Uma Proposta de Boas Práticas para Aplicação de Teste Baseado em Modelos em Métodos Ágeis

Aline Zanin[1], **Avelino Francisco Zorzo**[1], **Élder de Macedo Rodrigues**[2]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

[2]Universidade Federal do Pampa (Unipampa)

aline.zanin@acad.pucrs.br, avelino.zorzo@pucrs.br

elderrodrigues@unipampa.edu.br

Abstract. *This paper analyzes how Model-based Testing (MBT) can be applied by teams that use agile methods, more specifically Scrum. Hence, we performed a systematic literature review to identify the works that were published and show the problems related to MBT process when applied in the agile context. Furthermore, we performed also a field research conducted through interviews with professionals who work on software quality in agile teams. Finally, we correlated the opinion of the specialists against the literature review and provide a proposal of better practices for implementing MBT in agile teams. This proposal was validated through a survey.*

Resumo. *Este trabalho analisa como Teste Baseado em Modelos (Model-based Testing - MBT) pode ser aplicado em times que utilizam métodos ágeis, mais especificamente Scrum. Para isso, realizamos dois estudos: um mapeamento sistemático de literatura e uma pesquisa de campo. Os resultados do mapeamento sistemático foram utilizados para identificar os trabalhos já publicados que apresentam dificuldades encontradas no processo de MBT quando aplicado em projetos ágeis de desenvolvimento de software. A pesquisa de campo, por sua vez, foi realizada através de entrevistas realizadas com profissionais que trabalham focados em testes de software em equipes ágeis. Neste estudo objetivamos coletar as opiniões dos profissionais a respeito da utilização de MBT no contexto de suas equipes. Por fim, corroboramos o resultado obtido na entrevista com os especialistas e os resultados obtidos com o mapeamento sistemático e elaboramos uma proposta de boas práticas para implementação de MBT em equipes ágeis. Os resultados atuais, obtidos através de uma survey indicam que a proposta é adequada para o contexto a qual se propõe.*

1. Introdução

O desenvolvimento ágil de software teve sua popularização e ascensão no ano de 2001, através de um documento chamado manifesto ágil de software [3]. Neste documento estão descritos os princípios e valores que devem ser seguidos por empresas e profissionais que desejam adotar os métodos ágeis de desenvolvimento. Dentre estes princípios destacam-se: interação com o cliente acima de documentação e contratos fechados; boa receptividade à mudança de requisitos; multidisciplinaridade e responsabilidade coletiva nas equipes de desenvolvimento, integrando pessoas relacionadas a negócios e equipe de desenvolvimento para o trabalho conjunto diário.

Esta multidisciplinaridade das equipes exige uma responsabilidade coletiva sob todas as atividades de um projeto e neste contexto, um dos segmentos que empreendeu mudanças foi o segmento de testes de software. Se antes, na abordagem tradicional de desenvolvimento de sistemas, os profissionais de qualidade, embora envolvidos no processo de desenvolvimento de sistemas como um todo, tinham como atividade e responsabilidade principal a realização do controle de qualidade dos produtos e processos, agora, em métodos ágeis, cria-se uma cultura de responsabilidade coletiva. Nesta cultura, todos os profissionais são responsáveis pela excelência no desenvolvimento de todo o trabalho e devem contribuir da melhor forma possível para o bom andamento dos trabalhos da equipe e da comunicação entre os envolvidos no projeto e com o cliente. Desta forma estabelece-se uma única equipe multidisciplinar [9].

Contudo para que este trabalho multidisciplinar possa ser executado com qualidade se faz necessário utilizar técnicas de desenvolvimento e de testes que favoreçam a comunicação entre os membros do time de desenvolvimento. Para isso é necessário olhar para técnicas que já apresentam histórico de sucesso em sua aplicação em projetos que utilizam ciclo de vida tradicional de software, por exemplo, cascata ou espiral, e analisar a sua aplicabilidade em equipes ágeis. Dentre as diversas abordagens, técnicas e estratégias de testes apresentadas na literatura, a técnica de teste baseado em modelos (*Model Based Testing* - MBT [2]) é um exemplo de técnica que apresenta diversos benefícios, já relatados em trabalhos acadêmicos, e cuja aplicabilidade em equipes ágeis ainda é pouco explorada.

Embora MBT demande a construção de modelos, o que pode por vezes causar uma desconfiança em relação a aplicabilidade desta técnica em métodos ágeis, é importante enfatizar que a representação dos requisitos de software em modelos é uma estratégia para tornar os requisitos de software mais visuais e intuitivos [6] e desta forma, pode inclusive favorecer a comunicação entre os membros de um projeto e o cliente. Com isso, a documentação se torna de mais fácil compreensão quando comparada a longos documentos textuais. Além disso a representação de requisitos em modelos, permite ao cliente visualizar a estrutura de funcionamento da funcionalidade que está sendo projetada o que ajuda a evitar ruídos de comunicação entre os membros do projeto e o cliente. Partindo do princípio que a comunicação entre os envolvidos é um dos pilares do manifesto ágil [3], considera-se que a aplicação de MBT pode agregar valor a um cenário ágil, envolvendo as vantagens da utilização de modelos e da realização eficaz de testes de software.

Contudo, embora os benefícios desta técnica sejam conhecidos e existam indícios que sua aplicação em equipes ágeis pode proporcionar grandes benefícios ao aliar as vantagens proporcionadas pela técnica de MBT e as vantagens proporcionadas pela utilização de métodos ágeis, se faz necessário analisar a melhor forma de aplicar esta técnica neste contexto. Isto porque a estrutura de trabalho e principalmente a periodicidade de entregas de produtos difere no ciclo de vida tradicional e na utilização de métodos ágeis.

