GIOVANNI VICENTE GENTILE DA SILVA

Utilização de *Behaviour-Driven Development* para apoio à Gestão das Comunicações

São Paulo

2021

GIOVANNI VICENTE GENTILE DA SILVA

Utilização de *Behaviour-Driven Development* para apoio à Gestão das Comunicações

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para a conclusão do curso de MBA em Tecnologia de Software.

Área de Concentração: Tecnologia de Software

Orientador: Profª. Drª. Jussara Pimenta Matos

São Paulo

2021

[verso da folha de rosto]

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

FICHA CATALOGRÁFICA

[Colocar na versão final do trabalho. Obter em: <https://www.poli.usp.br/bibliotecas/servicos/catalogacao-na-publicacao>]

Nome: [SOBRENOME, Nome do aluno]

Título: [Título da monografia]

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para a conclusão do curso de MBA em Tecnologia de Software.

Aprovado em:

Banca Examinadora

Prof(a). Dr(a). \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instituição: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Julgamento: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof(a). Dr(a). \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instituição: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Julgamento: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof(a). Dr(a). \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instituição: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Julgamento: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DEDICATÓRIA

*À memória de meus avós Giovanni e Isabella.*

AGRADECIMENTOS

À Universidade de São Paulo – USP que [bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla].

À Escola Politécnica da Universidade de São Paulo – EPUSP que [bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla].

Ao PECE – Programa de Educação Continuada em Engenharia que [bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla]

Aos meus pais que [bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla].

RESUMO

[SOBRENOME, Nome abreviado]. [Título da monografia]. [Ano]. [Número de folhas]. Monografia (MBA em Tecnologia de Software). Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo. São Paulo. [Ano].

[Deve conter motivações, objetivos, métodos empregados, resultados e conclusões em 1 parágrafo]

Palavras-chave: [bla, bla, bla, bla, bla]

ABSTRACT

[SOBRENOME, Nome abreviado]. [Título da monografia]. [Ano]. [Número de folhas]. Monografia (MBA em Tecnologia de Software). Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo. São Paulo. [Ano].

[Should have motivations, objectives, applied methods, results and conclusions in 1 paragraph]

Keywords: [bla, bla, bla, bla, bla]

LISTA DE ILUSTRAÇÕES

Figura 1 – Título da figura .................................................................……………… xx

Figura 2 – Título da figura .................................................................……………… xx

Figura 3 – Título da figura .................................................................……………… xx

LISTA DE TABELAS

Tabela 1 – Título da tabela ..................................................................................... xx

Tabela 2 – Título da tabela ..................................................................................... xx

Tabela 3 – Título da tabela ..................................................................................... xx

LISTA DE ABREVIATURAS E SIGLAS

[BLA Bla bla bla bla bla bla bla bla bla bla bla bla bla]

[BLA Bla bla bla bla bla bla bla bla bla bla bla bla bla]

SUMÁRIO

1. INTRODUÇÃO ............................................................................................... xx

1.1 Motivações........................................................................................................ xx

1.2 Objetivo ............................................................................................................ xx

1.3 Justificativas .................................................................................................... xx

1.4 Método de Pesquisa......................................................................................... xx

1.5 Estrutura do Trabalho ....................................................................................... xx

2. REVISÃO BIBLIOGRÁFICA... .......................................................................... xx

2.x Título da Seção................................................................................................. xx

2.y Considerações do Capítulo .............................................................................. xx

3. DESENVOLVIMENTO………. ........................................................................... xx

3.x Título da Seção………………. ......................................................................... xx

3.y Considerações do Capítulo ............................................................................. xx

4 ANÁLISE DE RESULTADOS ............................................................................ xx

4.x Título da Seção………………. ......................................................................... xx

4.y Considerações do Capítulo .............................................................................. xx

5. CONSIDERAÇÕES FINAIS................................................................................ xx

5.1 Conclusões....................................................................................................... xx

5.2 Contribuições do Trabalho .............................................................................. xx

5.3 Trabalhos Futuros ........................................................................................... xx

REFERÊNCIAS ..................................................................................................... xx

GLOSSÁRIO ......................................................................................................... xx

APÊNDICE A – [Título do Apêndice]............. ........................................................ xx

ANEXO 1 – [Título do Anexo] .............................................................................. xx

1. INTRODUÇÃO

Este capítulo apresenta as motivações, objetivo, justificativas, método de pesquisa e a estrutura do trabalho.

