RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 4 Nro. 2 2023

PLANEJAMENTO DE SISTEMA WEB PARA AUXILIAR O TRABALHO DO PRODUCT OWNER COM BASE NOS PRINCÍPIOS DA METODOLOGIA ÁGIL

Lauriany Borges Campos
Graduada em Engenharia de Software – Uni-FACEF
laurianycampos.npx@gmail.com

Carlos Alberto Lucas
Docente do Uni-FACEF
projetos@profcarloslucas.com.br

Resumo

O *Product Owner* desempenha um papel crucial no desenvolvimento de software, agindo como a ponte vital entre equipes de desenvolvimento e metas de negócios. Seus desafios envolvem a gestão de um *backlog* de produtos em constante evolução e a garantia de entregas de valor aos clientes. À medida que as abordagens ágeis ganham popularidade, este estudo enfatiza a importância do *Product Owner* e apresenta um planejamento de sistema web para enfrentar esses desafios. O objetivo é aprimorar a colaboração, a entrega contínua de valor e a adaptação às mudanças no processo de desenvolvimento. A metodologia concentra-se principalmente no levantamento de requisitos, incluindo uma entrevista detalhada com o *Product Owner* para compreender suas necessidades essenciais. Além disso, desenvolvemos elementos como modelagem com *BPMN*, diagramas de caso de uso, documentação de requisitos e considerações de *UX* e *UI* Design. Os resultados abrangem a criação de protótipos de alta fidelidade, oferecendo uma representação precisa do produto final. Na conclusão, enfatizamos a importância da documentação na fase inicial do planejamento de software e reconhecemos os desafios enfrentados durante o desenvolvimento do projeto. Além disso, destacamos a evolução contínua do projeto para atender às necessidades em constante evolução.

Palavras-chave: Ágil. Documentação. *Product Owner*.

Abstract

The Product Owner plays a crucial role in software development, serving as the vital bridge between development teams and business goals. Their challenges involve managing an ever-evolving product backlog and ensuring continuous delivery of value to customers. As agile approaches gain popularity, this study underscores the significance of the Product Owner and introduces a web system planning to address these challenges. The objective is to enhance collaboration, continuous value delivery, and adaptation to changes in the development process. The methodology primarily focuses on requirements gathering, including an in-depth interview with the Product Owner to understand their essential needs. Additionally, we have developed elements such as BPMN modeling, use case diagrams, requirement documentation, and considerations of UX and UI Design. The results encompass the creation of high-

fidelity prototypes, providing an accurate representation of the final product. In conclusion, we emphasize the importance of documentation in the early stages of software planning and acknowledge the challenges faced during project development. Furthermore, we highlight the potential for continuous project evolution to meet ever-evolving needs.

Keywords: *Agile. Documentation. Product Owner.*

1 Introdução

No cenário em crescimento da tecnologia e do mundo dos negócios, o papel central desempenhado pelo *Product Owner* (Proprietário do produto) na concepção de produtos e serviços reside na sua capacidade de conectar as necessidades dos usuários à visão estratégica da empresa. Ele desempenha um papel crucial ao propor soluções que atendam tanto às exigências do mercado quanto aos anseios dos clientes (BRASIL, 2023). Ainda mais, recai sobre o *Product Owner* a responsabilidade de tomar decisões de suma importância, como a definição das funcionalidades que devem ser incorporadas em cada ciclo de desenvolvimento e o momento propício para o lançamento do produto (ADIKARI et al., 2013).

É indiscutível que o *Product Owner* atua como um ponto de convergência entre os domínios de negócios, tecnologia e design, buscando meticulosamente a harmonização desses elementos para assegurar resultados que estejam perfeitamente alinhados com os propósitos estratégicos da empresa.

Nesse contexto, ele enfrenta diariamente uma série de desafios que vão desde dilemas técnicos relacionados à qualidade e organização da arquitetura do software até a gestão de requisitos em constante mutação, problemas de comunicação interna da equipe e a análise de riscos complexos (JONASSON et al., 2014).

Dentro dessa perspectiva, o foco central deste estudo é aprofundar a compreensão dos desafios diários enfrentados pelos *Product Owners* e, a partir do planejamento de um sistema, propor uma solução que se torne uma parceira indispensável para a gestão eficaz de suas responsabilidades. O propósito primordial é a minimização dos impactos negativos nos *backlogs* sob a responsabilidade do *Product Owner*, por meio do oferecimento de soluções que simplifiquem suas tarefas e aprimorem o processo de desenvolvimento de produtos e serviços.

Adicionalmente, este estudo tem como objetivo oferecer perspectivas valiosas sobre os processos e práticas de documentação de software. Além disso, abordamos de forma abrangente aspectos significativos relacionados ao Design de Experiência do Usuário (*UX*) e à Interface de Usuário (*UI*).

Portanto, o objetivo deste artigo se concentra na fase inicial do planejamento de software, oferecendo uma proposta de solução web capaz de facilitar o gerenciamento de *backlogs* para o *Product Owner* (PO). Cabe ressaltar que o escopo deste trabalho não se concentra no desenvolvimento de um sistema funcional, mas sim na elaboração das documentações e elementos essenciais dessa fase primordial de um projeto, por meio de ferramentas como *BPMN (Business Process Model and Notation)*, documentação de casos de uso, diagramas de atividades e conceitos de Experiência do Usuário (*UX* e *UI*).

Para conduzir este estudo, adotou-se uma abordagem metodológica que incluiu uma entrevista detalhada com o experiente *Product Owner* Matheus Capoia. Essa entrevista possibilitou uma compreensão aprofundada de suas necessidades, desafios e expectativas. A metodologia de levantamento de requisitos desempenhou um papel crucial, fornecendo assim a base sólida necessária para a elaboração da documentação subsequente.

2 Referencial Teórico

Nesta seção são abordados como principais temas a metodologia ágil e o papel do *Product Owner*. No que tange à Metodologia Ágil, destacamos sua crescente popularidade nos últimos anos, como uma abordagem popular e eficiente para gerenciar projetos de desenvolvimento de software. Quanto ao papel do *Product Owner,* enfatizamos sua importância como responsável por definir as funcionalidades do produto, priorizá-las de acordo com o valor para o negócio, além de garantir a entrega de valor ao cliente em cada iteração do produto.