Até onde conseguimos identificar, não existem trabalhos científicos que apresentem boas práticas na aplicação de MBT em time ágeis. Desta forma, buscamos com este trabalho diminuir esta lacuna, respondendo a seguinte questão de pesquisa: Quais são as boas práticas que devem ser observadas para implantação de MBT em equipes ágeis? Com o objetivo de responder a esta pergunta, realizamos um mapeamento sistemático de literatura buscando trabalhos que relatam os desafios da realização de testes ou descrevem

a aplicação da técnica de MBT em times ágeis; paralelamente foram conduzidas entrevistas com profissionais que trabalham focados em testes de software e integram equipes ágeis. Nestas entrevistas buscamos apresentar a técnica de MBT e coletar a opinião destes profissionais sobre as dificuldades e vantagens de aplicar esta técnica no contexto dos projetos nos quais estão inseridos nas empresas em que atuam.

A partir dos resultados obtidos no mapeamento sistemático de literatura e das respostas coletadas nas entrevistas com os profissionais, identificamos as dificuldades e as vantagens da aplicação de MBT em métodos ágeis que foram comuns nos dois estudos. Baseado nisso, e na experiência que possuímos, por integrarmos um grupo de pesquisa que trabalha faz dez anos com MBT [20], [8], [19], [18], [4], [5], analisamos estes pontos afim de propor um conjunto de boas práticas na implantação de MBT em times de desenvolvimento ágil. Por fim, realizamos uma *survey* com um conjunto de 30 profissionais que atuam em equipes ágeis de desenvolvimento de sistemas para verificar se o conjunto de boas práticas sugeridas são coerentes e se elas podem auxiliar no processo de adoção de teste baseado em modelos por equipes ágeis.

Este artigo está organizado da seguinte maneira: na Seção 2 apresentamos a metodologia de pesquisa. Na Seção 3 apresentamos os resultados da revisão literatura. Na Seção 4, por sua vez, apresentamos a estrutura utilizada para as entrevistas com os especialistas. Na Seção apresentamos os problemas comuns encontrados na literatura e nas entrevistas. Na Seção 6 apresentamos o conjunto de boas práticas para implantação de MBT em equipes ágeis. Na Seção 7 apresentamos uma primeira avaliação sobre a coerência e efetividade das boas práticas sugeridas. Finalmente, apresentamos as considerações finais, agradecimentos e trabalhos futuros.

2. Metodologia e Desenvolvimento

O estudo apresentado neste artigo foi realizado em quatro etapas. Sendo elas: mapeamento de literatura, entrevistas com especialistas, formulação de uma proposta de boas práticas e validação dos resultados.

Na etapa de mapeamento de literatura, efetuou-se uma busca na base de dados *Scopus Database* visando localizar artigos que apresentam abordagens, técnicas, métodos, processos e relatos de aplicação da técnica de MBT em times ágeis. Para cada trabalho localizado nesta pesquisa buscou-se analisar as dificuldades na implantação de MBT em times ágeis descritas pelos autores. Optou-se por utilizara base de dados *Scopus Database* porque esta base concentra mais de 6,8 milhões de artigos e indexa diversas outras bases de dados, entre elas a IEEE[1] e a ACM[2] que estão entre as principais bases eletrônicas de dados para ciência da computação [15].

Na etapa de entrevistas com especialistas efetuou-se uma pesquisa de campo, com o objetivo de obter um entendimento sobre a visão dos profissionais que atuam com testes de software sobre a aplicação de MBT em times de desenvolvimento ágil. Para esta etapa do estudo, foi realizado um conjunto de seis entrevistas com profissionais que atuam como analistas de testes e qualidade em equipes ágeis de desenvolvimento de sistemas. Os profissionais entrevistados estão geograficamente distribuídos em diversos estados brasileiros

[1] http://ieeexplore.ieee.org/Xplore/home.jsp
[2] http://dl.acm.org/

(Rio Grande do Sul, Santa Catarina e Minas Gerais) e por este motivo as entrevistas[3] foram realizadas fazendo uso de um software que permite comunicação pela Internet através de conexões de voz e vídeo.

Na etapa de formulação de uma proposta de boas práticas na implantação de MBT em equipes ágeis, efetuou-se uma comparação dos resultados obtidos no mapeamento de literatura com os resultados obtidos nas entrevistas com os profissionais. A partir disso, definiu-se um conjunto de dificuldades que foram relatos comuns nos dois estudos. Com o objetivo de sanar estas dificuldades, propusemos para cada uma delas, uma ou mais práticas para serem utilizadas na implantação de MBT em equipe ágeis.

Por fim, na etapa de validação dos resultados, realizamos uma *survey* com um conjunto de 30 profissionais que trabalham com teste de software. Esta *survey* foi executada para verificar se o conjunto de boas práticas sugeridas são coerentes e se elas podem auxiliar no processo de uso de MBT com métodos ágeis. A Figura 1 demonstra graficamente as etapas deste processo de pesquisa que serão descritas detalhadamente nas seções seguintes.

Figura 1. Fluxograma do Processo de Pesquisa

3. Revisão de Literatura

Nesta seção são apresentados alguns trabalhos disponíveis na literatura que descrevem metodologias, técnicas, processos, abordagens, ferramentas ou relatos de experiência com a realização de MBT em equipes ágeis de desenvolvimento de sistemas e que em seu escopo apresentam as dificuldades encontradas neste processo. Esta pesquisa foi realizada através de uma busca efetuada na base de dados Scopus[4], fazendo uso das seguintes palavras chave: "TITLE-ABS-KEY ((mbt OR "model-based testing"OR "model based testing"OR "model-based test"OR "model based test"OR "model-based software testing"OR "model based software testing") AND (approach OR method OR methodology OR technique) AND (agility OR agile OR scrum OR xp OR "Extreme Programming"OR fdd OR "Feature-Driven Development"OR tdd OR "Test Driven Development"OR lean OR kanban) AND (software)) ".

Esta busca retornou 39 artigos, sendo que destes, foram incluídos nesta pesquisa apenas artigos escritos entre 2001 e 2016, disponibilizados na web nos idiomas inglês e português, que descrevem, de forma detalhada, metodologias, ferramentas, processos ou técnicas utilizadas para testes de software baseado em modelos em times ágeis, que apresentam, de forma detalhada, alguma metodologia, ferramenta, processo ou técnicas utilizadas especificamente para criação de modelos baseado em *Gherkin, Specification By Example ou User Story*.

A partir dos resultados retornados por esta busca foram analisados os trabalhos que apresentam desafios na aplicação de MBT em métodos ágeis, sendo selecionados os seguintes.

[3]As perguntas efetuadas neste estudo, podem ser visualizados em: https://goo.gl/UwQhJ1
[4]https://www.scopus.com

Aichernig *et. al* [1] apresentam um trabalho que descreve uma estratégia para realização de *Test Driven Development -TDD* baseado em modelos. Nesta estratégia, modelos formais são criados e com base nestes é realizada a geração de *scripts* de testes. A partir disso é realizada a validação destes *scripts* e posteriormente aplicado TDD em seu formato tradicional, implementando os casos de teste anteriormente validados. Neste trabalho os autores apresentam como dificuldade na aplicação de MBT a garantia da qualidade dos *scripts* de testes gerados. Esta preocupação é devido a realização dos testes ser feita baseando-se em *scripts* gerados com base nos modelos e sendo assim, modelos incorretamente criados, ou modelos aonde existem lacunas de requisitos não modelados, impactam diretamente na qualidade dos *scripts* de teste.

Katara *et. al* (2006)[14] apresentam um trabalho que descreve uma abordagem baseada na utilização de diagramas de casos de uso para automação de teste de software. Em relação às dificuldades encontradas para integração de MBT e métodos ágeis, os autores descrevem que os modelos gerados por equipes ágeis costumeiramente possuem um nível de abstração de detalhes elevado, o que impede ou dificulta aplicação de MBT. Esta dificuldade se justifica, uma vez que, demanda-se um tempo adicional para que o modelo criado com alto nível de abstração possa ser completado com as informações requeridas para aplicação de MBT. Além disso os autores enfatizam a dificuldade mensurar se os requisitos foram cobertos integralmente pelos modelos criados.

Entin *et al.* (2011)[10] apresentam uma abordagem para combinar as técnicas de Captura e Repetição (CR) e MBT. Esta abordagem consiste em criar um conjunto reutilizável de caminhos de teste, através de uma ferramenta de CR e posteriormente integrar estes caminhos a modelos comportamentais que serão a base para aplicação da técnica de MBT e geração de *Scripts* de teste. Neste trabalho os autores apresentam como dificuldades na utilização de MBT em projetos ágeis que utilizam o método Scrum, o fato de a grande maioria das abordagens de MBT não deixarem claro como é feito o processo de migração de um caso de teste abstrato (gerado através dos caminhos percorridos nos modelos) para um caso de teste executável. Este processo é, em muitos casos, realizado de forma manual, o que ocasiona um problema de consumo excessivo de tempo e recurso e por vezes torna inviável a realização de MBT em uma *sprint*.

Jalalina *et. al* (2012)[13] apresentam um conjunto de padrões utilizados em MBT que quando aplicados a projetos ágeis podem auxiliar a resolver problemas com deficiência de documentação e atualização de *scripts*. Contudo, o trabalho apresentado pelos autores não tem foco exclusivo em MBT e sim em resolver problemas encontrados em métodos ágeis, sendo para alguns deles aplicado MBT como solução. Entre as dificuldades descritas neste trabalho, sobre a integração de MBT com métodos ágeis, encontram-se: a falta de sistemas de controle de versão com suporte ao gerenciamento de modelos, a integração de modelos com IDEs populares de desenvolvimento de sistemas, e a inexistência de integração entre modelo e código,

Entin *et al.* (2012)[11] apresentam os desafios enfrentados na implantação e utilização de MBT em um projeto de desenvolvimento ágil que utiliza Scrum. Os autores focam seu trabalho na criação e manutenção de modelos, realização de teste baseado em riscos na interface gráfica de usuário (GUI), algoritmos de derivação de casos de teste e, escolha de arquitetura. Em relação aos desafios apresentados, os autores iniciam abordando o fato de que projetos ágeis têm ciclos de desenvolvimento curtos e, por isso,

faz-se necessária a definição de um processo concreto de criação e manutenção dos modelos durante todo o processo de desenvolvimento. Outro desafio apontado diz respeito à necessidade de atualização contínua dos modelos para a realização de testes de regressão, isto porque no contexto apresentado, as equipes costumam realizar estes testes continuamente para garantir que alterações efetuadas no código por um desenvolvedor, ao serem incorporadas no sistema (processo de *branch merges*), não interfiram no funcionamento do sistema como um todo. Os autores apresentam também como desafio a necessidade de prover alternativas para a definição correta do nível abstração dos modelos, afim de que os modelos criados possam ser reutilizáveis para novos testes. E por fim, o trabalho aponta como desafio a necessidade de tornar possível gerar casos de teste para os cenários considerados críticos, de alta prioridade, otimizando desta forma o tempo de execução de testes.

Sivanandan *et al.* (2014)[21] apresentam o projeto de um *framework* que utiliza MBT para automação de Teste Baseado em Comportamento (*Behaviour Driven Test* - BDT) e analisa como este *framework* pode ser usado durante o desenvolvimento ágil. Para conceber este projeto, os autores propõem a integração da ferramenta *Graphwalker*, do *framework Robot* e da utilização de BDT. Entretanto, os autores não são claros a respeito de como é feita a integração, não sendo possível identificar como foram concebidos os modelos e gerados os *scripts* de teste. Neste trabalho os autores apresentam como desafios para utilização de MBT em métodos ágeis: criar e manter modelos pouco rígidos e flexíveis, manter a conformidades do código com o modelo e garantir a cobertura de testes.

Entin *et al.* (2015)[12] apresentam os resultados da análise da aplicação da técnica de MBT em um projeto Scrum. O processo proposto, e utilizado neste trabalho, faz uso da linguagem de domínio específico *Gherkin* para a definição dos requisitos e realiza a geração automatizada de modelos a partir destes requisitos. O autor apresenta como desafios na utilização de MBT em times ágeis, os seguintes pontos: construir um modelo fiel aos requisitos, uma vez que os requisitos criados por equipes ágeis costumam ser bem sintetizados; falta de domínio de modelagem de sistemas pelos responsáveis pelo negócio em times Scrum (*Product Owner – PO*); reduzir o esforço de manutenção de modelos afim de mantê-los atualizados conforme os requisitos do sistema forem sendo alterados, isto porque é comum em métodos ágeis a mudança de requisitos durante o projeto; construção e alteração manual dos modelos.

Nan Li *et al* (2016) [16] propõem a utilização de modelos comportamentais de *Unified Modeling Language* - UML para geração de cenários no formato aceito pela ferramenta Cucumber. Após este processo, o programador poderá utilizar estes cenários para a criação de casos de teste automatizados concretos. Neste trabalho os autores apresentam como principal dificuldade para utilização de MBT em equipes ágeis o fato de que os desenvolvedores costumam criar modelos comportamentais, contudo não costumam evolui-los para realização de testes. Uma das justificativas para isso é que estes modelos não são compatíveis com as ferramentas tradicionalmente utilizadas pelas equipes ágeis, *e.g.* Cucumber. Além disso, os autores apontam como dificuldade a complexidade da criação de modelos devido a quantidade de informações detalhadas que são necessárias inserir nestes para aplicar MBT.

4. Entrevistas com Especialistas

Com o objetivo de obter informações sobre a aplicação de MBT em equipes ágeis, realizamos um conjunto de entrevistas semi estruturadas com especialistas em teste de software que trabalham em times ágeis. Primeiramente foi apresentado a estes profissionais a técnica de MBT, para isso explicou-se como ela funciona e quais são as principais vantagens e desvantagens de sua utilização quando aplicada em uma estrutura de ciclo de vida tradicional de software. Posteriormente, ainda com objetivo de familiarizar os entrevistados com a técnica de MBT, realizou-se uma demonstração de um exemplo de aplicação prática da técnica.

Para a demonstração da técnica de MBT foi utilizada uma ferramenta de teste funcional baseado em modelos, desenvolvida no Centro de Pesquisas em Engenharia de Software da PUCRS [20], como parte de uma linha de produto de ferramentas de testes baseado em modelos. E como ferramenta a ser testada utilizou-se a ferramenta de gestão de projetos EPESI [5]. Esta ferramenta foi modelada e casos de teste foram gerados a partir dos modelos.

Posteriormente, estes profissionais responderam perguntas sobre como esta técnica poderia ser utilizada nos projetos em que eles trabalham, considerando uma estrutura ágil de desenvolvimento de sistemas. Cada profissional respondeu as perguntas em um momento diferente. As perguntas versaram sobre as dificuldades, vantagens, desvantagens e sugestões de possíveis melhorias para a inserção e utilização de MBT neste cenário.

Analisando as respostas das entrevistas, pudemos identificar pontos que são apontados pelos especialistas como as principais dificuldades para implantação de MBT nos times aonde estão inseridos. Neste contexto, os especialistas entrevistados são unânimes na percepção de que a maior dificuldade na aplicação da técnica de MBT está na etapa de construção dos modelos. Esta dificuldade se deve ao fato dos modelos precisarem ser construídos com base nas informações de negócio que as equipes possuem, sendo que, nem sempre estas informações são representativas suficientes para criação dos modelos. Esta dificuldade é comum nas equipes onde os entrevistados estão inseridos, mesmo considerando o fato de que cada equipe recebe os requisitos de uma forma, variando entre requisitos textuais, *mokups* de telas, histórias de usuário ou fluxogramas.

Além da falta e informações para a criação dos modelos, outro ponto para o qual deve ser dada atenção no processo de MBT é a gestão do tempo. De acordo com os entrevistados, devido ao fato das equipes ágeis, em sua maioria, trabalharem com ciclos curtos de entregas, o tempo disponível para análise, desenvolvimento e testes do produto que será entregue ao cliente é reduzido. Desta forma, se algum dos profissionais, ou até mesmo se toda equipe, dedicar seu tempo para modelar todo o sistema para aplicação de MBT, podem, nestes ciclos de desenvolvimento, não restar tempo hábil para execução de todo processo de desenvolvimento e de testes do produto.

Para reduzir este problema, uma solução plausível é a construção colaborativa dos modelos. Desta forma, os modelos devem ter sua criação iniciada no início do processo de desenvolvimento, ou seja, quando os requisitos são levantados junto ao cliente. Neste contexto, a modelagem iniciaria sendo feita pelo profissional que possui relação direta

[5] http://epe.si/

com o cliente e posteriormente seria atualizada por todo time de desenvolvimento, tendo as informações de teste inseridas pelos profissionais de qualidade. Contudo, é importante enfatizar que projetos que fazem uso de métodos ágeis tem em sua cultura a boa recepção a mudança de requisitos, por este motivo, um dos entrevistados aponta que uma dificuldade que pode ser encontrada no momento da aplicação de MBT em equipes ágeis é encontrar tempo e recursos humanos para fazer contínuas atualizações nos modelos.

Para que este fluxo de construção e atualização contínua dos modelos obtenha êxito, se faz necessário o envolvimento de toda a equipe no processo e isto faz com que surja outro desafio: conscientizar as equipes de desenvolvimento da importância de MBT e dos benefícios que ele pode trazer, facilitando o trabalho da equipe e melhorando a qualidade dos produtos entregues. Para isso, se faz necessário utilizar estratégias de desenvolvimento de equipes para prover a capacitação e fomentar a motivação da equipe de desenvolvimento para o emprego desta técnica. Isto pode ser considerado um desafio porque, devido ao princípio ágil de utilizar documentação enxuta, alguns projetos optam pela não utilização de modelos para representação de requisitos, o que faz com as equipes de desenvolvimento não estejam familiarizadas com linguagens de modelagem (*e.g.* UML).

Outro ponto de dificuldade relatado de forma unânime pelos especialistas diz respeito a falta de integração da ferramenta de MBT com as ferramentas de desenvolvimento de sistemas. Os especialistas entrevistados sugerem que além de gerar *scripts* automatizados, a ferramenta que realiza MBT deve ser integrada com as IDEs tradicionais de desenvolvimento de sistemas (*e.g.* Eclipse e Visual Studio). Além disso, os *scripts* devem ser gerados de forma automatizada e utilizando a estrutura padrão de alguma ferramenta de automação de testes (*e.g.* Seleniun WebDriver), para que a manutenção seja facilitada e as habilidades dos profissionais valorizadas.

Após a geração dos *scripts* de testes, em virtude da técnica de MBT gerar *scripts* para todos os caminhos de teste compreendidos pela funcionalidade modelada, se faz necessário utilizar, de forma conjunta, outras técnicas para viabilizar a execução de todos os testes. Estas técnicas devem ser aplicadas para reduzir o número de entradas as quais o sistema precisa ser submetido para ser testado. São exemplos de técnicas que podem ser empregadas *Pairwise test* (que foi citada por um dos especialistas entrevistados), partição e equivalência e análise de valor limite.

Juntamente com a redução do número de entradas as quais o sistema precisa ser submetido, é necessário que os *scripts* gerados possuam alguma forma de priorização, podendo assim evitar que sejam realizados testes de prioridade menor quando não existe tempo disponível para execução de todos os casos de teste gerados.

5. Definição e Descrição dos Problemas Comuns encontrados

Nesta seção apresentamos a definição e a descrição dos problemas que são comuns na literatura e nas entrevistas com os especialistas.

I. Ausência de tempo para criação de modelos: devido aos ciclos de desenvolvimento serem bastante curtos, em times de desenvolvimento ágil, se faz necessário prover mecanismos para que a criação dos modelos seja feita durante todo o ciclo de desenvolvimento e não apenas no intervalo de tempo destinado ao teste de software, conforme relatado por Entin *et al.* (2012)[11]. Isto é, quando for realizado o primeiro contato

com o cliente para levantamento de requisitos e/ou escrita de histórias de usuário já deve ser iniciada a criação dos modelos. Os modelos produzidos devem ser adaptados durante todo o processo de desenvolvimento. Alternativamente os modelos podem ser construídos automaticamente, baseados na especificação de requisitos e/ou nas histórias de usuário.

II. Falta de informações para criação dos modelos: uma dificuldade recorrente é que as documentações existentes em projetos ágeis não são suficientes para a concepção dos modelos para realização de MBT. Por este motivo embora algumas equipes costumem efetuar a criação de modelos, estes não costumam ser ricos em detalhes, conforme descrito por Katara *et. al* (2006)[14], e tendem a ser construídos baseados no conhecimento de negócio prévio do testador oque os torna são passíveis de falhas que impactam diretamente na qualidade dos testes. Além disso, devido a cultura de utilizar menos documentação e mais contato pessoal, muitas informações podem ser discutidas em reuniões e não documentadas o que também impacta na qualidade dos modelos.

III. Necessidade de dinamismo para adaptar as mudanças: um princípio básico dos métodos ágeis é que as equipes devem estar abertas para receber as mudanças de requisitos solicitadas pelos clientes. Neste contexto é necessário que a técnica de testes utilizada propicie a alteração contínua dos artefatos para suprir esta demanda. MBT pode auxiliar as equipes de desenvolvimento neste sentido, pois possibilita a automação do sincronismo das alterações realizadas nos modelos com a atualização dos artefatos de testes. Entretanto, é necessário um mapeamento eficiente entre modelo e requisito para que, quando ocorrerem mudanças de requisitos, possa ser alterado apenas a parte do modelo que é impactada pela mudança sem necessidade de refazer completamente a modelagem.

IV. Dificuldade na criação manual dos Modelos e execução manual de casos de teste: uma dificuldade inerente ao processo de MBT é a criação manual dos modelos que são a base da aplicação da técnica. Esta dificuldade acontece por vários motivos já descritos (*e.g.* tempo, recurso, conhecimento de modelagem). Desta forma para uma melhor eficácia de MBT de faz necessário encontrar técnicas para a geração automatizada destes modelos. Além disso, indo ao encontro a uma prática comum em métodos ágeis, a técnica de MBT precisa gerar *scripts* automatizados de testes e não casos de testes para execução manual.

V. Necessidade de adoção de técnicas de priorização para a seleção dos *scripts* e casos de teste de maior relevância: Este desafio surge, uma vez que, os métodos de geração de sequências de testes, costumeiramente utilizados pelas ferramentas de MBT (*e.g. W [7] e Harmonized State Identification (HSI) [17]*), efetuam a geração de sequências que cobrem todas as funcionalidades do sistema em teste, porém, embora estes métodos busquem selecionar os melhores caminhos para minimizar os testes e ainda garantir a cobertura, o número de *scripts* ou casos de teste gerados pode a ser elevado. Neste caso, quando o ciclo de entrega do projeto é curto, se faz necessário priorizar os testes, garantindo que as funcionalidades prioritárias sejam testadas antes das outras. Outro cenário em que a priorização ou seleção dos testes que serão executados prioritariamente se faz necessário é em testes de regressão, aonde o testador não deseja testar toda a funcionalidade e sim apenas uma fração dela, aonde foram feitas alterações.

VI. Necessidade de adoção de conjunta de MBT com outras técnicas de teste: Esta dificuldade, assim como a anterior, está relacionada a redução da quantidade de testes

que precisam ser feitos para validar um sistema. Em alguns cenários o número de testes é elevado devido a quantidade de entradas a qual o sistema pode ser submetido. Desta forma, utilizar técnicas de seleção de entradas como por exemplo partição e equivalência, análise de valor limite e *pairwise test* se tornam de grande importância para garantir a cobertura reduzir a quantidade de testes.

Baseado na problemática apontada pela literatura, considerando a análise das respostas obtidas nas entrevistas com os especialistas e a experiência que adquirimos nos últimos anos com aplicação de MBT, realizamos a concepção de um conjunto de boas práticas para a implantação de MBT em times ágeis que será apresentado na próxima seção.

6. Proposta de Boas Práticas

Para que possamos conceber este conjunto de boas práticas para implantação de MBT em Métodos Ágeis, primeiramente identificamos os principais problemas e desafios encontrados pelas equipes de desenvolvimento. Posteriormente, para cada problema identificado, propomos uma ou mais estratégias de solução através de boas práticas. Consideramos que ao seguir este conjunto de boas práticas, as equipes de desenvolvimento obterão maior eficácia na implantação de MBT, e por este motivo poderão usufruir melhor das vantagens que esta técnica proporciona e consequentemente agregar qualidade ao produto entregue ao cliente. Estes resultados serão atingidos através da prevenção de erros e problemas comuns nos trabalhos analisados na literatura e nos relatos dos entrevistados. A partir dos desafios citados e visando a melhor integração da técnica de MBT com métodos ágeis, elencamos as seguintes boas práticas para implantação de MBT em métodos ágeis:

1. Conscientize a equipe da importância da criação de modelos e dê ciência dos benefícios da utilização de MBT:
    - Apresente resultados positivos reais de equipes que utilizam modelos para guiar o processo de desenvolvimento.
    - Fale sobre a técnica de MBT, apresente as vantagens, e permita a equipe que sane suas dúvidas de forma a agregar confiança na técnica.
2. Proporcione para toda a equipe de desenvolvimento capacitação para criação dos modelos.
3. Conscientize a equipe de desenvolvimento sobre a importância do trabalho em equipe e especialmente da importância dos testes de software.
4. Inicie o processo de criação de modelos juntamente com a escrita das histórias de usuário:
    - Quando são escritas as histórias de usuário pelo *Product Owner*, o mesmo efetua a criação do primeiro modelo, a exemplo do proposto por [12].
    - Os modelos criados pelo *Product Owner* deverão ser evoluídos pela equipe de desenvolvimento. Esta evolução deverá ser feita sempre que forem alterados requisitos, seja esta alteração solicitada formalmente ou definida em reunião de equipe. Esta etapa visa solucionar a questão aponta na entrevista referente a falta de controle dos requisitos definidos em reuniões.
    - No momento da realização dos testes de software, que pode ser paralelo ao desenvolvimento da funcionalidade, devem ser revisados e otimizados os modelos e realizada a geração dos artefatos de testes.

Esta proposta busca auxiliar resolver os problemas I, II e II, descrito na Seção 5, pelos seguintes motivos: com a criação dos modelos no início do processo de desenvolvimento, ainda em contato com o cliente, as equipes ganharão maior tempo para construir os modelos e maior riqueza de detalhes dada a presença do cliente para sanar dúvidas. Além disso, com os modelos sendo construídos junto com o software estes poderão ser mais facilmente adaptados quando necessário por mudança de requisito ou manutenção.

5. Priorize a geração automatizada de modelos e *scripts* evitando a criação manual de modelos e de casos de teste. Esta prática visa auxiliar na solução dos problemas I e IV, uma vez que gerando automaticamente os modelos e os demais artefatos de testes otimiza-se o tempo de trabalho da equipe
6. Inclua, no processo de MBT, uma forma de realizar a priorização dos *scripts* de testes gerados, de forma que o responsável pelos testes possa definir qual parte do sistema em teste deseja testar prioritariamente. Esta definição pode ser feita através de uma parametrização na ferramenta de MBT que permita definir quais atividades de um determinado modelo devem obrigatoriamente ser cobertas pelos testes. Esta prática visa atender ao desafio número V, uma vez que visa solucionar o problema de priorização dos casos de teste.
7. Busque alinhar a atividade de testes de software com os princípios do manifesto ágil, evitando que as equipes se segmentem por visualizarem a atividade de testes como um processo separado das demais atividades da equipe de desenvolvimento. Esta prática visa ajudar na solução de todos os desafios encontrados, uma vez que quando a equipe estiver alinhada aos princípios e valores ágeis, o time deverá trabalhar de forma mais integrada afim de obter melhores resultados e se adequar melhor com as práticas supra citadas.

7. Avaliação da Proposta de Boas Práticas

Com objetivo de realizar uma validação inicial da nossa proposta de boas práticas, preparamos uma *survey* com profissionais que trabalham em equipes de desenvolvimento de sistemas para identificar qual a visão deles em relação às boas práticas propostas. Optamos pela realização desta *survey* porque acreditamos que as pessoas que estão diariamente trabalhando na indústria são as pessoas que melhor podem descrever como acontecem os processos cotidianos e que medidas podem ajudar ou não o seu trabalho.

Esta *survey* foi escrita em estrutura de questionário e distribuída de forma virtual. O questionário foi formado por perguntas para identificação do perfil do respondente e perguntas relacionadas diretamente com o processo de teste baseado em modelos e as sugestões de melhorias propostas [6]. Obtivemos 30 respostas para esta *survey*, sendo que 34,6% dos respondentes trabalham com testes a mais de 6 anos, 27,6 % entre 4 e 6 anos, 24,1% de 2 a 4 anos e 13,8% até dois anos. Analisando as respostas obtidas podemos constatar que obtivemos um retorno positivo em relação a proposta apresentada, sendo que, a síntese das respostas obtidas será apresentada na sequência.

As propostas de melhoria número um e dois foram validadas da seguinte maneira: questionamos primeiramente se o respondente acreditava que caso uma equipe de desenvolvimento ágil seja conscientizada da aplicabilidade e benefícios de utilizar MBT, a

[6]Este questionário pode ser visualizado na íntegra em https://goo.gl/iobm2e

equipe poderia motivar-se para aprender e explorar a técnica, e o resultado foi que 84% dos entrevistados acreditam que sim. Além disso, perguntamos também se os respondentes acreditavam que a realização de capacitações para a criação de modelos auxilia ou viabiliza a criação de modelos durante o processo de desenvolvimento e testes de software. Esta pergunta teve 92% de respostas positivas. O conjunto destas duas respostas nos leva a crer que as boas práticas propostas números um e dois estão condizentes com a realidade das equipes.