* 1. Motivações

No contexto de transformação digital que o mercado de tecnologia está vivenciando, a busca pela excelência na entrega dos softwares com qualidade (relizando-a dentro do prazo e custo, além de gerar valor dentro das expectativas do negócio), com centralidade no cliente e adaptabilidade a mudanças se faz cada vez mais necessária, e é preciso adotar técnicas e práticas para atingimento destes objetivos.

Dentre os fatores que podem influenciar de forma desfavorável os resultados de um projeto e o atingimento dos objetivos citados no parágrafo anterior, estão problemas no entendimento dos requisitos e ruídos nas comunicações.

Estes problemas não são novos em desenvolvimento de software, inclusive foram citados por Frederick Brooks em seu livro *The Mythical Man-Month* (BROOKS, FREDERICK P., 1975), que possui um capítulo destacando que adicionar mais pessoas a um projeto atrasado tende a atrasá-lo ainda mais pois, dentre outros fatores, adicionam novos canais de comunicação que se não forem gerenciados podem implicar em mais problemas do que soluções.

Com o advento das metodologias ágeis, a importância de realizar as comunicações de forma efetiva foi muito enfatizada, porém estudos como os realizados pelo (PMI, 2013) e (VERNER; SAMPSON; CERPA, 2008), corroboram que ainda temos oportunidades de melhoria e a discussão deste assunto segue relevante, ainda mais no contexto que estamos inseridos, no qual a dependência das organizações em seus sistemas de informação bem como a competitividade neste meio estão cada vez maiores e uma boa comunicação é fator decisivo para conclusão bem-sucedida de um projeto.

Diante deste cenário, existem duas práticas que podem ser utilizadas para mitigar os impactos causados por ruídos de comunicação e problemas no refinamento de requisitos: O BDD (*Behaviour-Driven Development*), e a Gestão das Comunicações proposta pelo PMBOK®.

*Behaviour-Driven Development* consiste em um conjunto de práticas, tipicamente aplicadas em um contexto de desenvolvimento ágil, pois foi oriundo do *Test-Driven Development* que é um conceito introduzido pelos métodos ágeis (OLIVEIRA, 2015), que procura criar uma linguagem ubíqua (ou seja, todos os envolvidos pronunciam, referenciam e têm interpretação idêntica dos conceitos tratados) e facilitar o entendimento de quais critérios precisam ser satisfeitos para atingimento dos objetivos do projeto, visto que busca descrever o comportamento esperado do *software* (DAN NORTH & ASSOCIATES, 2006) aumentando a eficiência das comunicações, principalmente entre os times técnicos e times que representam o negócio, bem como a probabilidade de sucesso dos projetos (HÄSER *et al*., 2016).

O PMBOK® é um guia de boas práticas para gerenciamento de projetos, dentre as práticas contidas no guia está a Gestão de Comunicações, que visa garantir a geração, coleta, armazenamento, recuperação e destinação das informações geradas ao longo do projeto, de modo a procurar minimizar problemas relacionados à quaisquer ruídos nestas comunicações.

Comunicação não abrange somente a transmissão de uma mensagem, mas também a compreensão do significado desta mensagem pelos *Stakeholders*, portanto o BDD pode se tornar um importante aliado à Gestão das Comunicações, principalmente para definição de requisitos e de seus critérios de aceite, de modo a contribuir com o andamento e entrega do projeto.

* 1. Objetivo

Propor a utilização das práticas de BDD para definir o comportamento e critérios de aceite do *software* (como abordagem de Engenharia de Requisitos e modelagem dos casos de teste), em conjunto com as definições de Gestão das Comunicações do PMBOK®, para avaliar a influência destas abordagens na diminuição de ruídos nas comunicações e compreensão dos requisitos bem como o seu auxílio na finalização bem-sucedida de um projeto.

Estas propostas serão definidas em um Checklist de práticas que buscam alinhar as definições dos comportamentos do sistema em uma linguagem ubíqua com o controle das comunicações do projeto, de modo a assegurar que as informações e definições sejam transmitidas de forma clara e atualizada para as partes interessadas do projeto.

* 1. Justificativas

De acordo com o estudo de (VERNER; SAMPSON; CERPA, 2008), que avaliou o andamento de 8 projetos de desenvolvimento de *software* que não obtiveram sucesso, decisões tomadas sem o entendimento pleno dos requisitos do sistema estiveram entre os fatores responsáveis pela falha de 73% dos projetos avaliados.