2.1 Metodologia ágil

Com o aumento da demanda por soluções de softwares ágeis e adaptáveis, a metodologia ágil tem se consolidado como uma abordagem cada vez mais popular nos últimos anos, sobretudo em ambientes de incerteza e mudanças constantes.

Por isso, muitas empresas têm adotado esta metodologia para gerenciar o processo de desenvolvimento de software de forma mais eficiente. Segundo Geo Report (2023), "Uma das mudanças mais significativas introduzidas pela Metodologia Ágil foi a entrega contínua de valor ao longo do projeto. Em vez de esperar até o final para entregar o produto final, os desenvolvedores começaram a entregar incrementos funcionais em curtos períodos de tempo. Isso permitiu que os clientes tivessem visibilidade do progresso do projeto e fizessem ajustes conforme necessário, evitando surpresas desagradáveis no final. "

A metodologia ágil é uma abordagem para gestão do desenvolvimento de software que valoriza a colaboração, a adaptação e a entrega constante de valor ao cliente. Essa metodologia surgiu como resposta aos problemas enfrentados por projetos de desenvolvimento de software tradicionais, que muitas vezes eram lentos, burocráticos e pouco flexíveis, como a metodologia cascata. Segundo Hirotaka Takeuchi e Ikujiro Nonaka (1986), a abordagem ágil se baseia em equipes multidisciplinares, que trabalham de forma colaborativa e iterativa, com foco em entregas frequentes e funcionais.

Um dos conceitos chave desta abordagem é a sprint, que é um período de tempo fixo durante o qual o time trabalha em um conjunto de tarefas específicas. A sprint geralmente dura de uma a quatro semanas e resulta na entrega de um incremento de software funcional. Durante o sprint, o time se reúne diariamente em uma reunião chamada de daily stand-up, na qual cada membro do time compartilha o que fez no dia anterior, o que pretende fazer no dia atual e se há algum impedimento para o trabalho (KNAPP et al., 2017).

2.1.1 Manifesto Ágil e seus princípios

O Manifesto Ágil é um documento fundamental para a filosofia de desenvolvimento de software que se tornou cada vez mais popular nas últimas décadas. Ele foi escrito em 2001 por um grupo de desenvolvedores de software que buscavam uma abordagem mais eficaz e eficiente para a criação de software (OBJECTIVE, 2021.).

O manifesto começa com a seguinte declaração: "Estamos descobrindo maneiras melhores de desenvolver software, fazendo-o nós mesmos e ajudando outros a fazê-lo" (AGILE MANIFESTO, 2001). Isso destaca a natureza colaborativa e inovadora do manifesto .

O Manifesto Ágil estabelece quatro valores fundamentais para a criação de software eficaz:

- Indivíduos e interações mais que processos e ferramentas;
- Software em funcionamento mais que documentação abrangente;
- Colaboração com o cliente mais que negociação de contratos;
- Responder a mudanças mais que seguir um plano.

Esses princípios fundamentais são ainda relevantes hoje em dia e muitas organizações continuam a adotá-los para melhorar a sua agilidade e adaptabilidade. Martin Fowler (2001), um dos autores do documento, observou que "o Manifesto Ágil não é uma ferramenta de marketing, mas um conjunto de princípios que orientam a tomada de decisões". Isso destaca a importância dos valores fundamentais do manifesto Ágil na tomada de decisões em relação ao desenvolvimento de software.

Em vez de seguir um processo rígido, as equipes ágeis devem estar sempre cientes dos valores fundamentais do manifesto e adaptar sua abordagem para garantir que esses valores sejam atendidos.

2.1.2 Frameworks Ágeis: Kanban, XP e Scrum

Nos últimos anos, a adoção de *frameworks* ágeis para o gerenciamento de projetos tem experimentado um notável crescimento, principalmente na área de desenvolvimento de software. Os três principais *frameworks* ágeis são Scrum, Kanban e *Extreme Programming* (XP). De acordo com Report (2023), "cada uma dessas metodologias possui sua própria abordagem e práticas específicas, porém, todas compartilham a mesma mentalidade ágil, pautada em adaptabilidade, entrega contínua e colaboração."

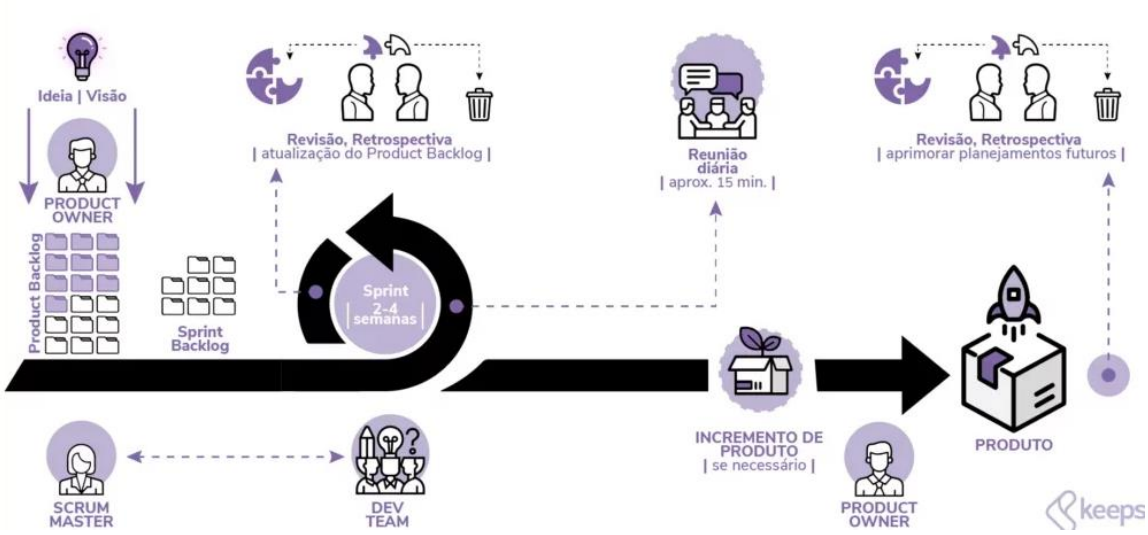
O Kanban é uma abordagem visual e flexível para gerenciamento de projetos ágeis. David J. Anderson, um dos principais responsáveis pela popularização do

método no mundo, afirma: "O Kanban é uma maneira de visualizar o trabalho, limitar o trabalho em andamento e maximizar a eficiência da entrega". O Kanban enfatiza a importância do fluxo contínuo de trabalho e da otimização do processo de trabalho.

