INTEGRAÇÃO DE PRÁTICAS ÁGEIS: KANBAN E SCRUM NA GESTÃO DE PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

INTEGRATION OF AGILE PRACTICES: KANBAN AND SCRUM IN THE MANAGEMENT OF SOFTWARE DEVELOPMENT

Adriely de Souza Lavareda[1]; Ana Paula Moraes e Silva[2]; Marcelo Albuquerque de Oliveira[3]; Gabriela de Mattos Veroneze[4];

RESUMO: Atividades de teste são de suma importância, uma vez que reduzem ao máximo a quantidade de erros até o produto chegar ao usuário final. Nesse cenário, a qualidade das páginas web de comércio eletrônico é crucial para alcançar vantagens competitivas. Deste modo, o estudo propõe investigar a integração de metodologias ágeis, como o Kanban e Scrum, com práticas de desenvolvimento orientado a testes de software e testes automatizados, visando fortalecer aqualidade e a confiabilidade das páginas web, garantindo que elas atendam às demandas do mercado. Utilizando uma abordagem qualitativa, a pesquisa explorou as experiências e percepções dos desenvolvedores, *stakeholders* e usuários finais em relação à implementação das práticas ágeis e TDD. Com base nessas constatações, os resultados indicam que asmetodologias ágeis possibilitaram uma entrega contínua de valor, adaptam-se de forma iterativa às necessidades do projeto.

Palavras-Chave: Teste de Software, Teste Automatizado, TDD, Kanban, Scrum.

ABSTRACT: Testing activities are of paramount importance, as they minimize the number of errors until the product reaches the end user. In this scenario, the quality of e-commerce web pages is crucial to achieve competitive advantages. Thus, the study proposes to investigate the integration of agile methodologies, such as Kanban and Scrum, with software development practices and automated testing, aiming to strengthen the quality and reliability of web pages, ensuring that they meet market demands. Using a qualitative approach, the research explored the experiences and perceptions of developers, stakeholders, and end users regarding the implementation of agile practices and TDD. Based on these findings, the results indicate that agile methodologies enabled continuous delivery of value, adapting iteratively to project needs.

Keywords: Software Testing, Automated Testing, TDD, Kanban, Scrum.

[1] Universidade Federal do Amazonas (UFAM) / email: adrielylavareda@super.ufam.edu.br
[2] Universidade Federal do Amazonas (UFAM) / email: anasilva@super.ufam.edu.br
[3] Universidade Federal do Amazonas (UFAM) / email: marcelooliveira@ufam.edu.br
[4] Universidade Federal do Amazonas (UFAM) / email: gveroneze@ufam.edu.br

1. INTRODUÇÃO

A evolução constante do comércio eletrônico ressalta a importância de páginas web robustas e confiáveis, especialmente aquelas responsáveis por processar transações de compra e venda online. A perspectiva do comércio eletrônico como suporte para transações comerciais por meio de infraestrutura digital apresenta vantagens que abrangem diversas utilizações da Web para incentivar transações comerciais (BLOCH et al., 1996). Diante desse cenário, o objetivo geral deste trabalho é investigar a integração de práticas ágeis, testes de software, Test-Driven Development (TDD) e testes automatizados, visando otimizar o processo de desenvolvimento e melhorar a qualidade do produto.

Para alcançar esse objetivo, é essencial reconhecer a importância dos testes de páginas web no contexto de comércio eletrônico. Eles são fundamentais para garantir uma experiência de usuário consistente e livre de erros, especialmente em sistemas de carrinho de compras. No entanto, para garantir que o processo de compras online seja bem-sucedido, é necessário superar não apenas desafios técnicos, mas também barreiras culturais e organizacionais (DINIZ, 1999). Este trabalho busca, assim, estudar a combinação de metodologias ágeis, como Scrum e Kanban, com TDD e testes automatizados, com o objetivo específico de identificar os desafios enfrentados na integração dessas práticas e como isso pode fortalecer a qualidade e a confiabilidade das páginas web.

A integração de Scrum e Kanban visa aproveitar o melhor de cada metodologia: Scrum é utilizado para planejar e gerenciar as iterações de desenvolvimento (Sprints), enquanto o Kanban permite uma visualização e monitoramento mais flexíveis e adaptáveis do fluxo de trabalho. Já a aplicação de TDD e testes automatizados objetiva garantir a robustez e estabilidade do código, promovendo a detecção precoce de falhas durante o ciclo de desenvolvimento. Dessa forma, o trabalho também objetiva estudar técnicas e ferramentas que possibilitem a implementação eficiente de testes automatizados em práticas ágeis e TDD até o final de janeiro de 2024.

Com esses objetivos específicos, pretende-se não apenas investigar os fundamentos e benefícios do TDD para a qualidade do código, mas também analisar os desafios e soluções possíveis para a integração de metodologias ágeis com testes automatizados, proporcionando sistemas mais estáveis e funcionais no ambiente do comércio eletrônico.

2. REVISÃO DE LITERATURA

2.1 Gestão Ágil

O Gerenciamento Ágil de Projetos (APM) surgiu em 2001 como resultado de um movimento iniciado pela comunidade internacional de desenvolvimento de sistemas de informação, como resposta a uma demanda crescente por inovação, redução dos ciclos de desenvolvimento de produtos, concorrência acirrada e de adaptação a um contexto dinâmico (LIMA, 2011).

A gestão ágil define-se como uma abordagem que busca a flexibilidade, simplicidade, iterações para agregar valor ao produto de forma incremental. O APM é orientado a resultados, permitindo a adaptação do processo e absorvendo mudanças nos requisitos e no escopo do produto (ANGIONI, 2006).

Outros princípios que guiam as práticas da Gestão ágil de projetos são divididos em duas categorias: a primeira relacionada ao produto e clientes, e outra relacionada ao gerenciamento (AMARAL; BENASSI, 2007). Highsmith (2004), apresenta um modelo de gestão ágil de projetos fundamentado em cinco objetivos essenciais: Inovação contínua;

Adaptabilidade do produto; Tempos de entregas reduzidos (ciclos de entrega); Adaptabilidade do processo e das pessoas; e Resultados confiáveis.

A gestão ágil de projetos envolve o esforço necessário para estimular, capacitar e habilitar as equipes de projeto a entregar valor de negócio de forma rápida e confiável, com a participação dos clientes e uma aprendizagem contínua, adaptando-se às mudanças do ambiente e as necessidades em constante evolução. (AUGUSTINE *et al.*, 2005).

2.2 Metodologias Ágeis

Os métodos ágeis surgiram como uma resposta aos modelos clássicos tradicionais de desenvolvimento, evidenciando a necessidade de realizar documentações completas (BECK et al., 2001). Considerada uma nova classe de metodologias de desenvolvimento de software criada para atender à crescente demanda do mercado por processos ágeis e leves, com o objetivo de tornar o desenvolvimento de software rápido, com custo controlável e melhorar a qualidade do software (ABRAHAMSSON, 2003; FOWLER, 2005).

As metodologias ágeis atuam com feedback constante, permitindo a facilidade de adaptar-se rapidamente às mudanças nos requisitos (BARBOSA; LIBARDI, 2010). As metodologias ágeis são adaptativas ao invés de preditivas, permitindo que se ajustem a novos fatores que aparecem durante

o desenvolvimento do projeto e os resultados devem ser entregues em pequenos espaços de tempo (SOARES, 2004).

As metodologias ágeis por meio de funções e atribuições bem definidas, garantem a melhoria de desempenho e promovem o aprendizado da equipe a partir de gerenciamento simples, flexível e interativo, possibilitando a obtenção de melhores resultados ao considerar aspecto de tempo, custo e qualidade (AMARAL et al., 2019).

2.2.1 Kanban

O Kanban é um subsistema do sistema Toyota de produção (STP), tem como principalpropósito simplificar e acelerar as operações de programação, controle e acompanhamento dos sistemas de produção em lotes (FERNANDES; FILHO, 2007). A essência da produção puxada no sistema Kanban é que nada é produzido até que o cliente, seja interno ou externo ao processo, solicite a fabricação de um item específico (ALVES; SANTOS, 2013). Esse sistema opera baseado no uso de sinalizações para acionar a produção e a movimentação dos itens dentro da fábrica, e são tradicionalmente facilitadas por cartões Kanban e painéis porta-Kanbans (ESPARRAGO, 1988). Os cartões são feitos de materiais duráveis para suportar o manuseio frequente ao serem movidos entre os estoques do cliente e do fornecedor do item (COELHO, 2003).

As principais características do sistema Kanban (SHINGO, 1996; TUBINO, 1999): Melhoria total e contínua dos sistemas de produção; Regulação do fluxo de itens globais com controle visual para executar essas funções com precisão; Simplificação do trabalho administrativo, conferindo autonomia ao chão de fábrica; Informação transmitida de maneira organizada e rápida.

O Kanban é uma metodologia que integra eficazmente os princípios da filosofia Lean, buscando a melhoria contínua dos sistemas de produção promovendo o ajuste do fluxo de itens de forma global, permitindo um controle visual preciso para executar essas operações (MACHADO, 2014). Além disso, o Kanban simplifica as tarefas administrativas, conferindo autonomia ao chão de fábrica e proporcionando a transmissão de informações de maneira organizada e rápida. Este método é essencial para otimizar a eficiência, reduzir desperdícios e garantir a entrega de valor ao cliente de maneira ágil e precisa (JUNIOR et al., 2008; SLACK,2008).

2.2.2 SCRUM

Tem como objetivo definir um framework iterativo e incremental em que as atividadessão

realizadas por três papéis principais: *Product Owner, Scrum Master e Time de Desenvolvimento* (FURTADO; MARÇAL, 2010). Destaca-se dos demais métodos ágeis pela maior importância demonstrada ao gerenciamento do projeto, em situações em que planejar em longo prazo torna-se mais difícil (SUTHERLAND *et al.*, 2008).

O ponto de partida do Scrum começa com o Product Backlog, realizada por meio de reuniões com toda a equipe para analisar os requisitos, a tecnologia e suas habilidades e assim, produzir e conceder o melhor software (SCHWABER, 2004). O Product Backlog é priorizado e dividido em releases, e o Sprint é a prática central do Scrum. No início do Sprint, a equipe trabalha na implementação dos itens definidos no Product Backlog que geralmente tem duração de 2 a 4 semanas, embora não haja uma regra fixa para isso (ABRAHAMSSON et al., 2002). Ao final do Sprint, os recursos são codificados, analisados e integrados ao produto ou sistema em desenvolvimento (BALLE, 2011). Além disso, ocorre uma revisão no final do Sprint, no qual os membros do projeto demonstram a nova funcionalidade ao Product Owner e a outras partes interessadas, que fornecem feedback o qual pode influenciar no planejamento do próximo Sprint (COHN, 2002). O último mecanismo do Scrum é o gráfico *burndown*, uma representação gráfica que demonstra o trabalho restante em comparação com o trabalho realizado, sendo útil para observar as etapas do trabalho até ser completado(CARVALHO; MELLO, 2009).

O uso do *Scrum* proporciona uma maior integração dos membros, resultando nasolução rápida dos problemas, além de fortalecer os vínculos com a participação mais ativa docliente no processo, reduzindo os riscos do projeto e entrega de demandas mais frequentes e funcionais (RACHID; STOPA, 2019).

2.3 Teste de Software

Constituem de um conjunto de atividades que podem ser planejadas antecipadamentee conduzidas sistematicamente, podendo possuir estratégias para analisar o software (PRESSMAN, 2006). Testar um software significa verificar, por meio de uma execução controlada, se seu comportamento está conforme o especificado, visando identificar o númeromáximo de falhas com o mínimo de esforço necessário (NETO, 2007). O objetivo do teste de software é demonstrar aos desenvolvedores do sistema e aos clientes que o software está suficientemente preparado para uso (SOMMERVILLE, 2007).

O teste abrange quatro fases: o planejamento de teste, o projeto de casos de teste, a execução

de testes e a avaliação dos resultados dos testes (DELAMARO et al., 2007). Essas atividades devem ser realizadas durante todo o processo de desenvolvimento de software e, geralmente, são divididas em três fases, sendo elas: o teste de unidade, o teste de integração e o teste de sistema (PRESSMAN, 2005).

2.4 Test-Driven Development (TDD)

O TDD é uma metodologia ágil de desenvolvimento de software, derivado do método *Extreme Programming* (XP) e do *Agile Manifesto*. Envolve a realização de um conjunto de interações para completar uma tarefa (BECK, 2002). Testes unitários automatizados são desenvolvidos incrementalmente antes mesmo do código de produção ser desenvolvido (BECK, 2003).

É o processo de execução de um produto para avaliar se ele cumpre com as especificações e opera corretamente no ambiente para o qual foi projetado, com o objetivo de identificar falhas pela equipe de desenvolvimento e corrigi-las antes da entrega final (NETO, 2012; RIBEIRO et al., 2013), sendo uma importante prática de teste que visa crescer a produtividade no desenvolvimento de produtos (KOSKELA, 2008).

O *Test Driven Development* apresenta atributos que resultam em vantagens positivas no desenvolvimento de software, como: reduzir receios do desenvolvedor ao escrever código, simplificar o design da aplicação, diminuir os custos relacionados a defeitos de implementação e detectar impacto de mudanças durante as manutenções preventiva e evolutiva (SILVA, 2017).

2.5 Testes Automatizados

Os testes automatizados consistem na prática de tornar os testes de software independentes de intervenção humana, com o objetivo de aprimorar a qualidade dos sistemas através da verificação e validação (BERNARDO, 2011). Atualmente, os testes automatizados são considerados como uma prática básica para garantir a qualidade de um software. São programas ou scripts simples que executam as funcionalidades do sistema em teste e realizam verificações automáticas nos efeitos colaterais produzidos (DOROTHY; MARKY, 1999).

Uma maneira de se desenvolver softwares dirigidos por testes é implementar trechosde código, conhecidos como testes unitários ou testes funcionais, para testar a menor parte passível de verificação de um programa. Os testes unitários fornecem o feedback que indica se algo parou de funcionar devido a alterações realizadas em uma funcionalidade, identificando erros que ocorreram

em algum ponto do sistema (SPIRLANDELI; ROLAND, 2019) A execução regular dos testes permite descobrir imediatamente os erros e suas origens durante a implementação do código, reduzindo os custos de correção das falhas quando detectadas em usos futuros (ANICHE, 2014).

Os testes automatizados oferecem vantagens significativas nos casos de teste, pois podem ser repetidos de forma rápida e fácil a qualquer momento, sendo assim, a reprodutibilidade dos testes garante que etapas importantes não sejam ignoradas por falhas humanas e facilitando a identificação de erros (SCOTT, 2006; GOJKO, 2009).

3. METODOLOGIA

3.1 Tipo de pesquisa e abordagem

Para alcançar os objetivos propostos, a pesquisa caracteriza-se como aplicada, concentrando-se na aplicação prática das metodologias ágeis, *Test-Driven Development* (TDD) e testes automatizados no desenvolvimento da página web do carrinho de compras. Foi adotada uma abordagem qualitativa, buscando compreender em profundidade as experiências e percepções dos desenvolvedores, *stakeholders* e usuários finais em relação à implementaçãodas práticas ágeis e TDD.

3.2 População e amostra

A amostra é composta por desenvolvedores e *stakeholders* envolvidos no projeto de desenvolvimento da página web do carrinho de compras, bem como usuários finais que irão interagir com o sistema.

3.3 Coleta de dados

Durante o período de cinco semanas no ano de 2023, foram coletados dados sobre o processo de desenvolvimento da página web de carrinho de compras. Incluiu informações sobre o planejamento das Sprints, a execução das tarefas conforme definido no Kanban, a implementação dos testes automatizados e o desenvolvimento das funcionalidades seguindo a abordagem do TDD. Esses dados foram fundamentais para a análise e avaliação da eficácia daintegração das metodologias e práticas propostas neste estudo.

3.4 Análise de dados

Para analisar os dados coletados durante o projeto, utilizamos o Excel para organizar, limpar e

formatar os dados. Em seguida, criamos tabelas para visualizar diferentes aspectos doprocesso de desenvolvimento, permitindo a identificação de tendências e padrões relevantes.

3.5 Etapas

O estudo foi dividido em dez etapas a seguir:

1. Definir equipe scrum
2. Definir artefatos scrum
3. Definir cerimônias scrum
4. Definir Quadro Kanban
5. Definir limite de trabalho (Work In Progress)
6. Definir cronograma e atualização do quadro kanban
7. Descrever teste TDD
8. Descrever desenvolvimento do frontend do carrinho de compras
9. Descrever automação
10. Descrever revisão

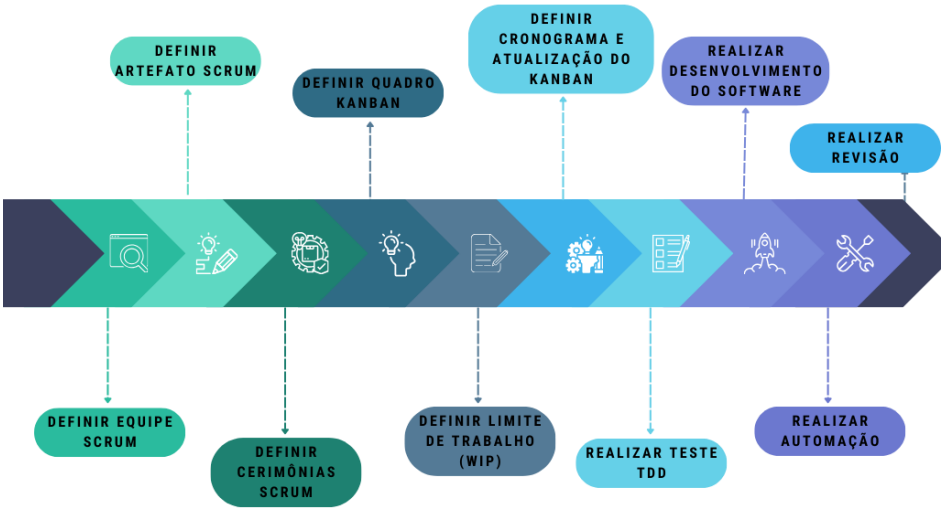
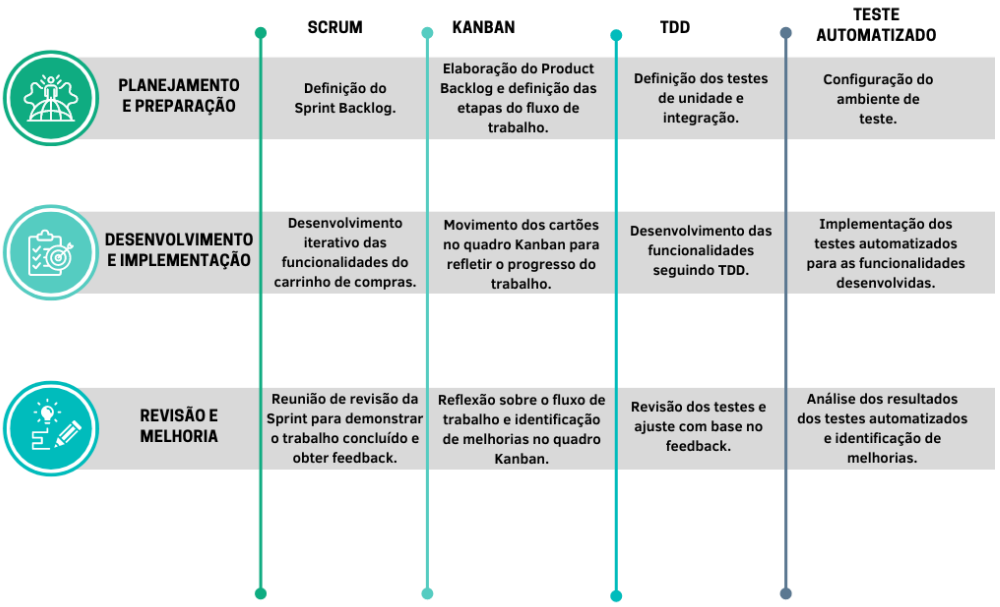


Figura 1 –Processo de desenvolvimento com metodologia ágil.
Fonte: Os autores (2024)

A matriz abaixo resume as atividades planejadas para cada fase do ciclo de vida do projeto. Na etapa de Planejamento e Preparação, são definidos estratégias e recursos, incluindo a elaboração de artefatos Scrum e Kanban, configuração do ambiente de teste automatizado e definição dos testes TDD. Durante Desenvolvimento e Implementação, ocorrea execução das atividades planejadas, como desenvolvimento iterativo, movimentação no quadro Kanban e implementação de testes. Por fim, em Revisão e Melhoria, são realizados reflexões e ajustes com base em revisões e análise de resultados, assegurando uma abordagemsistemática ao ciclo de vida do projeto.

Figura 2 - matriz de implementação.



Fonte: Os autores (2024)

5. RESULTADOS E DISCUSSÕES

5.1 Requisitos do Projeto

Os requisitos para o projeto de desenvolvimento de um carrinho de compras onlineque utilizamos neste projeto são:

- Adicionar Item ao Carrinho: Deve permitir que os usuários adicionem produtos aocarrinho de compras.
- Adicionar Desconto: Deve oferecer a opção de aplicar descontos aos itens no carrinho.
- Finalizar Compra: Deve permitir que os usuários finalizem a compra dos itens no carrinho.
- Forma de Pagamento: Deve disponibilizar opções de pagamento, como cartão de crédito, boleto bancário etc.
- Para implementar o Desenvolvimento Orientado a Testes (TDD) no projeto docarrinho de compras online, os requisitos específicos foram:

- Testes de Unidade: Nesse teste é possível desenvolver testes automatizados para validar o comportamento individual de cada unidade de código, como funções e métodos. Deve garantir que cada componente do sistema funcione conforme o esperado.
- Testes de Integração: Tem o objetivo de criar testes automatizados para verificar a interação entre os diferentes componentes do sistema. Deve assegurar que os diversos módulos do sistema se comuniquem corretamente e funcionem em conjunto sem erros.

Esses dois requisitos são fundamentais para o sucesso do TDD no projeto do carrinho de compras online, pois garantem a validação tanto das partes individuais do código quanto da integração entre essas partes. Isso ajuda a detectar e corrigir erros precocemente, promovendo um desenvolvimento mais eficiente e uma maior confiabilidade do sistema final.

5.2 Equipe SCRUM

Como o projeto foi desenvolvido por duas pessoas, foram elencados os cargos de SCRUM para os membros assumirem uma posição, um dos benefícios dessa decisão é a comunicação direta.

Membro 1: Este membro assumirá o cargo de Product Owner (PO) que é responsável por representar os interesses do cliente e definir as necessidades do produto. E assumirá o cargo de equipe de Desenvolvimento (DEV), que é responsável por implementar as funcionalidades definidas no backlog do produto durante as sprints.

Membro 2: Este membro assumirá o cargo de Scrum Master (SM) que é responsável por garantir que o processo do Scrum seja seguido corretamente e remover quaisquer obstáculos que possam impedir o progresso da equipe. E assumirá o cargo de equipe de Desenvolvimento (DEV) que assim como o outro membro da equipe, este membro é responsável por implementar as funcionalidades definidas no backlog do produto durante as sprints. Eles trabalham em estreita colaboração com o Product Owner para entender osrequisitos e garantir que o produto atenda às expectativas do cliente.

5.3 Artefatos SCRUM

Os artefatos Scrum desempenham um papel crucial na organização e execução do processo

de teste, proporcionando uma visão clara das funcionalidades a serem desenvolvidas e validadas, para o presente teste, definimos os Artefatos Scrum em um Projeto de TDD para um Carrinho de Compras como os seguintes:

- Product Backlog: No início do projeto, o Product Backlog é elaborado, incluindo itens relacionados ao teste de um carrinho de compras, como "Adicionar Produto", "Adicionar Desconto", "Excluir Produto" e "Efetuar a Compra". O Product Owner prioriza esses itens, garantindo que a equipe de desenvolvimento e teste esteja alinhada com asprioridades do negócio.
- Sprint Planning: Durante a reunião de Sprint Planning, a equipe seleciona itensdo Product Backlog para trabalhar durante o próximo Sprint. Os Casos de Teste específicos para o TDD podem ser considerados como requisitos detalhados para a implementação, e a equipe decide quais desses casos serão abordados durante o Sprint.
- Sprint Backlog: Os Casos de Teste, equivalentes aos itens do Product Backlog, são refinados e detalhados na forma de tarefas específicas durante o Sprint Backlog. Aqui, a equipe de teste e desenvolvimento colabora para planejar a execução dos testes, garantindo que cada incremento do produto seja testado de maneira adequada.
- Incremento de Produto: Durante o Sprint, a equipe executa os testes conforme especificado nos Casos de Teste. A cada incremento de produto, a validação dos testes é crucial. Este processo iterativo permite uma entrega contínua de incrementos de produto, cadaum atendendo aos critérios de aceitação estabelecidos no Product Backlog.

Os artefatos Scrum fornecem uma estrutura organizada para integrar efetivamente o TDD em um projeto de teste de software. Desde a definição do Product Backlog até a entrega contínua de incrementos de produto, a metodologia Scrum facilita a transparência, inspeção e adaptação necessárias para um desenvolvimento e testes ágiles e eficazes.

5.4 Cerimônias SCRUM

Tendo em vista que o desenvolvimento não é uma entrega para um cliente externo, optou-se por simplificar as cerimônias, então diminuiu-se a quantidade excluindo o Sprint Retrospective, logo as cerimônias ficaram definidas da seguinte maneira:

- Sprint Planning: Na cerimônia de Sprint Planning, a equipe revisa os itens do Product Backlog relacionados ao teste do carrinho de compras. Entre esses itens estão oscasos de testes detalhados, como "Adicionar Produto", "Adicionar Desconto", "Excluir Produto" e "Efetuar a Compra".

Cada caso de teste é discutido em termos de requisitos, critérios de aceitação e prioridades.

- Daily Scrum: Durante o Daily Scrum, a equipe revisou o progresso diário em relação aos casos de teste. Por exemplo, pode ser discutido como foi a execução do caso de teste "Adicionar Produto" e se houve desafios ou necessidade de ajustes nos testes automatizados correspondentes. Os membros da equipe compartilham insights sobre os casos de teste em andamento.
- Sprint Review: Na Sprint Review, a equipe demonstra os incrementos de produto relacionados aos casos de teste. Para o caso de teste "Efetuar a Compra", por exemplo, a equipe apresenta como o processo de compra foi implementado e validado. O Product Owner e outras partes interessadas avaliam a funcionalidade, proporcionando feedback valioso para aprimoramentos futuros.

Os casos de teste específicos, como "Adicionar Produto", "Adicionar Desconto", "Excluir Produto" e "Efetuar a Compra", tornam-se os pilares fundamentais durante as cerimônias Scrum. Desde a elaboração no Sprint Planning até a validação e apresentação na Sprint Review, esses casos de teste moldam a narrativa do desenvolvimento ágil, garantindouma abordagem orientada a testes no ciclo de vida do projeto.

5.5 Quadro Kanban

O uso de um Quadro Kanban no contexto de um projeto ágil, com ênfase em práticas de Scrum e desenvolvimento orientado a testes (TDD), é uma estratégia eficiente para a gestão visual e acompanhamento contínuo do progresso. Este quadro é projetado para proporcionar transparência, facilitar a colaboração da equipe e integrar as cerimônias Scrum aos processos de desenvolvimento e teste, resultando nos seguintes itens do quadro:

- Backlog Refinement: Atividades relacionadas ao refinamento do ProductBacklog, incluindo a elaboração e discussão de Casos de Teste.
- Sprint Planning: Cerimônia de planejamento da Sprint, onde são definidos os objetivos e selecionadas as tarefas para o Sprint.
- To Do, Doing, Testing, Review, Done: Etapas tradicionais do Quadro Kanban,agora com uma coluna específica para a revisão dos casos de teste durante o Sprint Review.

Movimentação no Quadro: As tarefas progridem de uma coluna para outra, representando seu progresso no ciclo de vida do desenvolvimento e nas cerimônias Scrum.

Exemplos de uso do quadro desenvolvido: Tarefas específicas para o Caso de Teste "Adicionar Produto". Elas percorrem as etapas desde a elaboração dos Casos de Teste até a execução e revisão

durante a Sprint Review.

5.6 Cronograma

Este cronograma semanal segue uma abordagem ágil e estruturada para o desenvolvimento de software, utilizando Scrum e TDD em um projeto de carrinho de compras. A previsão de tempo para cada cerimônia permite uma gestão eficiente do tempo da equipe, mantendo o projeto alinhado com as expectativas do Sprint, havendo uma sprint por caso de teste e mais uma para revisão, totalizando 5 semanas, ou seja, 5 Sprints, que serão organizados da seguinte forma:

Segunda-feira: Sprint Planning (2 horas)

A equipe dedicou aproximadamente duas horas na manhã de segunda-feira para a cerimônia de Sprint Planning. Durante este período, o Product Backlog será revisado e os itens mais prioritários serão selecionados para o Sprint. Os objetivos serão definidos e os Casos de Teste serão elaborados.

Terça-feira a Quinta-feira: Daily Scrum (30 minutos)

Ao longo das manhãs de terça a quinta-feira, serão reservados 30 minutos diários para a cerimônia do Daily Scrum. Durante esses encontros, a equipe atualizará uns aos outrossobre o progresso em relação aos Casos de Teste, discutirá possíveis impedimentos e coordenará esforços para manter o projeto alinhado com as metas estabelecidas.

Sexta-feira: Revisão de Sprint (1 hora)

Na manhã de sexta-feira, está prevista uma hora para a cerimônia de Revisão de Sprint. Durante este período, será realizada a demonstração do incremento de produto, apresentando os Casos de Teste implementados e validados. O Product Owner e outras partes interessadas fornecerão feedback para orientar melhorias futuras. Na tarde de sexta-feira, será reservado tempo para ajustes e aprimoramentos adicionais.

5.7 Teste TDD

Nesta etapa, exploramos o uso do TDD (Desenvolvimento Orientado a Testes) como uma metodologia essencial no desenvolvimento de software. Em vez de escrever código primeiro e depois testá-lo, o TDD preconiza a escrita de testes automatizados antes daimplementação do código de produção. Isso permite que os desenvolvedores se concentremna especificação dos requisitos e no comportamento esperado do software desde o início do processo de desenvolvimento. Ao seguir o ciclo do TDD - escrever um teste que falha, implementar o código para fazer o teste passar e refatorar

o código conforme necessário - os desenvolvedores podem garantir a qualidade do software, identificar e corrigir problemas precocemente e promover um código mais robusto e de fácil manutenção. O TDD também contribui para uma documentação viva do software, uma vez que os testes servem como exemplos claros e concretos do comportamento esperado das funcionalidades. Dessa forma, o TDD não apenas melhora a qualidade do software, mas também impulsiona a eficiência e a confiabilidade do processo de desenvolvimento. A figura abaixo demonstra o funcionamento do TDD.

5.7.1 Casos de Teste e Caso de Uso

Combinar casos de uso e casos de teste é importante para garantir que o software seja um sucesso. Neste projeto, integramos casos de uso e casos de teste para capturar os requisitos funcionais do sistema de forma clara e validar se ele está funcionando conforme especificado. Isso garante a qualidade e precisão do software entregue, identificando falhas antes do lançamento.

Para capturar de forma abrangente os requisitos funcionais do sistema, empregamos casos de uso. Esses casos fornecem uma descrição detalhada das interações entre os usuários eo sistema.

5.8 Desenvolvimento do Frontend de carrinho de compras

Desenvolvemos uma aplicação Python simples usando o framework Flask para criar uma API de carrinho de compras e desenvolvemos também um frontend básico utilizando HTML, CSS e JavaScript para interagir com essa API. Utilizamos a IDE Pycharm com a linguagem de programação Python. Essa interface precisa ter os seguintes requisitos:

Antes de começar a escrever o código, é preciso ter os requisitos das funcionalidades, sendo elas:

Adicionar Item ao Carrinho: Deve permitir adicionar um item ao carrinho.

Remover Item do Carrinho: Deve permitir remover um item do carrinho. Aplicar

Desconto: Deve permitir aplicar um desconto ao total do carrinho.

Finalizar Compra: Deve permitir finalizar a compra e escolher a forma de pagamento.

5.9 Desenvolvimento dos testes automatizado

Para o desenvolvimento dessa automação, foi utilizado o PyCharm, um dos ambientes de desenvolvimento integrado (IDE) mais populares para Python. Essa automação foi desenvolvida

utilizando a biblioteca Selenium em Python, que é amplamente utilizada para automação de testes em páginas web. Ao combinar o PyCharm como ambiente de desenvolvimento e o Selenium como biblioteca para automação de testes, foi possível criar uma automação robusta e flexível, capaz de validar o comportamento de páginas web deforma automatizada e confiável.

Os testes automatizados foram realizados em uma aplicação de carrinho de compras, seguindo a metodologia do Desenvolvimento Orientado por Testes (TDD) e utilizando os casos de testes. Durante o processo de desenvolvimento, cada funcionalidade foi testada individualmente para garantir a integridade do sistema.

Em cada teste, uma mensagem de falha é exibida caso a aplicação não atenda às expectativas. O TDD orienta esse processo, incentivando o desenvolvimento de testes antesda implementação do código, o que ajuda a garantir uma cobertura abrangente e a manter a qualidade do software ao longo do tempo.

Cada bloco de automação criado realiza a ação correspondente e verifica se o resultado é o esperado, lidando com possíveis mensagens de erro ou alertas durante o processo.

5.10 Revisão da Sprint

Na revisão deste projeto, a aplicação do TDD foi fundamental para garantir a qualidade e a robustez do sistema de carrinho de compras. Durante a análise, foram observadas as seguintes contribuições específicas:

- **Cobertura Completa das Funcionalidades Principais**: Os testes automatizados abrangeram todas as principais funcionalidades do carrinho de compras, incluindo adicionar item sem nome, adicionar item com nome, aplicar desconto e confirmar compra. Isso garante que todas as partes cruciais do sistema estejam sendo testadas de forma abrangente.

- **Identificação Antecipada de Problema**s: O TDD permitiu detectar e corrigir problemas de forma precoce no ciclo de desenvolvimento. Ao implementar o teste para adicionar um item sem nome, foi possível identificar rapidamente a necessidade de uma validação no código-fonte para lidar com essa situação, evitando assim possíveis falhas no sistema.

- **Aprimoramento Gradativo das Funcionalidades**: Cada funcionalidade foi desenvolvida de forma incremental, com foco na criação de testes antes da implementação do código real. Isso proporcionou um aprimoramento gradual das funcionalidades, garantindo que cada etapa do desenvolvimento fosse validada e testada antes de prosseguir para apróxima.

- **Refatoração Orientada por Testes**: Durante o processo de desenvolvimento, foram realizadas várias refatorações no código para melhorar sua qualidade e legibilidade, sempre assegurando que os testes continuam passando após cada alteração. Isso resultou em um código mais limpo, organizado e fácil de dar manutenção.
- **Integração com o Fluxo de Desenvolvimento**: A automação de testes foi integrada ao fluxo de desenvolvimento contínuo, permitindo a execução automática dos testessempre que uma nova alteração no código fosse feita. Isso garantiu uma rápida identificação de problemas e a manutenção da estabilidade do sistema ao longo do tempo.

5.11 Integração de Scrum e Kanban

A integração do Kanban e Scrum no desenvolvimento de um carrinho de compras online usando TDD demonstrou ser uma abordagem altamente eficaz para gerenciar o processo de desenvolvimento de software de maneira ágil e orientada a testes.

O Kanban ofereceu uma estrutura visual para acompanhar o progresso das tarefas e garantir uma gestão transparente do fluxo de trabalho. Ao dividir as atividades em colunas como Backlog Refinement, Sprint Planning e etapas de progresso (To Do, Doing, Testing, Review, Done), o quadro Kanban proporcionou uma visão clara das atividades em andamento,facilitando a colaboração da equipe e garantindo a conclusão eficiente das tarefas.

Por sua vez, o Scrum proporcionou um conjunto de cerimônias estruturadas, como Sprint Planning, Daily Scrum e Sprint Review, que permitiram à equipe se reunir regularmente para revisar o progresso, identificar possíveis impedimentos e ajustar o plano conforme necessário. A integração do TDD nos requisitos do projeto, com testes de unidade e integração, garantiu que o código fosse desenvolvido de maneira robusta e confiável, com validação contínua das funcionalidades conforme eram implementadas.

A equipe Scrum desempenhou papéis específicos para garantir que o processo fosse seguido corretamente e que as necessidades fossem atendidas. O cronograma semanal detalhado proporcionou uma estrutura para as atividades do projeto, desde a elaboração dos casos de teste até a revisão de sprint e ajustes finais. Esse cronograma permitiu uma gestão eficiente do tempo da equipe e garantiu que o projeto avançasse de acordo com as metas estabelecidas, proporcionando uma entrega contínua de valor.

Embora a integração do Kanban, Scrum e TDD tenha proporcionado uma abordagem eficaz

para o desenvolvimento do carrinho de compras online e seus testes automatizados, é importante reconhecer e abordar tanto os pontos positivos quanto os negativos para garantir o sucesso e a eficiência do processo. Sendo eles:

Pontos Positivos:

Gestão Visual Eficiente: A combinação do Kanban e Scrum proporcionou umavisualização clara do progresso das tarefas e do fluxo de trabalho, facilitando a identificação de gargalos e a otimização do processo.

Colaboração eficaz: A integração dessas metodologias promoveu uma colaboração eficiente entre os membros da equipe, permitindo uma comunicação transparente e o compartilhamento de conhecimento.

Entrega Contínua de Valor: Graças à adoção do TDD e práticas ágeis, o projeto conseguiu entregar incrementos de produto de alta qualidade de forma iterativa e adaptativa, atendendo às necessidades do cliente de maneira contínua.

Validação Constante: Os testes de unidade e integração integrados ao TDD garantiram uma validação constante das funcionalidades do carrinho de compras, resultando em um código mais robusto e confiável.

Pontos Negativos:

Sobrecarga de Processos: A utilização de múltiplas práticas, como Kanban, Scrum e TDD, pode gerar uma sobrecarga de processos se não forem gerenciadas adequadamente, tornando o fluxo de trabalho mais complexo do que o necessário.

Curva de Aprendizado: Equipes inexperientes podem enfrentar uma curva de aprendizado significativa na adoção dessas metodologias, exigindo tempo e recursos adicionais para adaptação.

Dependência da Equipe: O sucesso desse modelo depende do comprometimento e colaboração de todos os membros da equipe, o que pode ser um desafio se houver falta de alinhamento ou motivação.

Rigidez Estrutural: A rigidez das estruturas do Kanban e Scrum pode limitar a flexibilidade da equipe para lidar com mudanças de requisitos ou emergências durante o desenvolvimento do projeto.

Esses pontos, tanto positivos quanto negativos, devem ser considerados e gerenciados de forma adequada para garantir o sucesso e eficiência do processo de desenvolvimento do carrinho de

compras online.

5.12 Possíveis ganhos

A importância da aplicação dessas práticas reside na capacidade de melhorar a eficiência, a qualidade e a adaptabilidade do processo de desenvolvimento de software. Em um mercado competitivo e em constante mudança, as empresas que adotam abordagens ágeis e práticas de desenvolvimento de software centradas na qualidade estão mais bem posicionadas para entregar valor aos clientes de forma rápida e eficaz.

A integração de práticas ágeis como Kanban e Scrum na gestão de projetos de desenvolvimentos de software com testes automatizados utilizando TDD tem uma série de benefícios significativos, além de resultar em economias e ser viável. A tabela abaixo destaca a importância da aplicação do projeto, os possíveis ganhos que podem ser alcançados e como esses ganhos se traduzem em retornos monetários viáveis para o projeto. Além disso, mostra como a aplicação do projeto pode resultar em economias significativas em várias áreas, contribuindo para o sucesso geral do empreendimento.

Tabela 1 - Ganhos potenciais.

Aspecto	Importância	Possíveis Ganhos	Ganhos Monetários Viáveis	Economias Potenciais
Eficiência da Equipe	Maximizar a utilização dos recursos da equipe e garantir que as tarefas sejam concluídas de maneira eficiente e oportuna.	Redução do tempo ocioso da equipe - Melhoria da produtividade e eficiência - Menos retrabalho e desperdício de tempo	Redução de custos associados à mão de obra, como horas extras ou realocação de recursos para atividades mais produtivas.	Economia de recursos financeiros significativos através de uma melhor gestão do tempo e dos recursos disponíveis.
Qualidade do Software	Assegurar que o produto final seja confiável, livre de defeitos e atenda aos requisitos de qualidade.	Redução de custos de manutenção e correção de bugs - Aumento da confiança do cliente - Maior estabilidade e confiabilidade do	Redução de custos associados à correção de bugs em estágios avançados do projeto ou após o lançamento.	Redução dos custos associados à obsolescência do produto e à necessidade de re-trabalho devido a mudanças nos requisitos.

		software		
Satisfação do Cliente	Garantir a entrega de um produto de alta qualidade que atenda às necessidades e expectativas dos clientes.	Maior satisfação do cliente - Maior probabilidade de retenção de clientes - Mais recomendações e referências	Aumento das receitas devido à maior satisfação do cliente e maior probabilidade de retenção de clientes.	Redução dos custos associados à aquisição de novos clientes devido à maior probabilidade de retenção de clientes e recomendações.

Entrega rápida de valor	Entregar valor ao cliente de forma rápida e contínua, garantindo que as necessidades do cliente sejam atendidas de maneira oportuna	Entrada mais rápida no mercado - Maior vantagem competitiva - Aumento das receitas	Aumento das receitas devido à entrada mais rápida no mercado e à maior vantagem competitiva.	Redução dos custos associados ao tempo de desenvolvimento prolongado e ao ciclo de vida estendido do projeto.
Adaptabilidade	Capacidade de responder rapidamente a mudanças nos requisitos do projeto ou nas necessidades do cliente.	Maior flexibilidade e adaptabilidade - Redução do risco de obsolescência do produto - Melhoria da capacidade de inovação e diferenciação	Redução dos custos de desenvolvimento de produtos futuros.	Redução dos custos associados à obsolescência do produto e à necessidade de retrabalho devido a mudanças nos requisitos.

Fonte: Os autores (2024)

6. CONCLUSÕES

A integração das práticas ágeis, testes de software, Test-Driven Development (TDD) etestes automatizados no contexto de projetos de desenvolvimento de software proporcionou uma abordagem eficaz para otimizar o processo de desenvolvimento e garantir a qualidade do produto final. Ao longo deste estudo, buscamos investigar os fundamentos e benefícios do TDD para a qualidade do código, identificar os desafios enfrentados na integração de práticas ágeis, TDD e testes automatizados, e estudar técnicas e ferramentas para implementação eficiente de testes

automatizados em práticas ágeis e TDD.

As metodologias ágeis, como Scrum e Kanban, desempenharam um papel fundamental no sucesso deste estudo. Através da adoção de abordagens ágeis, pudemospromover uma colaboração mais eficaz entre os membros da equipe, facilitar a adaptação a mudanças de requisitos e aumentar a transparência no processo de desenvolvimento. Além disso, a metodologia ágil permitiu uma entrega contínua de valor, garantindo que as necessidades fossem atendidas de forma interativa e adaptativa ao longo do projeto.

No entanto, não podemos ignorar os desafios enfrentados durante este estudo. Aintegração de práticas ágeis, TDD e testes automatizados exigiu uma curva de aprendizado significativa e exigiu um esforço adicional para garantir a colaboração efetiva da equipe.Além disso, a rigidez das estruturas do Kanban e Scrum às vezes limitava nossa flexibilidadepara lidar com mudanças de requisitos ou emergências durante o desenvolvimento do projeto.

Para futuras pesquisas, recomendamos explorar ainda mais as técnicas de teste automatizado, avaliar ferramentas de desenvolvimento de software e conduzir estudos de casoadicionais sobre a implementação de metodologias ágeis em diferentes contextos. Essasiniciativas podem fornecer insights valiosos para aprimorar ainda mais a integração de práticas ágeis, TDD e testes automatizados em projetos de desenvolvimento de software.

Em última análise, este estudo contribuiu não apenas para a comunidade acadêmica, mas também para a prática profissional, demonstrando os benefícios tangíveis de adotar abordagens ágeis e práticas de teste de software no processo de desenvolvimento de software. Ao continuar explorando e refinando essas metodologias, podemos continuar a impulsionar a inovação e a excelência no desenvolvimento de software.

AGRADECIMENTOS

O presente trabalho é decorrente do projeto de Pesquisa e Desenvolvimento (P&D) 001/2020, firmado entre a Samsung, Universidade Federal do Amazonas e FAEPI, que conta com financiamento da Samsung Eletrônica da Amazônia Ltda, usando recursos da Lei Federal nº 8.387/1991, estando sua divulgação de acordo com o previsto no artigo 39.º do Decreto nº 10.521/2020.

REFERÊNCIAS

ABRAHAMSSON, P. et al. **Agile software development methods – review and analysis**. Espoo: VTT Publications, 2002.

ABRAHAMSSON, P.et al.New directions on agile methods: a comparative analysis.In: INTERNATIONAL CONFERENCEON SOFTWARE ENGINEERING, 25., 2003, Portland.

Anais eletrônicos [...]. Portland: ICSE, p. 244 - 254, 2003.

ALVES, Jakeline Aleixo; DOS SANTOS, Aparecida da Penha. Logística lean para redução dos efeitos da variação da demanda no abastecimento de linhas de produção. **Perspectivas Contemporâneas**, v. 8, n. 1, 2013.

AMARAL, Vanessa de Araújo; OLIVEIRA, Angela Maria Fleury de; ISONI, Taciana Maria. Inovação social e metodologias ágeis: uma história de sucesso. Nova Lima: **Revista DOM**, 2019.

ANGIONI, M. et al. Integrating XP project management in development enviroments.

Journal of Systems Architeture. Elsevier, n. 52, ago. p. 619-626, 2006

AUGUSTINE, S., PAYNE, B., SENCINDIVER, F. and WOODCOCK, S. Agile Project

Management: Steering from the Edges. *Communications of the ACM*, 48, 85- 89, 2005.

Beck,K., Manifesto for Agile Software Development. 2001.

Beck, K. "Test Driven Development: By Example", Addison Wesley, 2002.

Beck, K. Test-Driven Development: By Example Addison-Wesley, 2003.

BENASSI, J.L.G.; AMARAL, D. C. Gerenciamento ágil de projetos aplicado ao desenvolvimento de produto físico. In: SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO, 14, 2007. Bauru. **Anais.**.. Bauru: UNESP, 2007.

BERNARDO, Paulo Cheque. **Padrões de testes automatizados**. Tese de Doutorado. Universidade de São Paulo, 2011

BLOCH, M.; PIGNEUR, Y.; SEGEV, A. On the road of electronic commerce: a business value framework, gaining competitive advantage and some research issues. **CITM Working Paper,** Mar. 1996.

CARVALHO, B. V.; MELLO, C. H. P. Revisão, análise e classificação da literatura sobre o método de desenvolvimento de produtos ágil Scrum. In: SIMPÓSIO DE ADMINISTRAÇÃO DA PRODUÇÃO, LOGÍSTICA E OPERAÇÕES INTERNACIONAIS – SIMPOI, 12., 2009,

São Paulo. **Anais**... São Paulo, 2009.

COELHO, Rodrigo Marcelo. Implantação e simulação do sistema KANBAN de movimentação de materiais: Estudo de caso na indústria eletrônica do segmento EMS. Dissertação de mestrado, Universidade Federal de Minas Gerais, 2003.

COHN, M. *Introduction to Scrum - An Agile Process.* **Mountain Goat,** Lafayette, 2002.

Delamaro, M. E. Maldonado, J. C., e Jino, M. Introdução ao Teste de Software. **Elsevier**,2007.

DINIZ, Eduardo Henrique. Comércio eletrônico: fazendo negócios por meio da internet.

Revista de Administração Contemporânea, v. 3, p. 71-86, 1999.

ESPARRAGO, Romeo A. Jr. Kanban. Production and Inventory Management Journal; FirstQuarter, ABI/INFORM Global, 1988.

FERNANDES, F. C. F; GODINHO FILHO, M. Sistemas de coordenação de ordens: revisão, classificação, funcionamento e aplicabilidade. **Revista Gestão & Produção,** São Carlos, v.14,n.2, 2007.

HIGHSMITH, Jim. Agile Project Management:Creating Inovative Products. Boston: Addison-Wesley, 2004.

JUNIOR, M. L; FILHO, M.G. Adaptações ao sistema kanban: revisão,classificação, análise eavaliação. **Gestão e Produção,** v.15, n.1, jan/abr. 2008.

KOSKELA, L.,Test Driven Practical TDD and Acceptance TDD for JavaDevelopers, **Ed. Manning**, 2008.

LIMA, Nilton C. A Gestão Ágil como Novo paradigm no Gerenciamento de Projetos.In: **Trabalho apresentado no VII Congresso Nacional de Excelência e Gestão**. 2011.

MACHADO, Cátia Milena Lopes. Ferramenta computacional para apoio à minimização dosdesperdícios do processo produtivo. 2014.

MARÇAL, Ana Sofia Cysneiros; FURTADO, Maria Elizabeth Sucupira. SCRUMMI: Um processo de gestão ágil baseado no SCRUM e aderente ao CMMI. In: **Anais do IX Simpósio Brasileiro de Qualidade de Software**. SBC, 2010. p. 425-439.

MARK F; DOROTHY G. *Software Test Automation.* Addison-Wesley Professional, 1999.

GOJKO A, *Bridging the Communication Gap: Specification by Example and Agile Acceptance Testing.* Neuri Limited, 2009.

NETO, Arilo. Introdução a teste de software. **Engenharia de Software Magazine**, v. 1, p. 22,2007.

Neto, Arilo. Introdução a Teste de Software, **Engenharia de Software Magazine**, 2012.

PRESSMAN, R. *Software engineering - a prsctitioner´s approach.* 6th ed. McGraw-Hill, 2005. Pressman, R. S. Engenharia de Software, McGrawHill, 6ª ed., São Paulo, 2006.

RIBEIRO, Samira et al. Uma avaliação da abordagem TDD (Test Driven Development) em uma empresa desenvolvedora de software madura. In: **Anais do XII Simpósio Brasileiro de Qualidade de Software**. SBC, 2013. p. 302-309.

SCOTT W. Test driven database design. *Tass Quarterly magazine,* Pág 4. September,2006

SILVA, Leonardo Barroso. TDD (Test Driven Development). In: **Congresso de Tecnologia da Informação**. 2017.

SCHWABER K. ***Agile Project Management With Scrum.*** 1º edição. [S.I]: Microsoft Press, 2004.

SLACK, N. et al. Gerenciamento de Operações e de Processos. Porto Alegre, Bookman, 2008.

SOARES, Michel dos Santos. Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software. **Revista Eletrônica de Sistemas de Informação**, [S.l.], v. 3,n. 1, june 2004. Sommerville, I.Engenharia de Software,Pearson Education, 8ª ed., São Paulo, 2007

SPIRLANDELI, Cleber; ROLAND, Carlos Eduardo de França. A utilização de testesautomatizados no desenvolvimento de software. 2019.

STOPA, Gabriel Rocha; RACHID, Christien Lana. Scrum: Metodologia ágil como ferramenta de gerenciamento de projetos. **CES Revista**, v. 33, n. 1, p. 302-323, 2019.

SUTHERLAND, J; JAKOBSEN, C; JOHNSON, K. Scrum and **CMMI** Level 5: The Magic

Potion for Code Warriors. 41 st International Conference on System Sciences, Havaí, 2008.