Um artigo publicado pelo (PMI, 2013), relata que mais da metade do orçamento de um projeto entra em risco logo em seu início devido à comunicações ineficientes, o que mostra que estes ruídos acarretam em maiores custos e podem fazer com que um projeto falhe ou seja entregue fora do orçamento e até mesmo prazo.

Os resultados do estudo e artigo citados nos parágrafos anteriores corroboram o impacto que ocorre quando não há a preocupação em modelar os requisitos e critérios de aceitação do *software* de um modo que o entendimento sobre o que precisa ser feito é comum aos Stakeholders, e também quando não há uma gestão efetiva das comunicações.

1.4 Metodologia

A metodologia utilizada neste trabalho é uma pesquisa de natureza qualitativa, realizada após a aplicação de um Checklist de práticas em um projeto real de *software* de uma instituição financeira.

Para o desenvolvimento deste trabalho, as seguintes etapas foram realizadas:

1. Definição do escopo do trabalho e motivações para sua realização.
2. Levantamento bibliográfico e estudo de material relacionado à prática de definição de comportamentos esperados e modelagens de casos de teste utilizando BDD e notação Gherkin, verificando como isso auxilia na comunicação entre os *Stakeholders* do projeto, bem como estudo do tópico de Gestão das Comunicações do Projeto presente no PMBOK®.
3. Criação de um *Checklist* com práticas de Gestão das Comunicações, considerando documentos de requisitos e plano de testes no formato proposto pelo BDD.
4. Aplicação do *Checklist* em um projeto real de desenvolvimento de *software*.
5. Aplicação de um questionário e análise dos resultados de modo a identificar se a utilização de BDD aliado às práticas de Gestão das Comunicações contribuíram para o sucesso do projeto.

1.5 Estrutura do Trabalho

O Capítulo 1 INTRODUÇÃO apresenta as motivações, o objetivo, as justificativas, método de pesquisa e a estrutura do trabalho.

O Capítulo 2 REVISÃO BIBLIOGRÁFICA apresenta os principais conceitos relacionados ao trabalho, que são Metodologias Ágeis, *Behaviour-Driven Development*, Gestão de Projetos, Gestão das Comunicações e Engenharia de Requisitos dentro do proposto pelo PMBOK®.

O Capítulo 3 PROPOSTA E APLICAÇÃO DE CHECKLIST PARA GESTÃO DAS COMUNICAÇÕES COM BDD apresenta um Checklist de práticas, focadas em refinamento de requisitos mediante definição de comportamentos com BDD e Gestão das Comunicações. Este Checklist será aplicado em um projeto real de desenvolvimento de *software* para verificar se essas práticas terão influência positiva no sucesso do projeto.

O Capítulo 4 ANÁLISE DE RESULTADOS irá verificar, mediante questionário aplicado com os participantes do projeto, se as práticas propostas contribuíram para uma Gestão de Comunicações eficiente bem como quais as contribuições destas práticas na condução das atividades de gestão do projeto. O próprio *software* entregue também será parâmetro para avaliar os resultados da aplicação deste Checklist, sendo que a principal pespectiva será a da Qualidade (ou seja, se o *software* faz aquilo que o usuário realmente precisa).

O Capítulo 5 CONSIDERAÇÕES FINAIS descreve a conclusão, contribuições e propostas para trabalhos futuros.

REFERÊNCIAS contém os artigos, livros, monografias e documentos utilizados na pesquisa realizada para elaboração deste trabalho.

GLOSSÁRIO relaciona a definição dos termos e siglas apresentados neste trabalho.

2. REVISÃO BIBLIOGRÁFICA

2.1 Considerações Iniciais

Este capítulo fornece o referencial teórico utilizado para realização deste trabalho.

Inicialmente, serão tratados os assuntos: Metodologias Ágeis, Extreme Programming e TDD, para que seja possível fornecer um contexto sobre as origens do BDD, que é um dos temas centrais do trabalho.

Posteriormente, serão compartilhadas as definições e conceitos do BDD, com foco na definição de requisitos e critérios de aceite com utilização de uma linguagem ubíqua, através da sintaxe Gherkin.

Por fim, o outro tema central do trabalho será explorado, que é a disciplina de Gestão das Comunicações do PMBOK®, para relacionar suas práticas com as técnicas de modelagem de requisitos do BDD, e verificar como essas duas abordagens podem se complementar para facilitar as comunicações e consequentemente o sucesso de um projeto.