O *Extreme Programming* (XP) é um *framework* ágil que enfatiza a qualidade do software, a colaboração da equipe e a adaptação às mudanças. Como Ron Jeffries, um dos criadores do XP, afirma: "A qualidade do software é crucial para o sucesso de qualquer projeto de desenvolvimento de software. O XP enfatiza a importância do teste automatizado, da programação em pares e da integração contínua para garantir a qualidade do software".

O Scrum é um *framework* popular para gerenciamento de projetos ágeis e enfatiza a entrega de valor de maneira repetitiva e incremental. Como Ken Schwaber, um dos criadores do Scrum, afirma: "A cada sprint, o software é testado e avaliado. Com isso, ele pode ser melhorado continuamente. Essa abordagem incremental leva a uma melhoria contínua do produto final". O Scrum também enfatiza a importância da colaboração, da transparência e da adaptação às mudanças. Na Figura 1, é ilustrado como funciona a metodologia Scrum.

Figura 1 - Como a metodologia Scrum funciona.



Fonte: KEEPS, 2022

2.1.3 Papéis e Responsabilidades na Metodologia Ágil

A definição de papéis e responsabilidades é fundamental para garantir a eficácia do processo ágil e a entrega de valor para o cliente. Nesse sentido, existem alguns papéis e responsabilidades que são comuns em muitas metodologias ágeis, tais como Scrum e Kanban.

O *Product Owner* desempenha um papel fundamental ao definir o escopo do projeto e gerenciar o *backlog* do produto. Segundo Loulakis e White (2023) o *Product Owner* é descrito como um "visionário, pintando uma imagem do produto futuro e reavaliando constantemente a trajetória para chegar lá." Essa definição enfatiza a importância de ter uma visão clara do produto final e estar sempre adaptando e ajustando o caminho para alcançar esse objetivo.

O *Scrum Master* é responsável por garantir que a equipe de desenvolvimento siga as práticas ágeis e que o processo do Scrum seja implementado corretamente. De acordo com Ken Schwaber "o *Scrum Master* é responsável por ajudar a equipe a compreender e aplicar o Scrum, bem como ajudar o *Product Owner* a entender e lidar com o *backlog* do produto".

Por fim, a equipe de desenvolvimento é responsável por entregar o produto. Conforme Jeff Sutherland, co-criador do Scrum, "a equipe de desenvolvimento é auto-organizada e multifuncional. Ela é responsável por entregar o produto no final de cada sprint".

No Kanban papéis não são tão bem definidos quanto o Scrum, mas ainda assim é importante definir as responsabilidades de cada pessoa envolvida no processo. De acordo com David J. Anderson, criador do método Kanban, "é importante que todos os membros da equipe estejam cientes das políticas, do processo e do trabalho a ser feito".

2.2 *Product Owner*

O papel do *Product Owner* é fundamental no desenvolvimento ágil de software. Ele é responsável por definir as funcionalidades do produto e priorizá-las de acordo com o valor que trazem para o negócio. Além disso, ele é o principal responsável por manter a visão do produto alinhada com os objetivos do negócio e com as necessidades dos usuários finais.

2.2.1 Definição e responsabilidades do *Product Owner*

O *Product Owner* é fundamentalmente descrito como "a pessoa que guia a equipe em direção ao alvo correto, auxiliando o time a alcançar essa meta de forma eficiente" (Cohn, 2009, p. 125). As principais funções do *PO* (*Product Owner*) incluem garantir que todos os membros da equipe estejam buscando uma visão comum para o projeto, estabelecendo prioridades para que a funcionalidade de maior valor esteja sempre sendo trabalhada e tomando decisões que levem a um bom retorno sobre o investimento no projeto.

O papel do *Product Owner* é crucial para o sucesso do desenvolvimento ágil de software. De acordo com Visagie (2023), o *PO* assume a responsabilidade de estabelecer o *backlog* do produto, priorizando os itens nele contidos e garantindo que a equipe tenha uma compreensão clara das funcionalidades e requisitos do produto para o sucesso do negócio.

É fundamental que o *PO* tenha a habilidade de ajustar o *backlog* do produto de acordo com as mudanças nas necessidades do negócio e dos usuários ao longo do tempo e traduzi-las em requisitos do produto que agreguem valor para o negócio. Isso mostra que o *PO* deve estar em constante comunicação com os usuários finais e com os stakeholders do negócio, para garantir que o produto entregue o valor esperado.(SCALED AGILE FRAMEWORK, 2023)

2.2.2 Desafíos enfrentados pelo *Product Owner*

A partir do mapeamento sistemático conduzido por Carolin, Jil e Curt (2019), ficou evidente a importância crucial das tomadas de decisões realizadas pelo *Product Owner*, ao mesmo tempo que revelava os desafios decorrentes da falta de informações estruturadas. A função do *PO* é desafiadora, demandando habilidades de gestão e uma visão estratégica do projeto para direcionar a equipe em relação às funcionalidades cruciais para o sucesso do negócio e alinhadas às expectativas dos usuários finais.

No contexto do desenvolvimento ágil de software, o *PO* desempenha um papel fundamental, que abrange a definição e priorização das funcionalidades do produto, bem como a garantia de que a equipe entregue valor ao cliente em cada ciclo do

produto (DIANA, 2016). Abaixo, apresentamos alguns dos desafios mais comuns enfrentados pelos POs:

- Comunicação efetiva entre todas as partes interessadas: o *PO* precisa equilibrar as expectativas dos clientes, os objetivos do negócio e as capacidades técnicas da equipe de desenvolvimento.
- Indisponibilidade durante a Sprint: quando o *PO* não está disponível, pode haver uma falta de direcionamento claro para a equipe de desenvolvimento, levando a dúvidas sobre prioridades, requisitos e decisões a serem tomadas.
- Lidar com a volatilidade e incerteza do mercado: em setores como a Fintech, as condições econômicas podem mudar rapidamente, afetando as necessidades e prioridades dos clientes. o *PO* deve adaptar continuamente a estratégia do produto para se manter competitivo.
- Compreensão dos requisitos técnicos do produto: é essencial que o *PO* compreenda e comunique eficazmente os requisitos técnicos do produto para a equipe de desenvolvimento.
- Foco excessivo no Sprint *Backlog*: um erro comum é concentrar-se demais no que está ou não no Sprint *Backlog*, perdendo de vista o quadro geral do projeto.
- Dificuldade em priorizar: decidir o que priorizar pode ser desafiador, especialmente quando as equipes têm expectativas diferentes, e o *PO* pode aprender com a experiência que fazer essa escolha nem sempre é fácil.
- Gerenciamento inadequado das expectativas das partes interessadas: não conseguir gerenciar adequadamente as expectativas das partes interessadas pode resultar em conflitos e desalinhamento.
- Falta de comunicação da visão do produto: é essencial que o *PO* comunique claramente a visão do produto para toda a equipe, para que todos estejam alinhados com os objetivos.
- Desconexão com a equipe: ficar desconectado da equipe pode levar a problemas de comunicação e falta de entendimento mútuo.

Esses desafios destacam a complexidade do papel do *Product Owner*, que desempenha um papel vital no sucesso do desenvolvimento ágil de software e no alcance dos objetivos do negócio (DIANA, 2016).

2.2.3 Definição de *Backlog* do Produto, Épicos, *Features* e Histórias de Usuário

O *backlog* do produto representa o ponto central que abriga todo o trabalho a ser realizado no futuro pela equipe responsável pelo desenvolvimento do produto. De acordo com Frey (2023), esse *backlog* é composto por uma lista priorizada de desafios, tais como épicos, histórias de usuários e tarefas.

Conforme delineado por Mike Cohn (2005), os épicos são como os grandes capítulos de uma história. Eles representam funcionalidades ou componentes de alto nível que contribuem para o escopo geral do projeto. Em termos mais simples, os épicos são as partes amplas e distintas do produto que podem ser divididas em partes menores. Eles são uma maneira de agrupar funcionalidades relacionadas e ajudam a manter uma visão geral do que está sendo desenvolvido.

As features, ou funcionalidades, são os elementos intermediários entre os épicos e as histórias de usuário. Elas representam partes significativas do épicos, sendo mais específicas e detalhadas que os épicos, mas ainda não tão granulares quanto as histórias de usuário. As features geralmente se concentram em aspectos funcionais do produto e podem ser decompostas em várias histórias de usuário relacionadas.

As histórias de usuário são descrições específicas das funcionalidades do produto, do ponto de vista do usuário. Cada história descreve uma tarefa que um usuário deve realizar, com uma estrutura simples: quem é o usuário, o que ele deseja fazer e por quê. Elas são a base do trabalho da equipe de desenvolvimento ágil, conectando diretamente o desenvolvimento do produto às necessidades dos usuários.

2.2.4 Priorização de requisitos, planejamento de *sprints* e controle do projeto

Segundo Leffingwell (2018), a priorização de requisitos deve ser feita com base no valor de negócio que cada requisito traz para o produto. É importante que o *Product Owner* esteja envolvido nessa etapa, de forma a garantir que os requisitos mais importantes sejam priorizados.

De acordo com Schwaber e Sutherland (2010), a sprint é caracterizada como um período de tempo definido, durante o qual a equipe de desenvolvimento concentra seus esforços em um conjunto de funcionalidades previamente determinadas. Um ponto crucial reside na necessidade de que o planejamento das sprints seja fundamentado no *backlog* do produto.

O controle do projeto é uma etapa crítica em equipes que utilizam a metodologia ágil. Essa etapa é responsável por garantir que o projeto esteja seguindo o cronograma previsto, que os requisitos estejam sendo atendidos e que os riscos sejam identificados e gerenciados de forma eficiente (DRUMOND, s.d.).

Além disso, é importante que a equipe realize reuniões diárias de acompanhamento do projeto, conhecidas como Daily Scrum. Segundo essas reuniões são uma oportunidade para que a equipe discuta o andamento do projeto, identifique impedimentos e ajuste o planejamento conforme necessário (SUTHERLAND, 2014).

3 Resultados

Nesta seção, abordamos as principais etapas do processo de planejamento do software, incluindo o levantamento de requisitos, modelagem com *BPMN*, diagrama de caso de uso juntamente com sua documentação, diagrama de atividade e consideração de aspectos de *UX* e *UI Design*.

3.1 Levantamento de Requisitos

A importância dos requisitos no sucesso de um projeto de software é indiscutível, pois eles representam as necessidades fundamentais das partes interessadas envolvidas. Segundo Pressman (2011, c. 5), um sistema pode ser considerado elegante em sua concepção, mas se não resolver corretamente os problemas apresentados, não atenderá às necessidades de ninguém. Portanto, é imprescindível que o sistema a ser desenvolvido suprir essas demandas de forma eficiente.

Os requisitos desempenham um papel crucial no desenvolvimento do projeto, servindo como base para o planejamento, gerenciamento de riscos, realização de testes e controle de mudanças, tornando-se ferramentas essenciais para o sucesso

do empreendimento. Ao garantir que o software atenda aos requisitos, é possível assegurar que ele estará alinhado com as necessidades dos clientes (PRESSMAN, 2021).

Nesse contexto, conduziu-se um levantamento de requisitos por meio de uma entrevista com Matheus Capoia, *Product Owner*, com o propósito de compreender suas necessidades, desafios e obstáculos diários e documentar um sistema eficaz para atendê-los. O entrevistado autorizou a divulgação da entrevista, com o termo de consentimento disponível no repositório de documentação deste artigo (Figura 13).

