANÁLISE DOS RESULTADOS

Com base na realização do experimento proposto foi possível elencar informações sobre a utilização de TBM para aplicações móveis. O primeiro dado, apresentado pela Tabela 2 consiste na cobertura de instruções e *branches* atingida ao associar os conjuntos de teste derivados dos 12 modelos de teste e executá-los para cada uma das aplicações. Para a aplicação `arXiv-mobile` atingiu-se 68% de cobertura de instruções e 40% de *branches*. Já para a aplicação móvel `WhoHasMyStuff` obteve-se 63% de cobertura de instruções e 40% de *branches*.

Tabela 2 – Total de cobertura atingida ao se executar todos os modelos de teste construídos para cada aplicação móvel utilizada no experimento

Aplicação Móvel	Instruction Coverage	Branch Coverage
arXiv-mobile	68%	40%
WhoHasMyStuff	63%	40%

Fonte: Elaborado pelo autor

Nenhum dos 12 participantes do experimento conseguiu atingir, com seus modelos de teste, os valores da cobertura total. Isso mostra que cada modelo elaborado "percorreu" um caminho diferente nas aplicações durante os testes, executando instruções distintas.

Esta diferença entre a cobertura total e cobertura de cada sujeito fornece sustento para a hipótese de pesquisa do atual trabalho: "Existe impacto significativo na cobertura de código a partir do momento que diferentes testadores projetam modelo de teste para uma mesma funcionalidade". Além da cobertura total, a hipótese de pesquisa pode ser sustentada pela análise de diferentes aspectos presentes em cada modelo de teste elaborado ao longo do experimento.

Ao observar a distribuição das informações de cobertura de instruções e *branches* obtidos pela execução dos modelos de teste construídos ao longo do experimento foi possível elencar os achados da Tabela 3.

Tabela 3 – Fatos observados ao analisar a variação de cobertura

Achados	Fato observado
Achado 1	Nenhum testador foi capaz de produzir um modelo de teste que inclua todos os modelos de outros testadores em relação à cobertura de código.
Achado 2	A execução dos modelos de teste, criados por cada participante, resultam em diferentes valores de cobertura com variações de 11% até 18%.
Achado 3	O fator aprendizagem dos participantes, entre as interações, não implicou em um aumento nos valores de cobertura.
Achado 4	O tempo de modelagem não teve influência na eficiência dos modelos de teste gerados.
Achado 5	Existe uma tendência de que quanto maior o tamanho do modelo de teste, maior a cobertura de código.

Fonte: Elaborado pelo autor

DISCUSSÕES E CONCLUSÃO

A partir da métrica de cobertura foi possível elencar diferentes aspectos presentes no modelo de teste que podem impactar e diferenciar os resultados obtidos. Para que essas nuances pudessem ser notadas, os participantes criaram um modelo de teste em cenários iguais: mesmos requisitos de teste, técnica de modelagem e objetivos.

A primeira diferença observada foram os próprios resultados em si, ou seja, cada modelo de teste após ser concretizado e executado, gerou valores de cobertura de instrução e *branch* divergentes. Este foi o primeiro indício de que cada modelo de teste elaborado testava a aplicação móvel de uma forma quase única (com desvio padrão relativo de até 5,61% para os valores de cobertura de

instruções; 4,49% para *branches*). A evidência desta característica foi a coleta dos dados de cobertura quando os modelos de teste foram executados em conjunto. A cobertura atingida a partir da união de todos os casos de teste dos modelos não foi contemplada por nenhum dos modelos executados de forma singular.

Com os resultados, também foi possível notar que houveram características que não tiveram relação direta com os valores de cobertura. Primeiro avaliou-se o fator aprendizagem observando e comparando a cobertura atingida pelos participantes na primeira e segunda interação com a aplicação móvel. Os valores permaneceram na mesma faixa de valores tanto para a aplicação `arXiv-mobile` quanto para `WhoHasMyStuff`. Em um segundo momento, além do fator de aprendizagem, examinou-se a existência da relação tempo de modelagem e cobertura atingida. Esta relação não pode ser generalizada, pois para a aplicação `arXiv-mobile` a cobertura aumentou de forma proporcional ao tempo de modelagem; por outro lado, para a aplicação `WhoHasMyStuff` a cobertura diminuiu conforme aumentou-se o tempo de modelagem.

A relação entre quantidade de elementos presente em cada modelo e a cobertura atingida mostrou se comportar de forma semelhante para ambas as aplicações.

Após a realização deste estudo, é possível observar que algumas diretrizes podem ser exploradas em futuras análises: (i) investigar mais a modelagem de teste; (ii) como gerar resultados de cobertura de código mais uniformes; (iii) investigar se a técnica de modelagem implica na efetividade da execução dos testes.

O artigo completo deste trabalho foi aceito no simpósio III SAST 2018 e pode ser acessado pelo seguinte link: https://drive.google.com/open?id=18IWIyVfoJMGNdBz-_49EYqUJW4KdJ7Hn.

REFERÊNCIAS

ANAND, S., BURKE, E. K., CHEN, T. Y., CLARK, J., COHEN, M. B., GRIESKAMP, W., HARMAN, M., HARROLD, M. J., MCMINN, P., BERTOLINO, A., et al. (2013). An orchestrated survey of methodologies for automated software test case generation. Journal of Systems and Software, 86(8):1978–2001.

BUDNIK, CHRISTOF J. (2007). Test generation using event sequence graphs. [Elektronische Ressource]. DOI: http://d-nb.info/983413118/34

CHOUDHARY, S. R.; GORLA, A.; ORSO, A. (2015). Automated Test Input Generation for Android: Are We There Yet? (e). In Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), ASE'15, pages 429-440, Washington, DC, USA. IEE Computer Society.

DELAMARO, Marcio E.; MALDONADO, José C.; JINO, Mario. Introdução ao teste de software. Elsevier Editora Ltda. Rio de Janeiro, 2007. 394 p.

ENDO, André T. Model based testing of service oriented applications. Tese (Doutorado). USP - São Carlos, SP. 2013.

ENTIN, V.; WINDER, B.; ZHANG, B.; CLAUS A. A process to increase the model quality in the context of model-based testing. 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Graz, 2015, pp. 1-7.

FARTO, Guilherme de C. Uma Contribuição ao Teste Baseado em Modelo no Contexto de Aplicações Móveis. Dissertação (Mestrado). UTFPR – Cornélio Procópio, PR. 2016.

FARTO, Guilherme de C.; ENDO, André T. (2017). Reuse of model-based tests in mobile apps. The 31st Brazilian Symposium on Software Engineering (SBES). DOI: 10.1145/3131151.3131160

FEVZI, B., J., B. C., and LEE, W. Event-based modelling, analysis and testing of user interactions: approach and case study. Software Testing, Verification and Reliability, 16(1):3–32.

GAROUSI, V. and ELBERZHAGER, F. (2017). Test automation: not just for test execution. IEEE Software, 34(2):90–96.

GUDMUNDSSON, V.; ACETO, L.; BERGTHORSSON, J.; GANESAN, D.; LINDVALL, M. (2016). Model-based Testing of Mobile Systems - An Empirical Study on QuizUp Android App. Electronic Proceedings in Theoretical Computer Science. 208. 16-30. 10.4204/EPTCS.208.2.

IQBAL, M. Z., SHERIN, S., et al. (2017). Empirical studies omit reporting necessary details: A systematic literature review of reporting quality in model based testing. Computer Standards & Interfaces.

JENSEN C, S.; PRASAD, M. R.; MOLLER, A. Automated testing with targeted event sequence generation. In: Proc. Of the 22nd International Symposium on Software Testing and Analysis (ISSTA), p. 66-77, 2013.

LECHETA R. R. Google Android – Aprenda a criar aplicações móveis para dispositivos móveis com o Android SDK. Novatec, 5ª edição. 2015.

LEGEARD Bruno (2010). Model-based Testing: Next Generation Functional Software Testing. In: Practical Software Testing: Tool Automation and Human Factors, 2010.

LINSCHULTE, M. (2013). On the Role of Test Sequence Length, Model Refinement, and Test Coverage for Reliability. Ph.d. dissertation, Universitat Paderborn, Germany.

MAREK, J., MIKA, K., and TUULA, P. Obstacles and opportunities in deploying model-based gui testing of mobile software: a survey. Software Testing, Verification and Reliability, 22(5):313–341.

MICSKEI, Z. Model-based testing (MBT), jan. 2016. Disponível em: <http://mit.bme.hu/~micskeiz/pages/modelbased_testing.html>. Acesso em: 14 set. 2017.

MUCCINI, H.; DI FRANCESCO, A.; ESPOSITO, P. Software testing of mobile applications: Challenges and future research directions. In: Proc. Of the 7th International Workshop on Automation of Software Test (AST), p. 29-35, 2012.

MYERS, Glenford. J.; SANDLER, Corey; BADGETT, Tom. The art of software testing. John Wiley & Sons. New Jersey, 2011.

OLIVEIRA M. Desenvolva aplicativos móveis com Android Studio, nov. 2016. Disponível em: <http://terminalroot.com.br/2016/11/blog-linux-android2.html >. Acesso em: 20 set. 2017.

ORSO, A.; ROTHERMEL, G. 2014. Software testing: a research travelogue (2000–2014). In Proceedings of the on Future of Software Engineering (FOSE 2014). ACM, New York, NY, USA, 117-132. DOI: http://dx.doi.org/10.1145/2593882.2593885

PRESSMAN, Roger S. Engenharia De Software. McGraw Hill, 7a EDIÇÃO. 2010.

SCHULZE, C., GANESAN, D., Lindvall, M., CLEAVELAND, R., and GOLDMAN, D. (2014). Assessing model-based testing: an empirical study conducted in industry. In Companion Proceedings of the 36th International Conference on Software Engineering, pages 135–144. ACM.

ROSENFELD, A.; KARDASHOV, O.; ZANG O. 2017. Automation of Android Applications Testing Using Machine Learning Activities Classification. ArXiv e-prints. DOI: https://arxiv.org/pdf/1709.00928.pdf

TAKALA T.; KATARA M.; HARTY J. Experiences of system-level model-based GUI testing of an Android application. In: Proc. of the 4th IEEE International Conference on Software Testing, Verification, and Validation (ICST), p. 377-386, 2011.

UTTING, Mark; PRETSCHNER, Alexander; LEGEARD, Bruno. A Taxonomy of Model-Based Testing. Technical Report, Hamilton, New Zealand, 2006.

AGRADECIMENTOS

A. T. Endo é parcialmente financiado pelo CNPq/Brasil (processo 445958/2014-6). H. N. Silva é bolsista PIBIC da UTFPR/Brasil.