2.2 Metodologias Ágeis

Os fundamentos das metodologias ágeis surgiram no Manifesto Ágil (KENT *et al.*, 2001), que compartilha os valores e princípios destas metodologias, sendo uma alternativa às abordagens tradicionais utilizadas na época.

Apesar de representarem uma resposta à burocracia e baixa adaptabilidade à mudanças de modelos como o Cascata, estas metodologias envolvem processos disciplinados e o sucesso na adoção das mesmas estão relacionados aos fatores abaixo (PAULK, 2002):

1. Aplicar as metodologias com fidelidade às filosofias e princípios do desenvolvimento ágil.
2. Implementar a metodologia de forma adequada ao ambiente da organização e características da aplicação a ser desenvolvida.

Através da identificação e compilação de melhores práticas de desenvolvimento que utilizavam, e a fim de colaborar com outros profissionais na tarefa de desenvolver *software* (OLIVEIRA *apud* KENT *et al.,* 2015), os autores apresentaram os seguintes valores (KENT *et al.*, 2001):

1. Indivíduos e interações, mais que processos e ferramentas.
2. *Software* em funcionamento, mais que documentação abrangente.
3. Colaboração com o cliente, mais que negociação de contratos.
4. Responder a mudanças, mais que seguir um plano.

De acordo com os autores, os itens à esquerda devem ser mais considerados e tratados com maior prioridade do que os itens à direita. Isso não implica que os valores à direita não tenham importância, mas direciona as prioridades sobre o que deve ser mais valorizado.

Os autores também apresentaram os 12 princípios que estão por trás do Manifesto Ágil e que são seguidos por eles (KENT *et al.*, 2001):

1. Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de *software* com valor agregado.
2. Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.
3. Entregar frequentemente *software* funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.
4. Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.
5. Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.
6. O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.
7. *Software* funcionando é a medida primária de progresso.
8. Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
9. Contínua atenção à excelência técnica e bom design aumenta a agilidade.
10. Simplicidade (a arte de maximizar a quantidade de trabalho não realizado) é essencial.
11. As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis.
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Os valores e princípios do Manifesto Ágil estão presentes em frameworks consolidados como o Scrum (SCHWABER; SUTHERLAND, 2013) e metodologias como o XP (Extreme Programming). Este capítulo irá enfatizar o XP devido à relação desta metodologia com as práticas de teste unitário que posteriormente foram evoluídas para o BDD.

2.3 Extreme Programming

O Extreme Programming (XP), introduzido por Kent Beck, é um conjunto de práticas de desenvolvimento ágil de *software* que enfatiza o trabalho colaborativo (WELLS, 1999) com o objetivo de se obter um produto de maior qualidade e aderente à necessidade dos clientes (OLIVEIRA *apud* KENT, 2015).

Os valores do XP (comunicação, simplicidade, feedback, respeito e coragem), incentivam a colaboração entre desenvolvedores, times de negócio e demais envolvidos, promovendo um ambiente de entregas contínuas, transparência e rápida adaptabilidade a mudanças (WELLS,1999).

O ciclo de vida de um projeto utilizando XP passa pelas seguintes fases: Exploração; Planejamento; Iterações para versões; Produção e Manutenção.

Na fase de exploração, há o leventamento das primeiras histórias de usuário e elaboração da versão inicial do projeto (*design*) de arquitetura do projeto. Na fase de planejamento, as estimativas são realizadas e as histórias de usuário que serão implementadas na próxima iteração são definidas. Na iteração, uma pequena versão do produto é codificada e testada (a nível de teste unitário e posteriormente teste de aceitação). As funcionalidades testadas e aprovadas pelo cliente são liberadas para a fase de produção, e as rejeitadas geram novas histórias que serão reconsideradas na fase de planejamento e adicionadas em versões futuras do produto. A fase manutenção implica em responder às mudanças que surgirem ao longo do ciclo de vida (HENRAJANI, 2007).

A figura 1 representa o fluxo do ciclo de vida e fases de um projeto utilizando XP.

Figura 1 – Fases de um projeto utilizando XP



Fonte: HENRAJANI (2007)

Dentre as práticas introduzidas, estão aquelas relacionadas à qualidade e propriedade coletiva de código (realizadas na fase de iteração), como programação em pares (promovendo revisão do código e aplicação de melhores práticas de desenvolvimento), desenvolvimento dirigido a testes (TDD, no qual os testes unitários são escritos antes da programação da funcionalidade), refatoração de código, integração contínua e testes de aceitação (WELLS, 2000). A figura 2 apresenta um fluxo com estas práticas de propriedade coletiva de código.