Todo esse processo foi aprimorado a partir da criação de uma documentação de requisitos, utilizando tabelas para especificar um ID para cada requisito, uma descrição, uma categoria, uma descrição detalhada do requisito e todas as regras envolvidas nele, como pode ser observado no exemplo da figura 2:

Figura 2 - Documentação de requisitos

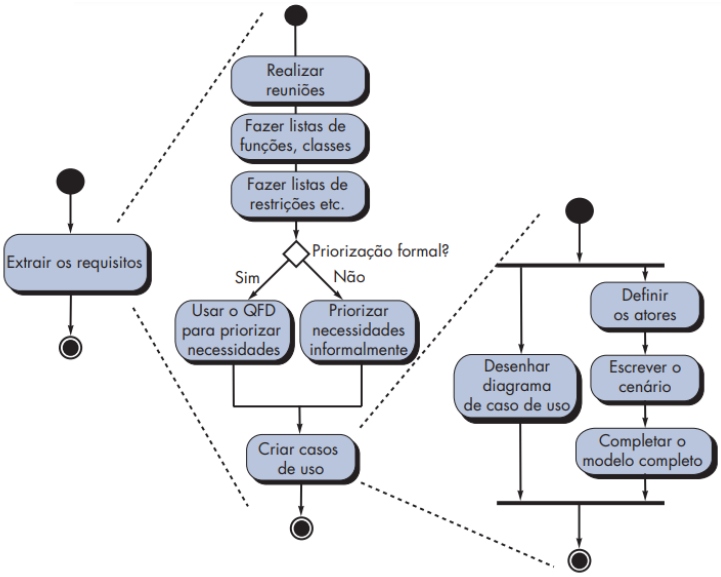
ID	RF002
Nome do Requisito	Gerenciamento do backlog de histórias de usuário
Descrição	O sistema web deverá permitir ao Product Owner (PO) gerenciar o backlog de histórias de usuário de forma eficiente, permitindo a organização das tarefas por ordem de prioridade e a visualização do progresso do projeto através de relatórios de produtividade. Isso auxilia o PO a tomar decisões baseadas em dados, identificar gargalos e alocar recursos de forma mais eficiente.
Categoria	Explícitos
Prioridades	Essenciais
Informações	O sistema web deverá permitir a criação, edição, exclusão e ordenação das histórias de usuário de forma simples e intuitiva. deverá ser possível também a atribuição de pontos de história (story points) e a definição de dependências entre as histórias.
Regra de Negócio	O backlog de histórias de usuário é uma ferramenta essencial para o sucesso do projeto, pois permite a definição clara dos objetivos e a priorização das tarefas de acordo com o valor entregue ao usuário final. É importante que o PO tenha acesso fácil e ágil a essa informação para garantir que o projeto esteja alinhado com as necessidades do cliente.

Fonte: os autores

3.2 Business Process Model and Notation (BPMN)

Neste trabalho, adotamos o modelo de levantamento de requisitos da *UML (Unified Modeling Language - UML)*, conforme recomendado por Roger Pressman em "Engenharia de Software: uma Abordagem Profissional" (página 143). O processo, ilustrado na Figura 3, envolve a extração dos requisitos e a realização de reuniões para, em seguida, criar listas de funções e classes. No entanto, optamos por uma mudança estratégica ao escolher a elaboração do *BPMN* para mapear os processos, em vez de utilizar a lista de funções e classes.

Figura 3: Diagramas de atividades UML para levantamento de requisitos



Fonte: Pressman (2011)

O *BPMN*, ou *Business Process Model and Notation*, é uma notação gráfica amplamente utilizada para modelar processos de negócio, proporcionando uma representação visual clara e padronizada. WHITE (2008) destaca sua utilidade na gestão de processos, ajudando a identificar gargalos, facilitar tomadas de decisão e promover melhorias contínuas.

No contexto do trabalho do *Product Owner*, o *BPMN* foi desenvolvido para fornecer uma visão precisa do processo, destacando as etapas do usuário (*Product Owner*) e mapeando com precisão os fluxos, assegurando eficiência e alinhamento com as necessidades do negócio. O *BPMN* está disponível no *QR code* da Figura 4, bem como pelo seguinte link: https://drive.google.com/file/d/1C7RgVcLJP6xidSFImg0ms8-GvYjQtlxy/view?usp=sharing.

Figura 4: *BPMN* do Projeto

Fonte: os autores

3.3 Diagrama de Caso de Uso

De acordo com Fonseca (2022), o diagrama de caso de uso é uma representação visual que ilustra as diferentes formas e cenários possíveis de utilização de um sistema. Essa poderosa ferramenta é amplamente utilizada em engenharia de software para capturar as interações entre os usuários e o sistema, destacando como um usuário realizará ações específicas e como o sistema responderá a essas ações.

Por meio do diagrama de caso de uso, é possível capturar os requisitos funcionais do sistema de forma clara e compreensível, permitindo uma visão abrangente das funcionalidades e dos fluxos de trabalho do software em desenvolvimento. É uma técnica fundamental para garantir que o software atenda às necessidades e expectativas dos usuários, contribuindo para o sucesso do projeto. O diagrama está disponível no QR *code* da Figura 5, bem como pelo seguinte link: https://drive.google.com/file/d/1gVbDcbQwyGXoOq4GaZGaiH21iK5kc3Hq/view?usp=sharing.

3.4 Documentação de Caso de Uso

A documentação de um caso de uso fornece instruções sobre seu funcionamento, atores envolvidos, atividades a serem executadas, eventos que o acionam e possíveis restrições (GUEDES, 2018). No contexto deste projeto, foi elaborada a documentação de cada caso de uso do diagrama, que está disponível no *QR code* da figura 6, bem como pelo seguinte link: https://drive.google.com/file/d/1gO4FpH7PCDh1DYundkrrg8jpc5qCN579/view?usp=sharing.

Figura 5: Diagrama de Caso de Uso

Fonte: os autores

Figura 6: Documentação do Caso de Uso

Fonte: os autores

3.5 Diagrama de Atividades

O diagrama de atividade na UML é uma ferramenta valiosa para descrever e modelar processos complexos. Ele pode representar uma ampla gama de atividades, desde métodos de programação até processos de negócios. Esse diagrama oferece uma visão clara e visual das etapas envolvidas em uma atividade, tornando-o essencial para a compreensão e documentação de sistemas complexos (GUEDES, 2018).