A proposta de boas práticas número três, que diz respeito ao trabalho colaborativo foi avaliada através da seguinte pergunta: Conscientizar a equipe da importância do trabalho coletivo, e da importância da realização de testes no processo de desenvolvimento de sistemas pode motivar os profissionais a envolver-se com a criação e edição de modelos que representem o funcionamento esperado para o software? Obtivemos, para esta pergunta 96% de respostas favoráveis e 4% contrários. O que nos leva a crer que os profissionais que foram questionados estão alinhados com a nossa proposta de que a conscientização da equipe auxilia o processo de implantação de MBT.

A proposta de boas práticas número quatro, que diz respeito a iniciar a criação dos modelos juntamente com a escrita das histórias de usuários, juntamente com as suas sub propostas, foi avaliada da seguinte forma: primeiramente questionamos os profissionais se as equipes em que eles atuam costumam construir modelos comportamentais. Para esta questão, obtivemos uma resposta apontando que 55,2% das equipes de desenvolvimento efetuam, durante o processo de desenvolvimento, a construção de algum modelo comportamental. Posteriormente, questionamos quem era responsável pela construção do modelo e nesta resposta os mais diversos papéis foram citados. Estes dados podem ser vistos na Figura 2, no gráfico "Construção de Algum Modelo Comportamental". Mais especificamente 25% responderam que os modelos são construídos pelos desenvolvedores; 20,2% construídos pelo *Project Owner* (PO); 8,3% construídos pelo *Scrum Master*; 8,3% construídos pelo cliente. Além disso 37,5% responderam que os modelos são construídos por outros papéis não listados, *e.g.* testador, analista de testes e analista de sistemas. Estes dados podem ser vistos na Figura 2, no gráfico "Quem Constrói os Modelos".

A diversidade na responsabilidade da construção dos modelos é um indício que este trabalho pode ser dividido entre todos os profissionais e todo o processo de desenvolvimento de sistemas, sem sobrecarregar apenas a etapa de testes de software. Isto nos leva a crer que a proposta de boas práticas descrita no item quatro, está condizente com a realidade. Além disso, questionamos os profissionais referente a viabilidade do PO ficar responsável pela criação inicial do modelo que representa o fluxo principal do software. Obtivemos uma aceitação de 86,2% (Figura 2, no gráfico Criação do Modelo Inicial pelo PO) o que vem ao encontro ao proposto na proposta de melhoria número quatro.

A proposta de boas práticas número cinco foi validada através de duas perguntas. A primeira: A geração automatizada de *scripts* para automação de testes baseado nos modelos, pode apresentar vantagens quando aplicada em equipes ágeis? E a segunda: Uma vez criados os modelos iniciais pelo PO, se você fosse desafiado a realizar atualizações nos modelos, de acordo com a evolução do desenvolvimento, para posteriormente gerar automaticamente os artefatos de teste, qual seria a sua opinião? Para a primeira pergunta obtivemos uma aceitação de 96,6% (Figura 2 - Vantagem da Geração Automatizada) e para a segunda, 82,8% dos respondentes declararam que seriam favoráveis, 13,8% fa-

voráveis com ressalvas e apenas 3,4% contrários (Figura 2 - “Realizar Atualização dos Modelos”).

Na sequência afim de validar a proposta de boas práticas número seis, que diz respeito a gerar *scripts* de teste com prioridades distintas, ou então permitir a seleção da fração da funcionalidade que deseja testar, para que não sejam gerados *scripts* que representem toda a funcionalidade, quando deseja-se testar um fluxo específico, realizamos a seguinte pergunta: Considerando um diagrama de atividades, ou outro diagrama que represente o fluxo funcional do software de forma similar, você considera importante que possam ser gerados *scripts*, que representem apenas as funcionalidades pré-selecionadas do software representado no modelo? Obtivemos nesta pergunta 75,9% de resposta favorável e 24,1% contrária. Estes dados podem ser vistos na Figura 2, no gráfico “Geração de *Scrips* por Funcionalidade”.

Analisando as respostas de todas as perguntas podemos constatar que com a aplicação destas boas práticas, pode-se propiciar uma facilitação da implantação de MBT em métodos ágeis, uma vez que as práticas propostas estão alinhadas com a visão dos entrevistados e resolvem lacunas específicas que foram comuns na literatura e na entrevista com profissionais especialistas na área.

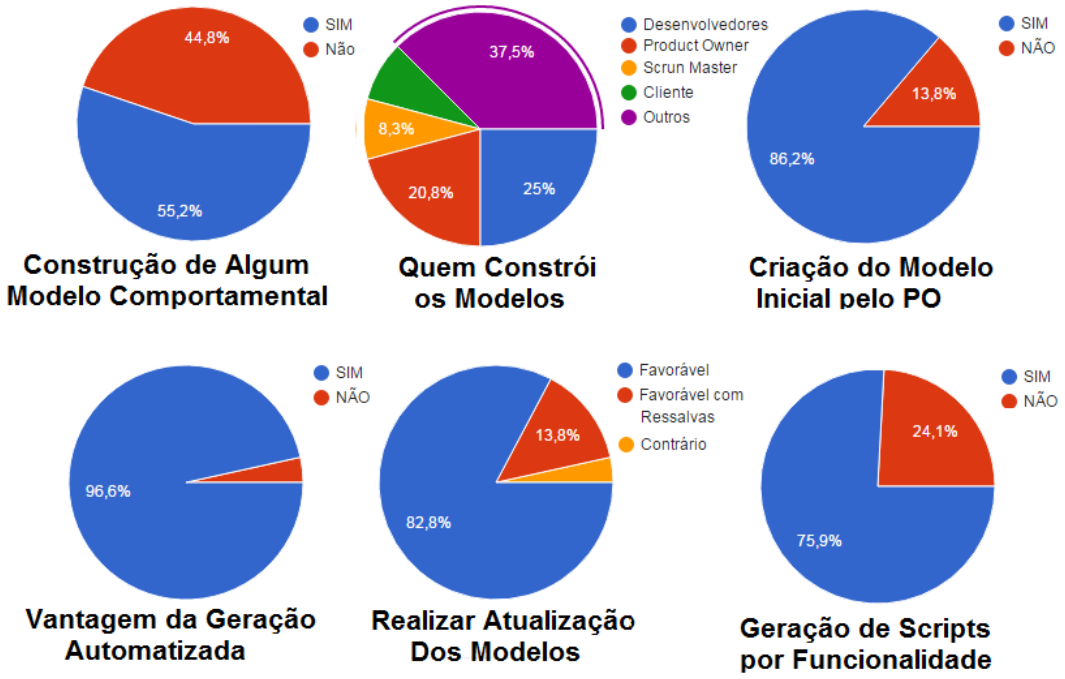

Figura 2. Gráficos contendo os resultados da Pesquisa

8. Considerações Finais e Trabalhos Futuros

Apresentamos neste trabalho uma proposta de boas práticas para aplicação de MBT em equipes de desenvolvimento ágil. Cada uma das boas práticas sugeridas, visa minimizar um problema identificado como comum na revisão de literatura e relatado por profissionais da área em entrevistas. Uma primeira avaliação das boas práticas sugeridas foi

realizada com 30 profissionais da área de teste que responderam a uma *survey*. Esta avaliação inicial indica que as boas práticas sugeridas são coerentes e relevantes dada a boa aceitação demonstrada pelos resultados deste estudo.

Contudo, embora os resultados preliminares apontem uma boa viabilidade e benefícios na aplicação desta proposta de boas práticas, ainda existem diversos pontos para serem explorados afim de evoluir este trabalho. Entre estes, objetivamos realizar um estudo de caso, em um ambiente industrial, para avaliar se as boas práticas propostas neste trabalho apresentaram benefícios reais para as equipes de desenvolvimento e teste. Assim iremos mitigar um das principais ameaça a validade do estudo apresentado, que é o fato do mesmo não ter sido aplicado ainda, de forma prática, em alguma empresa que trabalha com métodos ágeis. Como resultado teremos como avaliar a nossa proposta com base nas experiências reais de profissionais que a aplicaram em um ambiente real de desenvolvimento. Ainda, almejamos evoluir este conjunto de boas práticas para um *framework* de suporte a implantação e realização de testes baseados em modelos em times de desenvolvimento ágil.

Referências

[1] B. K. Aichernig, F. Lorber, and S. Tiran. Formal test-driven development with verified test cases. In *Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on*, 2014.

[2] L. Apfelbaum and J. Doyle. Model based testing. In *Software Quality Week Conference*, 1997.

[3] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, et al. Manifesto for agile software development. http://www.agilemanifesto.org, 2001. acessado em 07/03/2017.

[4] M. Bernardino, E. M. Rodrigues, and A. F. Zorzo. Performance testing modeling: an empirical evaluation of DSL and UML-based approaches. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, 2016.

[5] M. Bernardino, A. F. Zorzo, and E. M. Rodrigues. Canopus: A domain-specific language for modeling performance testing. In *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2016.

[6] D. A. Carlson. Designing xml vocabularies with uml (poster session). In *Addendum to the 2000 Proceedings of the Conference on Object-oriented Programming, Systems, Languages, and Applications (Addendum)*, 2000.

[7] T. S. Chow. Testing software design modeled by finite-state machines. *IEEE Transactions on Software Engineering*, 4(3), 1978.

[8] L. T. Costa, R. M. Czekster, F. M. de Oliveira, E. M. Rodrigues, M. B. da Silveira, and A. F. Zorzo. Generating performance test scripts and scenarios based on abstract intermediate models. In *Proceedings of the 24nd International Conference on Software Engineering and Knowledge Engineering, 2010*, 2012.

[9] L. Crispin and J. Gregory. *Agile testing: A practical guide for testers and agile teams*. Pearson Education, 2009.

[10] V. Entin, M. Winder, B. Zhang, and S. Christmann. Combining model-based and capture-replay testing techniques of graphical user interfaces: An industrial approach. In *Proceedings of the IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops*, 2011.

[11] V. Entin, M. Winder, B. Zhang, and S. Christmann. Introducing model-based testing in an industrial Scrum project. In *Proceedings of the 7th International Workshop on Automation of Software Test*, 2012.

[12] V. Entin, M. Winder, B. Zhang, and A. Claus. A process to increase the model quality in the context of model-based testing. In *Proceedings of the IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2015.

[13] D. Jalalinasab and R. Ramsin. Towards model-based testing patterns for enhancing agile methodologies. In *SoMeT*, 2012.

[14] M. Katara and A. Kervinen. Making model-based testing more agile: a use case driven approach. In *Proceedings of the Haifa Verification Conference*, 2006.

[15] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman. Systematic literature reviews in software engineering–a systematic literature review. *Information and software technology*, 51(1), 2009.

[16] N. Li, A. Escalona, and T. Kamal. Skyfire: Model-based testing with cucumber. In *Proceedings of the IEEE International Conference on Software Testing, Verification and Validation, ICST 2016*, 2016.

[17] G. Luo, A. Petrenko, and G. v. Bochmann. Selecting test sequences for partially-specified nondeterministic finite state machines. In *Protocol Test Systems*. 1995.

[18] E. M. Rodrigues, F. M. de Oliveira, L. T. Costa, M. Bernardino, A. F. Zorzo, S. d. R. S. Souza, and R. Saad. An empirical comparison of model-based and capture and replay approaches for performance testing. *Empirical Software Engineering*, 20(6), 2015.

[19] E. M. Rodrigues, R. S. Saad, F. M. Oliveira, L. T. Costa, M. Bernardino, and A. F. Zorzo. Evaluating capture and replay and model-based performance testing tools: An empirical comparison. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement- ESEM*, 2014.

[20] E. M. Rodrigues, L. D. Viccari, A. F. Zorzo, and I. Gimenes. PLeTs - Test automation using software product lines and model based testing. In *Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering, 2010*, 2010.

[21] S. Sivanandan and Y. C. B. Agile development cycle: Approach to design an effective model based testing with behaviour driven automation framework. In *Proceedings of the 20th Annual International Conference on Advanced Computing and Communications*, 2014.