Figura 2 – Ciclo de Propriedade Coletiva de Código no XP



Fonte: WELLS (2000), traduzido pelo autor

Das práticas de propriedade coletiva de código promovidas pelo XP, o trabalho irá explorar o TDD, que é o desenvolvimento dirigido por testes unitários.

2.4 *Test-Driven Development*

O Test-Driven Development (TDD) é uma abordagem cíclica de desenvolvimento apresentada por (BECK, 2002) no XP, na qual codificação deve se iniciar pela escrita dos testes, e posteriormente a implementação da funcionalidade que irá fazer aquele teste passar é realizada.

Essa dinâmica visa a obtenção de maior agilidade na entrega das funcionalidades, maior facilidade de manutenção do código e aumento da confiabilidade na solução de modo que os custos com correção de defeitos em produção sejam reduzidos, pois uma *baseline* de testes automatizados é gerada e executada a cada nova versão do *software*, antes de o mesmo ser enviado para produção.

Apesar destas vantagens, a percepção geral é que o TDD é algo complicado de aprender e aplicar, além da questão contra-intuitiva de que a produtividade, ao invés de melhorar, irá diminuir com o surgimento das novas tarefas de escrita de testes no processo. Além disso, a exigência de mais dedicação e disciplina do que nas práticas de desenvolvimento *test-last* mais tradicionais, reforçam a necessidade de capacitar os profissionais no entendimento e aplicação destas técnicas *test-first* (LATORRE, 2013) para que os benefícios de as utilizar sejam obtidos.

O ciclo do TDD ocorre conforme a figura 3:

Figura 3 – Ciclo do Test-Driven Development



Fonte: OLIVEIRA (2015), adaptado pelo Autor

A classificação das tarefas ocorre da seguinte maneira (BECK, 2003):

* *Red tasks*: escrita de um teste que vai falhar pois o código não está implementado.
* *Green tasks*: implementação do código de maneira pragmática para o teste que falhou anteriormente passar.
* *Refactor tasks*: melhoria do código e evoluções na implementação, com o cuidado de eliminar possíveis duplicidades e *Code Smells* (más práticas de implementação).

A fim de colaborar com a limpeza dos testes e do código, e para consolidar as práticas de TDD entre os profissionais de desenvolvimento, Robert C. Martin propôs que a criação dos testes antes da implementação é o início da prática, e que as três leis do desenvolvimento dirigido à testes devem ser consideradas (MARTIN, 2008):

1. Não se deve escrever o código de produção até criar um teste de unidade de falhas.
2. Não se deve escrever mais de um teste de unidade do que o necessário para falhar, e não compilar é falhar.
3. Não se deve escrever mais códigos de produção do que o necessário para aplicar o teste de falha atual.

Entretanto, conforme já explorado nos parágrafos anteriores, existe a percepção de que a aplicação do TDD é complexa, principalmente em seus estágios iniciais, nos quais os desenvolvedores sentem dificuldades na compreensão do que seria a escrita de um teste que falhe.

Uma das respostas à essas dificuldades foi o BDD (DAN NORTH & ASSOCIATES, 2006), que além das abordagens de testes de aceitação, busca apoiar a compreensão sobre os comportamentos esperados do sistema, de modo que seja possível tirar proveito dessa compreensão não somente nas atividades de testes, mas também de refinamento de requisitos.

2.5 *Behaviour-Driven Development*

O *Behaviour-Driven Development* (BDD), é um conjunto de práticas apresentado em um artigo de Dan North publicado em 2006, no qual o autor propõe uma evolução das práticas do TDD, com o objetivo de minimizar as dificuldades de entendimento dos desenvolvedores sobre o quê deve ser testado, de que forma aquilo deve ser testado e mesmo porquê um teste falhou.

Por focar na especificação e testes dos comportamentos que o sistema deve apresentar, essas práticas aproximam os desenvolvedores dos times de negócios e testes, visto que a linguagem utilizada é comum para as parte (ubíqua) e o entendimento do quê deve ser feito e de como o sistema deve se comportar é equalizado. Isso também serve como apoio sobre quais funcionalidades (ou seja, comportamentos) devem ser priorizadas nas entregas de valor do projeto (DAN NORTH & ASSOCIATES, 2006).