O diagrama de atividade UML desempenha um papel crucial na análise e modelagem de processos de negócios, proporcionando uma representação clara de todas as etapas do modelo e de seu fluxo de trabalho. Para profissionais que atuam nessa área, a compreensão aprofundada desse diagrama é fundamental, pois facilita a identificação de oportunidades de otimização e aprimoramento (GREEN, 2021).

Nesse contexto, elaboramos três diagramas de atividades que se concentram nas partes mais complexas de nosso trabalho, acessíveis por meio do QR Code localizado na Figura 12 em nosso repositório de documentações e também apartir do link: https://drive.google.com/drive/folders/1U1MhQBiyhOQI2Ql3zV5LKDo4ksjscSRp?usp=sharing.

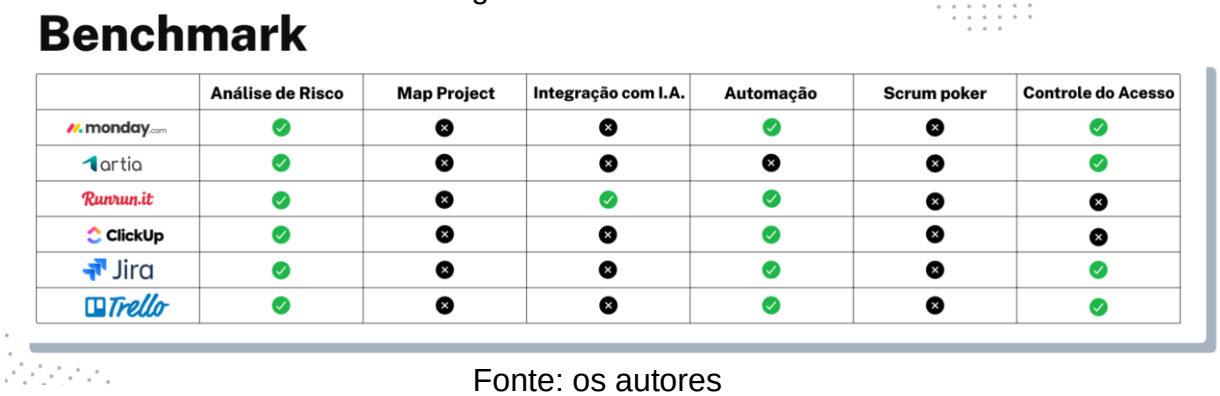
3.6 Benchmarks

Para melhor compreender nossos concorrentes e suas ofertas, conduzimos uma pesquisa de mercado. Identificamos as funcionalidades-chave dos sistemas correlatos focados na gestão de projetos e desenvolvemos um benchmark comparativo, visualizável na Figura 7. Benchmarks são amplamente usados para aprimorar eficiência, comparar produtos e fornecer informações aos consumidores sobre desempenho tecnológico (BIGOGNO, 2014).

É fundamental entender os principais competidores, avaliar seu desempenho no atendimento ao público-alvo e identificar suas características-chave para comparação. Também é importante considerar as características planejadas pela

nossa equipe, permitindo avaliar nossa proposta em relação à concorrência e obter insights para melhorias futuras (LEMOS, 2008).

Figura 7: Benchmark



Fonte: os autores

3.7 Personas

A construção de Personas desempenha um papel fundamental na segmentação de dados analíticos, permitindo obter insights valiosos para otimizar a experiência do usuário. Delimitar adequadamente esses perfis é crucial para focar nas necessidades e desafios específicos do público-alvo, evitando generalizações excessivas e garantindo soluções alinhadas com expectativas e requisitos reais (SIQUEIRA, 2022).

Por meio da criação de personagens fictícios fundamentados em dados de pesquisa e na compreensão das necessidades dos usuários, as Personas enriquecem a documentação do projeto, conferindo-lhe maior profundidade e respaldo empírico (LISBOA, 2017). Neste contexto, com base nas práticas dos processos de *UX* e *UI* Design, uma persona foi delineada para este trabalho, que pode ser visualizada pelo *QR Code* da Figura 8 e também com o seguinte link: https://drive.google.com/file/d/1MMeXy-0CENahwFvYRKsiDuD7ZAyuNFO0/view?usp=sharing.

3.8 Jornada de Usuário

A jornada do usuário é uma ferramenta de extrema importância para compreender como os indivíduos interagem com um produto ou serviço. Ela

desempenha um papel fundamental na organização eficaz dos pontos de contato, resultando em uma experiência unificada e fluida (MACEDO, 2016).

Quando aplicada ao contexto das Personas, essa abordagem permite a criação de um mapa da jornada do usuário que oferece uma visão abrangente das etapas e das emoções envolvidas durante a utilização do produto. O mapa correspondente está disponível para visualização no *QR Code* da Figura 9 e também acessível pelo link a seguir:https://drive.google.com/file/d/19iOqVdEbFdw2L1yisJhrWhqIy0GLMs3j/view?usp=sharing .

Figura 8: Personas

Fonte: os autores

Figura 9: Jornada de usuário

Fonte: os autores

3.9 Wireframes e Styleguide

O Styleguide é essencial para definir o design digital de um produto, economizando tempo e reduzindo custos de desenvolvimento (JUNIOR, 2018). Ele inclui elementos padronizados, como grids, tipografia, cores, ícones e botões, garantindo consistência visual e a identidade do produto. Nosso Styleguide está disponível para visualização pelo *QR Code* da Figura 10 e acessível pelo link: https://www.figma.com/file/NI6dTqtzRoedjIZsIiXr6/Styleguide?type=design&node-id=0%3A1&mode=design&t=daewaqBKMFWhPwCZ-1.

O Wireframe oferece uma representação visual da estrutura básica da interface e permite esboçar a disposição dos elementos principais da interface sem necessariamente incluir a identidade visual completa ou conteúdo específico. É amplamente empregada nas fases iniciais do desenvolvimento para proporcionar uma visão clara da organização e do layout da interface (REPLOGLE, 2023). Nosso Wireframe pode ser visualizado pelo *QR Code* da Figura 11 e acessado por meio deste link: https://abre.ai/gNRO.

Figura 10: Styleguide

Fonte: os autores

Figura 11: Wireframe

Fonte: os autores

4 Discussão

Nesta seção, é apresentada a prototipação de alta fidelidade elaborada para abranger de forma precisa e abrangente os diversos fluxos de processos da aplicação, oferecendo uma visão detalhada das interações e funcionalidades do software em planejamento.

4.1 Prototipação de Alta fidelidade

Conforme definido por Rogers, Sharp e Preece (2013, p. 390), um protótipo é:

> Um protótipo é uma manifestação de um design que permite aos stakeholders interagirem com ele explorarem sua adequação [...] um protótipo pode ser qualquer coisa desde um storyboard à base de papel até uma peça complexa de software [...].

A criação de um protótipo de alta fidelidade é extremamente importante para entender a experiência e o comportamento do usuário de forma abrangente. Ao representar fielmente o produto final, essa abordagem ajuda a prevenir e minimizar erros após o lançamento no mercado. O protótipo, rico em detalhes e interatividade, apresenta com precisão todo o fluxo do produto (TERA, 2020b). O nosso protótipo pode ser visualizado no *QR Code* da Figura 12 e acessado pelo link: https://abre.ai/gNRL.

Figura 12: Prototipação de Alta fidelidade do Projeto

Fonte: os autores

5 Conclusão

Diante da complexidade inerente ao papel do *Product Owner* no cenário empresarial contemporâneo, este estudo abordou os desafios cotidianos enfrentados por esses profissionais que desempenham um papel crucial como o ponto de convergência entre as esferas de negócios, tecnologia e design. Aprofundando nossa compreensão desses desafios, nossa principal meta foi planejar um sistema que simplificasse suas tarefas e aprimorasse o processo de desenvolvimento de produtos e serviços.

Embora o mercado atual conte com uma variedade de aplicativos para gestão de projetos, nenhum deles se dedica exclusivamente a atender às necessidades específicas dos *Product Owner*s. A partir desse cenário, surgiu a ideia de desenvolver um projeto destinado a auxiliar esses profissionais cruciais na condução de planejamentos estratégicos que atendam a todas as demandas dos stakeholders.

Durante o curso deste estudo, destacou-se a importância da documentação de software, abrangendo ferramentas como *BPMN*, documentação de casos de uso, diagramas de atividades e princípios de Experiência do Usuário (*UX* e *UI*), buscamos fornecer insights valiosos sobre esses processos e práticas.

A elaboração do projeto não foi isenta de desafios significativos, que exigiram soluções criativas e estratégicas. Um dos principais obstáculos enfrentados foi a identificação de um diferencial sólido em nossa proposta, algo que verdadeiramente destacasse nosso projeto em um mercado já saturado de soluções diversas. Além disso, enfrentamos a complexa tarefa de apresentar as funcionalidades essenciais ao *Product Owner* de maneira que fossem intuitivas e visualmente acessíveis, garantindo que o sistema fosse facilmente compreensível e utilizável por esses profissionais em seu ambiente de trabalho. Esses desafios serviram como estímulo para aprimorar nossa abordagem e buscar soluções inovadoras ao longo do processo de desenvolvimento do projeto.

Elaborando a documentação e elementos essenciais durante a fase de planejamento de software, bem como conduzindo entrevistas detalhadas com o *Product Owner* Matheus Capoia, conseguimos realizar um levantamento detalhado de

requisitos que estabeleceu uma base sólida para a concepção da solução proposta. Ao final, esta solução atingiu plenamente os objetivos estabelecidos neste artigo. Ela proporciona uma abordagem robusta e altamente eficaz para satisfazer as necessidades do *Product Owner.*

É importante ressaltar que a falta de implementação prática não foi uma limitação deste trabalho, mas sim uma escolha deliberada. Nosso enfoque foi aprimorar a documentação e mitigar riscos na fase inicial do planejamento de software, visando compreender as regras de negócios e estabelecer uma base sólida para o desenvolvimento do software. Essa ênfase valoriza a importância dessa etapa fundamental, deixando a possibilidade de implementação prática para fases subsequentes do projeto.

Adicionalmente, este projeto possui potencial para evolução. Tanto a interface quanto as funcionalidades podem ser aprimoradas e expandidas para melhorar ainda mais o fluxo de trabalho dos Product Owners. A busca pela melhoria contínua é fundamental para garantir que o sistema esteja sempre alinhado com as necessidades gerais e em constante evolução.

Para uma melhor visualização e acesso às documentações desenvolvidas no projeto, disponibilizamos tanto o *QR Code* (Figura 13) quanto o link direto para o repositório para facilitar a consulta: https://drive.google.com/file/d/1C7RgVcLJP6xidSFImg0ms8-GvYjQtlxy/view?usp=sharing.

Figura 4: Repositório de Documentação do Projeto

Fonte: os autores

Referências

ADIKARI, S.; MCDONALD, C.; CAMPBELL, J. Agile user experience design: A design science enquiry. In: . Melbourne: RMIT University Press, 2013.

AGILE MANIFESTO. Princípios por trás do Manifesto Ágil. 2001. Disponível em: https://agilemanifesto.org/iso/ptbr/principles.html. Acesso em: 19 de maio de 2023.

ANDERSON, David J. Kanban: Mudança Evolutiva Bem-Sucedida para o seu Negócio de Tecnologia. Publicado em 2010. 13º edições.

BECKER, Eduardo. Wireframe: o guia definitivo para o design de interfaces. São Paulo: Editora Senac São Paulo, 2022.

BIGOGNO COSTA, Matheus; SALUTES, Bruno (Ed.). O que é benchmark? Canaltech, 14 de Agosto de 2014. Disponível em: https://canaltech.com.br/hardware/o-que-e-benchmark-26350/. Acesso em: 14 ago. 2023.

BRASIL, Sérgio. Product Owner - Guia de boas práticas: Como liderar a estratégia do produto: Um guia prático para garantir o sucesso de seu produto. 10 de abril de 2023. 29 páginas. Português.

COHN, M. (2005). Agile Estimating and Planning. Edição Inglês. Editora: Prentice Hall PTR.

COHN, M. (2009, 1ª edição). Succeeding with Agile: Software Development Using Scrum. Editora: Addison-Wesley Professional.

DIANA. Conselhos para Product Owners em Formação. Disponível em: https://www.devmedia.com.br/conselhos-para-product-owners-em-formacao/34615. Acesso em: 25 de maio de 2023.

DRUMOND, Claire. Agile Project Management. Disponível em: https://www.atlassian.com/br/agile/project-management. Acesso em: 25 de maio de 2023.

FONSECA, Letícia. 10 exemplos de diagrama de caso de uso (e como criá-los). Jun 21, 2022. Disponível em: https://pt.venngage.com/blog/diagrama-de-caso-de-uso-2/. Acesso em: 23 jul. 2023.

FREY, Patrick. The CUBUDE method: How to prioritize your product backlog to maximize your chances for success. Edição Inglês. Publicado em 12 de janeiro de 2023.

GEO REPORT. Metodologia Ágil: Os Princípios da Eficiência e Inovação. 1. ed. 2023.

GREEN, Daniel. 6 Exemplos de Diagrama de Atividade UML e Tutorial Detalhado. Última atualização em 23-06-2021. Disponível em: https://gitmind.com/pt/modelos-diagrama-atividade-uml.html. Acesso em: 10 de julho de 2023.

GUEDES, Gilleanes T. A. UML2: Uma Abordagem Prática. 3ª Ed. São Paulo: Novatec Editora, 2018.

JUNIOR, Apparicio. Styleguide o jeitinho fácil. Medium, 2018. Disponível em: [inserir link]. Acesso em: 10 de julho de 2023.

KEEPS. "Scrum: o que é, como usar e quais as principais etapas deste método ágil". Disponível em: https://keeps.com.br/scrum-o-que-e-como-usar-e-quais-as-principais-etapas-deste-metodo-agil/. Acesso em: 1 de abril de 2023.

KNAPP, Jake; ZERATSKY, John; KOWITZ, Braden et al. Sprint: O método usado no Google para testar e aplicar novas ideias em apenas cinco dias. Edição Português. Intrínseca, 2017.

LEFFINGWELL, D. (2018). SAFe 4.5 Reference Guide: Scaled Agile Framework for Lean Enterprises. Editora: Addison-Wesley Professional.

LEMOS, M. L. F. (2008). Um roteiro para análise da concorrência e da estratégia competitiva. Revista do BNDES, 14(29), 235-276.

LISBOA, A. C. (2017). Personas: uma ferramenta para a documentação de projetos de design de interação. In: Anais do 15º Congresso Brasileiro de Pesquisa e Desenvolvimento em Design. São Paulo: Blucher, p. 2620-2631.

LOULAKIS, C., & White, E. (2023). The Scrum Master's Guide to the Galaxy: A Companion for Undertaking the Role of Scrum Master.

MACEDO, Paula. Mapeando a jornada e a experiência do usuário. In: UX Collective. Published in: UX Collective. Aug 18, 2016. Disponível em: https://brasil.uxdesign.cc/mapeando-a-jornada-e-a-experi%C3%AAncia-do-usu%C3%A1rio-49d2c921cbf. Acesso em: 15 ago. 2023.

OBJECTIVE. Manifesto Ágil. 2021. Disponível em: https://www.objective.com.br/insights/manifesto-agil/. Acesso em: 18 de maio de 2023.

PRESSMAN, Roger S. Engenharia de software – Uma Abordagem Profissional. 7. Ed. Porto Alegre. AMGH Editora Ltda, 2011.

REPLOGLE, Nicole. Melhores Ferramentas de Wireframe. Publicado em 5 de julho de 2023. Disponível em: https://zapier.com/blog/best-wireframe-tools/. Acesso em: 11 de julho de 2023.

SCALED AGILE, Inc. Product Owner | Scaled Agile Framework. Disponível em: https://scaledagileframework.com/product-owner/. Acesso em: 18 de maio de 2023.

SCHWABER, Ken. Agile Project Management with Scrum. Microsoft Press, 2004. 192 páginas, Paperback.

SCHWABER, Ken; SUTHERLAND, Jeff. Scrum Guide. 1ª edição. scrum.org, 2010.

SHARP, H., ROGERS, Y., & Preece, J. (2013). Design de interação [recurso eletrônico]: Além da interação humano-computador (3ª Ediçao ed.). (I. Gasparini, Trans.) Porto Alegre: Bookman Editora.

SIQUEIRA, André. Persona: o que é? Disponível em: https://resultadosdigitais.com.br/marketing/persona-o-que-e/. Acesso em: 16 jun. 2023.

SUTHERLAND, Jeff. Scrum: The Art of Doing Twice the Work in Half the Time. 1ª edição. Editora Currency, 2014. 237 páginas, Hardcover. ISBN: 9780385346450. Publicado em 30 de setembro de 2014.

SVERRISDOTTIR, H. S.; INGASON, H. T.; JONASSON, H. I. The role of the product owner in scrum - comparison between theory and practices. In: Selected Papers from the 27th Ipma. Dubrovnik, Croatia: Elsevier, 2014. (Procedia Social and Behavioral Sciences, v. 119), p. 257–267.

TAKEUCHI, Hirotaka; NONAKA, Ikujiro. The New New Product Development Game. Harvard Business Review, [S.l.], v. 64, n. 1, p. 137-146, 1986.

TERA, Somos. Prototipagem de alta fidelidade: o que é, quando, por que e como usar?.[S.l.]: Somos Tera, 2020. Disponível em: https://medium.com/somos-tera/prototipagem-de-alta-fidelidade-635d745b662b. Acesso em: 25 set. 2022.

UNGER-WINDELER, Carolin & Klünder, Jil & Schneider, Kurt. (2019). A Mapping Study on Product Owners in Industry: Identifying Future Research Directions.

VISAGIE, I. (2023). Agile Project Management with Scrum: A Comprehensive Guide for Professionals.

WHITE, S. A. (2008). BPMN Modeling and Reference Guide: Understanding and Using BPMN. Tampa, FL: Meghan-Kiffer Press.

HENDRICKSON, Mike; JEFFRIES, Ron; ANDERSON, Ann, et al. Extreme Programming Installed. 1st Edition. Addison-Wesley Professional, 2000. 288 páginas.