Além das boas práticas para escrita de testes (como nomear a assinatura do método de teste com uma frase com comportamento esperado), no BDD há a proposta de que requisitos também são comportamentos, com inspiração no conceito de linguagem ubíqua proposta nas abordagens do *Domain-Driven Design* – DDD (EVANS, 2003).

A fim de criar uma sintaxe que colabore com o conceito e aplicação de uma linguagem ubíqua, o autor buscou inspiração em um formato de história de usuário utilizado na época (DAN NORTH & ASSOCIATES, 2006), no qual os critérios de aceite estavam incorporados à escrita da história, conforme:

Como <ator>

Quero <alguma funcionalidade>

Para <atingir um objetivo>

Com estas bases, uma estrutura de especificação de cenários foi proposta (DAN NORTH & ASSOCIATES, 2006) em uma sintaxe que depois veio a ser conhecida como *Gherkin* (OLIVEIRA; MARCZAK, 2017).

Uma especificação de um cenário utilizando *Gherkin* tem a seguinte sintaxe (OLIVÉRIO, 2016):

Dado <algum contexto inicial> (em inglês, “Given”)

Quando <um evento ocorre> (em inglês, “When”)

Então, <garanta alguns resultados> (em inglês, “Then”)

Para escrita e validação de cenários no contexto do BDD, recomenda-se o uso de ferramentas como o *Cucumber* (WYNNE; HELLESØY, 2012), que possui suporte à sintaxe *Gherkin*, permitindo a criação das documentações de requisitos nesta linguagem, além da automação de testes de aceitação oriundos da escrita destes cenários.

As próprias recomendações sobre como utilizar o *Cucumber* e seu propósito maior de validar o entendimento mútuo dos stakeholders sobre como o *software* deve se comportar reforçam que as práticas do BDD, antes de qualquer coisa, se referem à alcançar este entendimento, de modo que a automação dos testes seja consequência (HELLESØY, 2015), conforme citação direta de um artigo de Aslak Hellesøy, criador do *Cucumber*:

*Cucumber* não é uma ferramenta de testar *software*. É uma ferramenta para testar o entendimento das pessoas em como um *software* (ainda a ser escrito) deve se comportar.

Diante deste entendimento e destas recomendações, o enfoque será na utilização do BDD como mecanismo de modelagem dos requisitos para equalizar o entendimento mútuo entre os times de desenvolviemento e negócios dos comportamentos desejados do *software*, criando uma base de conhecimento cujo acesso e utilização podem ser amplificados utilizando técnicas de Gestão das Comunicações, que é o próximo tópico abordado neste trabalho.

2.y Considerações do Capítulo

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

3. DESENVOLVIMENTO [Colocar título adequado relativo ao tema do trabalho]

[Este capítulo constitui a parte principal do trabalho relativo ao tema. Pode apresentar propostas, análises comparativas, síntese ou extensões de métodos e técnicas, etc. de acordo com os estudos apresentados no capítulo anterior. Pode ser divido em seções e subseções. Colocar títulos adequados]

3.x Título da Seção

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

3.y Considerações do Capítulo

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

4. ANÁLISE DE RESULTADOS

[Este capítulo apresenta uma análise crítica do capítulo anterior. Pode ser complementado com capítulo adicional através de um Estudo de Caso ou Experimento de uma aplicação prática. Colocar títulos adequados]

4.x Título da Seção

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

4.y Considerações do Capítulo

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

5. CONSIDERAÇÕES FINAIS

[Este capítulo descreve as conclusões e as contribuições do trabalho e sugestões de trabalhos futuros que poderão ser desenvolvidos a partir deste trabalho]

5.1 Conclusões

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

5.2 Contribuições do Trabalho

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

5.3 Trabalhos Futuros

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

REFERÊNCIAS

[Para mais detalhes consultar a referência abaixo]

Divisão de Biblioteca – EPUSP. Diretrizes para Apresentação de Dissertações e Teses da USP. 4ª. Edição. São Paulo. 2020. 75p.

<https://doi.org/10.11606/9788573140651>

PAULK, Mark C., Agile Methodologies and Process Discipline. Carnegie Melon Software Engineering Institute, 2002.

GLOSSÁRIO

[As descrições a seguir foram extraídas de bla bla bla bla bla.]

*[Termo 1]* – bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

*[Termo 2]* – bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

Apêndice A – [Título do Apêndice]

[Texto ou documento elaborado pelo autor]

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.

ANEXO 1 – [Título do Anexo]

[Texto ou documento não elaborado pelo autor]

Bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla.