



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
CURSO DE GRADUAÇÃO DE ENGENHARIA DE SOFTWARE

JACKSON JACINTO TEIXEIRA MAIA

**DESENVOLVIMENTO DE UMA FERRAMENTA WEB PARA APOIO A
GERÊNCIA DE EQUIPES DE DESENVOLVIMENTO DISTRIBUÍDAS
DE *SOFTWARE***

QUIXADÁ
2022

JACKSON JACINTO TEIXEIRA MAIA

DESENVOLVIMENTO DE UMA FERRAMENTA WEB PARA APOIO A
GERÊNCIA DE EQUIPES DE DESENVOLVIMENTO DISTRIBUÍDAS DE
SOFTWARE

Monografia apresentada ao Curso de Graduação
de Engenharia de Software do Campus de
Quixadá da Universidade Federal do Ceará,
como requisito parcial à obtenção do grau de
bacharel em Engenharia de Software.

Orientadora: Prof.^a. Ma. Antônia Diana Braga
Nogueira

QUIXADÁ

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária

Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M186d Maia, Jackson Jacinto Teixeira.

Desenvolvimento de uma ferramenta web para apoio a gerência de equipes de desenvolvimento distribuídas de software / Jackson Jacinto Teixeira Maia. – 2022.
55 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2022.

Orientação: Profa. Ma. Antonia Diana Braga Nogueira.

1. Software-Desenvolvimento. 2. Scrum . 3. Desenvolvimento ágil de software. I. Título.

CDD 005.1

JACKSON JACINTO TEIXEIRA MAIA

DESENVOLVIMENTO DE UMA FERRAMENTA WEB PARA APOIO A
GERÊNCIA DE EQUIPES DE DESENVOLVIMENTO DISTRIBUÍDAS DE
SOFTWARE

Monografia Trabalho de Conclusão de Curso
apresentado ao Curso de Graduação de
Engenharia de Software do Campus de Quixadá
da Universidade Federal do Ceará, como
requisito parcial à obtenção do grau de bacharel
em Engenharia de Software.

Aprovada em: ___/ ___/ ___.

BANCA EXAMINADORA

Prof^ª. Ma. Antônia Diana Braga Nogueira. (Orientadora)
Universidade Federal do Ceará (UFC)

Prof^ª. Dra. Paulyne Matthews Jucá
Universidade Federal do Ceará (UFC)

Prof^º. Dr. Jefferson de Carvalho Silva
Universidade Federal do Ceará (UFC)

A Deus,

Aos meus pais e minha irmã,

A minha esposa e meus filhos.

AGRADECIMENTOS

Sou grato primeiramente a Deus que abençoou todo o caminho por mim trilhado e o de todas as pessoas que estavam à minha volta e por ter-me dado forças para conseguir concluir esse desafio chamado graduação.

Agradeço aos meus pais, por nunca terem desistido de mim e por terem investido na minha criação, acima de qualquer coisa com princípios e valores, estes que levo para o resto da minha vida, seguindo na esperança de conseguir recompensá-los por todo trabalho dedicado a mim

Agradeço a minha irmã, por toda a sabedoria, apoio e bons conselhos, que apesar de mais nova, me ensinou mais do que a vida poderia exigir de mim

Agradeço a minha esposa que me incentivou nos meus momentos mais difíceis, compreendeu minhas ausências, e se dedicou junto comigo desde o início para a realização do meu sonho

Aos meus filhos por serem a principal razão para continuar e por eles conseguirei realizar todos os meus objetivos.

A minha orientadora Dra. Diana pela paciência, por não ter desistido de mim, por toda sua sabedoria que me ajudou a concretizar meu projeto e por estar comigo nesta última oportunidade de concluir a graduação

“Nós só podemos ver um pouco do futuro, mas o suficiente para perceber que há muito a fazer.”

(Alan Turing)

RESUMO

Nos últimos tempos têm existido transformações de mercados locais para mercados globais, em um processo que tem criado novas formas de colaboração e competição. A quantidade de empresas que estão segmentando seu processo de desenvolvimento de software ao redor do mundo é cada vez maior, com o objetivo de ter um aumento da produtividade, redução de custos em mão de obra e melhorias na qualidade. A segmentação geográfica do time de desenvolvimento traz bastante benefícios, mas acarreta malefícios também. Algumas características do desenvolvimento distribuído de *software* dificultam o desenvolvimento de *software*: afastamento geográfica (distância física); dispersão temporal (diferenças de fuso-horário); e diferentes culturas (idioma, tradições, costumes, normas e comportamento); e, principalmente, a má comunicação. Levando isso em consideração, esse trabalho propõe a criação de uma ferramenta que dê suporte a equipe de desenvolvimento de software na gestão das suas tarefas, comunicação e projetos. Essa ferramenta foi desenvolvida utilizando uma adaptação do *framework Scrum* para equipes de um só membro, e esse projeto propõe principalmente detalhar o processo para criação da ferramenta Gine, mostrando desde os ciclos do desenvolvimento até os artefatos gerados através da execução de *sprints* do projeto.

Palavras-chave: *Software-Desenvolvimento. Scrum. Desenvolvimento ágil de software*

ABSTRACT

In recent times there have been transformations from local markets to global markets, in a process that has created new forms of collaboration and competition. The number of companies that are segmenting their software development process around the world is increasing, with the objective of increasing productivity, reducing labor costs and improving quality. The geographic segmentation of the development team has many benefits, but it also has disadvantages. Some characteristics of distributed software development make software development difficult: geographic remoteness (physical distance); temporal dispersion (time zone differences); and different cultures (language, traditions, customs, norms and behavior); and, above all, poor communication. Taking this into account, this work proposes the creation of a tool that supports the software development team in managing its tasks, communication and projects. This tool was developed using an adaptation of the Scrum framework for teams of one member, and this project mainly proposes to detail the process for creating the Gine tool, showing from the development cycles to the artifacts generated through the execution of project sprints.

Keywords: Scrum Solo. Web development. Software process. Distributed software development.

LISTA DE FIGURAS

Figura 1 - Fluxo do processo <i>Scrum</i>	19
Figura 2 - Fluxo do processo <i>Scrum Solo</i>	21
Figura 3 - Atividade de Requisitos do <i>Scrum Solo</i>	22
Figura 4 - Atividade de Sprint do <i>Scrum Solo</i>	23
Figura 5 - Atividade de Entrega do <i>Scrum Solo</i>	23
Figura 6 - Atividade de Gestão do <i>Scrum Solo</i>	24
Figura 7 - Protótipo de baixa fidelidade do sistema.....	32
Figura 8 - Cadastro de responsável	35
Figura 9 - Apresentação dos projetos	37
Figura 10 - Alocação de membros	38
Figura 11 - Listagem de tarefas.....	40
Figura 12 - Calendário de atividades	41
Figura 13 - Diagrama de fluxo.....	44
Figura 14 - Estrutura analítica do projeto.....	45
Figura 15 - Gráfico de Percentual de pessoas que atuam com desenvolvimento de software.....	48
Figura 16 - Gráfico de Percentual Área de desenvolvimento.....	48
Figura 17 - Gráfico de Forma de atuação.....	48
Figura 18 - Gráfico de Disponibilidade dos participantes para responder a 2ª fase.....	49
Figura 19 - Avaliação da usabilidade da ferramenta.....	50
Figura 20 - Avaliação para interface.....	50
Figura 21 - O que os usuários acreditam ser o maior suporte que a plataforma Gine	51
Figura 22 - O que os usuários acreditam ser o menor suporte que a plataforma Gine oferece.....	51

LISTA DE QUADROS

Quadro 1 - Comparação entre os trabalhos relacionados e o trabalho proposto	18
Quadro 2 - Ferramentas analisadas para a coleta de funcionalidades.....	29
Quadro 3 - <i>Product Backlog</i> da ferramenta Gine.....	31
Quadro 4 - Ferramentas e configuração da ferramenta Gine.....	33
Quadro 5 - <i>Sprint backlog</i> – <i>Sprint 1</i>	34
Quadro 6 - <i>Sprint backlog</i> - <i>Sprint 2</i>	36
Quadro 7 - <i>Sprint backlog</i> - <i>Sprint 3</i>	38
Quadro 8 - <i>Sprint backlog</i> - <i>Sprint 4</i>	39
Quadro 9 - <i>Sprint backlog</i> - <i>Sprint 5</i>	40
Quadro 10 - <i>Sprint backlog</i> - <i>Sprint 6</i>	41
Quadro 11 - <i>Sprint backlog</i> - <i>Sprint 7</i>	42
Quadro 12 - Ata de validação <i>Sprint backlog</i> - <i>Sprint 6</i>	43
Quadro 13 – Cronograma do projeto.....	46

SUMÁRIO

1	INTRODUÇÃO.....	14
1.1	Objetivos	16
1.1.1	<i>Objetivo Geral.....</i>	16
1.1.2	<i>Objetivos específicos.....</i>	16
1.2	Organização.....	16
2	TRABALHOS RELACIONADOS.....	17
3	FUNDAMENTAÇÃO TEÓRICA.....	19
3.1	Desenvolvimento distribuído de software.....	19
3.2	Scrum.....	19
3.3	Scrum Solo.....	21
3.3.1	<i>Os itens do Scrum Solo (PAGOTTO, TIAGO et al., (2016)</i>	22
3.3.2	<i>Os atores do Scrum Solo (PAGOTTO, TIAGO et al., (2016)</i>	24
3.3.3	<i>Artefatos do Scrum Solo (PAGOTTO, TIAGO et al., (2016)</i>	25
4	PROCEDIMENTOS METODOLÓGICOS.....	27
4.1	Desenvolvimento.....	28
4.2	Avaliação qualitativa do projeto.....	28
5	DESENVOLVIMENTO.....	28
5.1	Análise de ferrament.....	28
5.2	Desenvolvendo a ferramenta utilizando o Scrum Solo.....	30
5.2.1	<i>Requirement.....</i>	30
5.2.2	<i>Sprint.....</i>	33
5.2.3	<i>Management.....</i>	44
5.3	Avaliação da ferramenta.....	47
5.3.1	<i>Montagem da amostragem.....</i>	47
5.3.2	<i>Avaliação qualitativa.....</i>	49
6	CONSIDERAÇÕES FINAIS.....	52

1 INTRODUÇÃO

Atualmente, em um mercado cada vez mais competitivo, as empresas buscam maior produtividade e rapidez em atingir os seus objetivos. Não difere na área de desenvolvimento de *software* que, nessa corrida competitiva, tenta criar produtos de qualidade, com processos bem definidos, coesos e clientes satisfeitos. No decorrer da linha do tempo da Engenharia de *Software* (ES) surgiram muitos processos, metodologias e atividades que buscam produzir um produto final com maior qualidade, que possa evitar prazos e orçamentos estourados. Nesse contexto surge o conceito de Metodologias Ágeis que tem a intenção de desburocratizar o desenvolvimento, aproximar o cliente ao processo, diminuir cada vez mais a documentação e dar uma maior flexibilidade ao time. (BECK *et al.*, 2001).

As Metodologias Ágeis tornaram-se populares em 2001, quando alguns especialistas em processos de software estabeleceram princípios comuns, sendo criado o que viria a se chamar Aliança Ágil e assim estabelecido o Manifesto Ágil. (BECK *et al.*, 2001).

Dentre as principais metodologias ágeis estão o *Scrum* para gestão e planejamento de projetos de software que é utilizado na gerência de projetos e no desenvolvimento de software ágil, não é um processo que segue fielmente padrões estabelecidos, funciona como um *framework* com valores, princípios e práticas para que cada equipe adapte para a sua realidade de gestão e gerência conforme o tamanho do seu time.

Com a evolução e modernização do desenvolvimento de *software*, se expõem diversas necessidades que vão além da qualidade, como custo, produção e mão de obra qualificada. Nas últimas décadas, grandes investimentos permitiram um movimento de transformação de um mercado local para mercados globais, num processo que tem criado novas formas de colaboração e competição (HERBSLEB *et al.*, 2001). O número de empresas que estão segmentando seu processo de desenvolvimento de *software* ao redor do mundo é cada vez maior, visando ganho de produtividade, redução de custos e melhorias na qualidade (PRIKLADNICKI; AUDY, 2004).

O deslocamento de um mercado nacional para um mercado global gera novas formas de competição e cooperação que cruzam os limites nacionais, isto tem um profundo impacto não só na distribuição de produtos de *software*, mas na forma como eles são concebidos, projetados, construídos, testados e entregues aos clientes (HERBSLEB; MOITRA, 2001).

(Herbsleb e Moitra, 2001) também afirmam que os desafios existentes neste novo contexto estão geralmente relacionados não apenas com questões técnicas, mas também com questões estratégicas, culturais, e de gestão de conhecimento.

Segundo (Carmel, 1999), não existem dúvidas para qualquer profissional que trabalha na área de ES que tanto o desenvolvimento de *software* tradicional quanto o distribuído possuem diversas dificuldades. As principais características que os diferenciam são: dispersão geográfica (distância física); dispersão temporal (diferenças de fuso-horário); e diferenças culturais (idioma, tradições, costumes, normas e comportamento). A comunicação inadequada ou malfeita a tempos está entre as maiores causas de falha em projetos destacados por (DWIVEDI *et al.*, 2015).

De acordo com (DAMIAN; MOITRA 2006), mesmo que já tenha se obtido uma gama de conhecimento sobre desenvolvimento distribuído, ainda existe um entendimento a ser alcançado, métodos e técnicas a serem desenvolvidas e práticas a serem amadurecidas.

Segundo (Huzita et al. 2008), estratégias, soluções e práticas para tornar esta abordagem um sucesso torna-se imperativas. O desenvolvimento de ambientes, modelos e ferramentas para gerenciar processos de *software* neste contexto tornam-se cada vez mais importantes.

Este trabalho tem como objetivo detalhar o desenvolvimento de uma ferramenta *Web*, utilizando uma adaptação do *Scrum* no desenvolvimento da ferramenta, e detalhar todos os resultados das fases de desenvolvimento, artefatos gerados e decisões tomadas serão apresentadas neste trabalho.

O *software* resultado desse trabalho tem como escopo geral a função de auxiliar equipes de desenvolvimento geograficamente distribuídas, de modo a facilitar a comunicação e interação dos membros. A aplicação busca apoiar a gestão da equipe dos projetos, unindo pontos fortes e pontos fracos de ferramentas já existentes, além de oferecer suporte aos principais desafios existentes no desenvolvimento distribuído de *software* (DDS).

1.1 OBJETIVOS

Nesta Seção serão apresentados o objetivo geral e os objetivos específicos deste trabalho.

1.1.1 Objetivo geral

O objetivo deste trabalho é apresentar o desenvolvimento de um *software* direcionado a equipes distribuídas utilizando uma adaptação do *Scrum*, mostrando os resultados de cada etapa, artefatos desenvolvidos e decisões tomadas durante todo o projeto com intenção de apresentar a aplicação desse processo.

1.1.2 Objetivos específicos

- Aplicar uma adaptação do *Scrum* para equipes de um só indivíduo no desenvolvimento do *software*
- Elaborar uma ferramenta que possa auxiliar a gestão de equipes que atuam a distância e que venha sanar os principais problemas do DDS que segundo (Herbsleb et al. 2001) é comunicação, rastreamento de tempo e respeito a estrutura hierárquica da equipe.
- Aplicar uma avaliação no *software* baseado em entrevista para validar a qualidade do sistema desenvolvido.

1.2 ORGANIZAÇÃO

Este trabalho está organizado conforme segue: a Seção 2 apresenta os trabalhos relacionados, onde serão discutidas as propostas dos trabalhos relacionados e como eles se assemelham e se diferenciam da proposta aqui apresentada. A Seção 3 apresenta um resumo sobre os conceitos técnicos e teóricos necessários para a realização do trabalho. A Seção 4 mostra a descrição do procedimento metodológico que foi realizado no trabalho. Na Seção 5, é apresentado o desenvolvimento da aplicação. Por fim, na Seção 6 conclui-se o trabalho.

2 TRABALHOS RELACIONADOS

Para que se possa obter as principais carências na gerência de equipes que trabalham de modo distribuído, foi necessária uma análise nas principais ferramentas direcionadas a esse cenário analisando os principais pontos que possam oferecer suporte a comunicação, rastreamento de tempo e respeito a estrutura hierárquica da equipe que, segundo (Herbsleb et al. 2001), são os principais problemas causados pelas divergências culturais existentes no Desenvolvimento Distribuído. Tendo em vista isso, o trabalho baseia-se em informações, métodos, conceitos e técnicas coletadas de trabalhos científicos coletados das áreas de desenvolvimento distribuído de *software*. Para atender os desafios dessa modalidade de desenvolvimento citados, foram analisados os trabalhos descritos abaixo.

(Cavalcanti, Maciel e Albuquerque, 2009) apresentam uma ferramenta de código livre que dá suporte a equipes distribuídas que usam o *Scrum* chamada *FireScrum*. Os autores fazem uma análise de modo geral no *software* que vai desde a sua arquitetura até *interface* gráfica, detalhando todas as funcionalidades e os benefícios na aplicação do *Scrum* por equipes que trabalham a distância. Através dessa análise, coletam-se dados para uma comparação com 5 outras ferramentas utilizando as atividades do *Scrum* como métrica de qualidade.

O estudo foi realizado com a intenção de demonstrar, além da ferramenta que foi desenvolvida, um pouco do processo de desenvolvimento dessa ferramenta. O trabalho se assemelha ao que está sendo produzido pelo fato de se tratar de uma ferramenta direcionada a equipes de DDS, porém não descreve com detalhes o processo de criação da ferramenta em questão, o que é um dos focos principais desta obra.

(Huzita et al. 2007) apresenta o ambiente DiSEN (*Distributed Software Engineering Environment*), que demonstra ter como principal recurso oferecer suporte a comunicação, persistência e cooperação em equipes de DDS. Os autores falam sobre o DDS relatando as principais preocupações que equipes distribuídas geograficamente devem ter. Não detalham o processo de desenvolvimento do *software*, porém, fazem uma descrição geral do ambiente DiSEN explicitando a arquitetura a qual o sistema foi baseado, definem o apoio da ferramenta a infraestrutura do ambiente aplicado, o apoio a comunicação e apoio a persistência.

O trabalho descrito tem uma breve semelhança com o trabalho que está sendo produzido no sentido de apresentar uma plataforma de gerência de equipe distribuída,

mas não descreve um processo de desenvolvimento e não apresenta nenhum tipo de validação de qualidade do uso da ferramenta.

(Brunhera e Zanata, 2010) apresentam o *Scrum* descrevendo todas as suas fases e ciclos, definindo os seus papéis e detalhando o funcionamento de modo geral dessa metodologia ágil. O trabalho apresenta uma análise do processo criado pela organização descrevendo sobre todo o ciclo de vida do processo de modo minucioso e detalhado.

(Brunhera e Zanata, 2010) também demonstram uma aplicação desse processo em um estudo de caso, que mostra na prática como funciona a aplicação do processo da organização em um caso de desenvolvimento real de uma ferramenta para gerenciar a equipe de desenvolvimento.

A obra de (Brunhera e Zanata, 2010) se assemelha ao trabalho desenvolvido por apresentar o funcionamento de um processo de adaptação do *Scrum* existente e também pelo fato de colocar esse processo em ação para o desenvolvimento de um *software*, porém, em contrapartida, se diferencia no que se refere ao produto que está sendo criado.

Quadro 1 – Comparação entre os trabalhos relacionados e o trabalho proposto

	FOCADA NO DESENVOLVIMENTO O DISTRIBUÍDO DE SOFTWARE	DETALHAMENTO DO PROCESSO DE DESENVOLVIMENTO	CONTRIBUI NA GERÊNCIA DE EQUIPES
(Cavalcanti, Maciel e Albuquerque, 2009)	X		X
(Brunhera e Zanata, 2010)		X	X
(Huzita et al. 2007)	X		X
Presente Trabalho	X	X	X

Fonte: Elaborado pelo autor.

3 FUNDAMENTAÇÃO TEÓRICA

Nessa Seção serão abordados os principais conceitos relacionados a este trabalho e qual a contribuição de cada conceito para o desenvolvimento do trabalho.

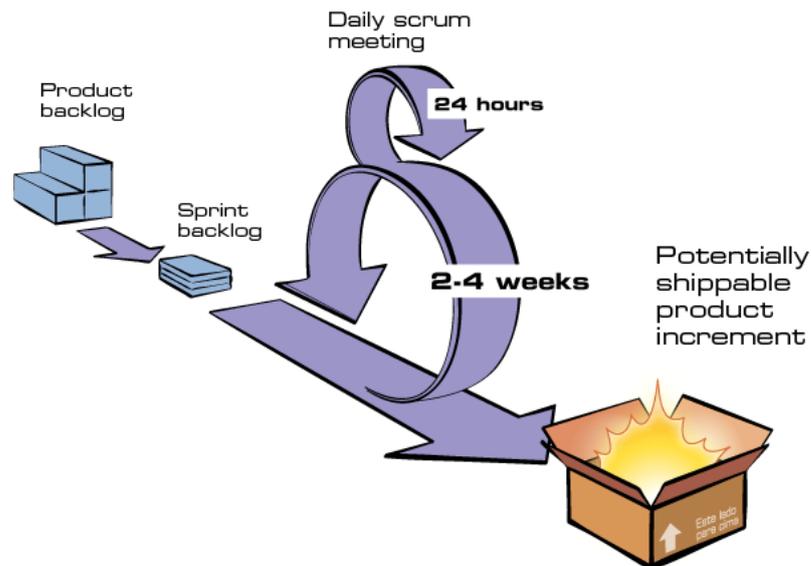
3.1 Desenvolvimento distribuído de software

Os últimos anos têm mostrado que a tendência do desenvolvimento de software é a distribuição global. De acordo com (Herbsleb et al. 2001) cada vez mais empresas de desenvolvimento de software tem demonstrado movimentos que as levaram em direção à globalização dos negócios e dos serviços intensivos. A chegada dessa globalização no meio do desenvolvimento de *software* não muda só a maneira de fazer negócio, mas também a maneira de produzir os produtos. Também de acordo com (Herbsleb et al. 2001) essa mudança está tendo um profundo impacto não apenas no *marketing* e distribuição, mas também na forma como os produtos são criados, projetados, construídos, testados e entregues aos clientes. E a partir dessas mudanças na criação do produto de *software* surge essa metodologia de desenvolvimento a distância que trata de equipes separadas geograficamente trabalhando em um único projeto e sendo geridas a distância.

O desenvolvimento distribuído de *software* será aplicado neste trabalho com a intenção de fundamentar a ferramenta que será desenvolvida.

3.2 Scrum

Figura 1 - Fluxo do processo *Scrum*



Fonte: (desenvolvimentoAgil.com.br)

O *Scrum* é uma metodologia ágil para gestão e planejamento de projetos de *software*. O Manifesto Ágil (BECK *et al.*, 2001) propõe a ideia de um processo baseado em iterações no desenvolvimento de *software*. O *Scrum*, por ser uma metodologia ágil, segue as diretrizes do manifesto e, portanto, pode ser caracterizado como um *framework* para desenvolvimento de *software* que trabalha com iterações. Na proposta padrão o *Scrum* possui três integrantes: o *Product Owner*, *Scrum Master* e o Time e tem as suas iterações nomeadas de *Sprints*.

As fases, artefatos e reuniões padrões do *Scrum* são:

- *Product Backlog*
- *Sprint Planning Meeting*
- *Sprint Backlog*
- *Sprint*
- *Daily Scrum*
- *Sprint Review Meeting*
- *Sprint Retrospective*

O *Scrum* será usado neste trabalho como embasamento para o escopo da ferramenta que será desenvolvida. Assim esse processo servirá como base do fluxo do processo e da gestão desse processo dentro do *software* que será elaborado

3.3 Scrum Solo

O *Scrum Solo* é uma adaptação do *framework Scrum* para desenvolvedores que atuam individualmente. Essa adaptação foi criada em 2012, os autores (Pagotto, Tiago et al., 2016) uniram o *Scrum* com o *Personal Software Process (PSP)* que segundo (Pagotto, Tiago et al., 2016) é um processo de melhoria que foi criado para dar auxílio aos desenvolvedores administrar e aperfeiçoar a sua competência para produzir *software* de qualidade. O *Scrum Solo* possui grandes semelhanças com *Scrum*, porém com diferenças para que o desenvolvimento individual possa se manter ágil e dinâmico. A Figura 3 apresenta o ciclo de desenvolvimento do *Scrum Solo*. Diferentemente do ciclo apresentado na Figura 2, o *Scrum Solo* considera os ciclos de desenvolvimento de uma semana, não há reuniões diárias, possui entrega de parte do projeto ao final, inclui um de validação e uma reunião de orientação.

Figura 2 - Fluxo do processo *Scrum Solo*



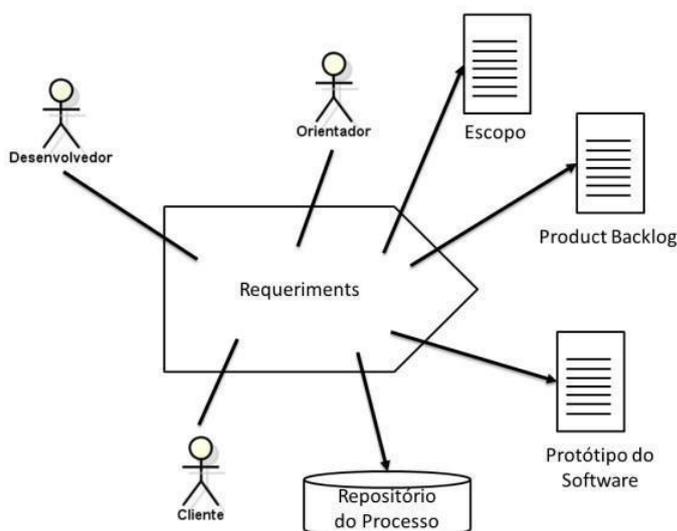
Fonte: Pagotto, Tiago et. al (2016)

3.3.1 Os itens do *Scrum Solo* (PAGOTTO, TIAGO et. al (2016):

3.3.1.1 *Requirement*

Objetiva definir o escopo do produto, caracterizar o cliente do produto e definir o *product backlog*. As informações coletadas com o cliente e orientador (pessoa que possui grande conhecimento sobre o processo de *software*) são usadas como entradas e, os artefatos gerados nessa atividade são: escopo, *product backlog* e protótipo do *software* (Figura 4). Percebe-se que todos os artefatos gerados são armazenados em um repositório de dados. Os autores recomendam que este repositório permaneça em uma nuvem.

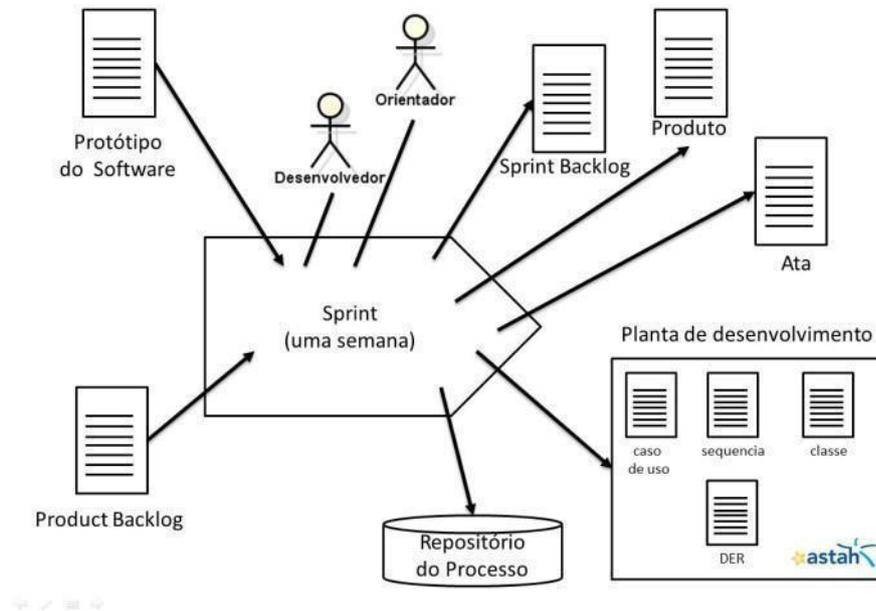
Figura 3 - Atividade de Requisitos do *Scrum Solo*



Fonte: PAGOTTO, Tiago et. al (2016)

3.3.1.2 *Sprint*:

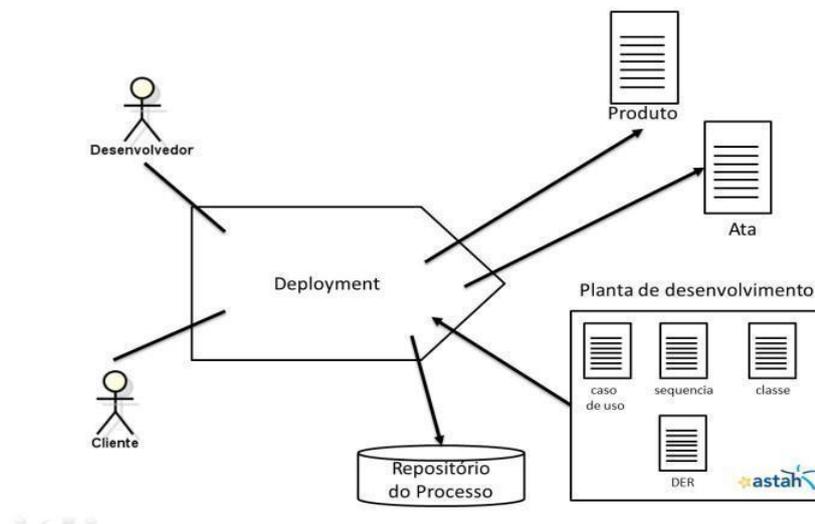
Objetiva desenvolver o conjunto de itens selecionados a partir do *product backlog* em duração máxima de uma semana. O *product backlog* e o protótipo do *software* são usados como entrada e, os artefatos gerados nessa atividade são: *sprint backlog*, produto, ata e planta de desenvolvimento (Figura 5).

Figura 4 Atividade de Sprint do *Scrum* Solo

Fonte: PAGOTTO, Tiago et. al (2016)

3.3.1.3 Deployment

Objetiva disponibilizar o produto para uso do cliente. A planta de desenvolvimento é usada como entrada e, os artefatos gerados nessa atividade são: produto e ata de validação.

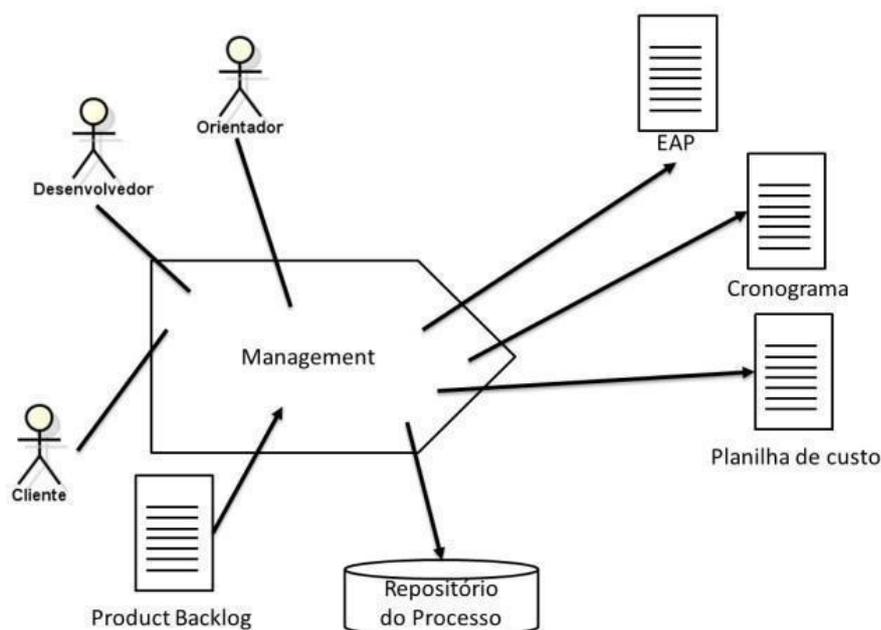
Figura 5 - Atividade de Entrega do *Scrum* Solo

Fonte: PAGOTTO, Tiago et. al (2016)

3.3.1.4 Management

Objetiva planejar, monitorar e controlar o desenvolvimento do produto. O *product backlog* é usado como entrada e, os artefatos gerados nessa atividade são: estrutura analítica do projeto (EAP), cronograma, planilha de custo e planilha de controle.

Figura 6 - Atividade de Gestão do *Scrum* Solo



Fonte: (Pagotto, Tiago et al., 2016)

3.3.2 Os atores do *Scrum* Solo (Pagotto, Tiago et al., 2016)

3.3.2.1 Product Owner

Caracterizado como proprietário do produto. Ele irá interagir diretamente com o produto e pode ser caracterizado como uma única pessoa, por exemplo: gerente, contador, secretária; ou, como um grupo de usuários, por exemplo: pessoas que necessitam de um aplicativo para verificar a disponibilidade de ônibus em todo o país.

3.3.2.2 Desenvolvedor

Responsável por executar o processo e construir o produto.

3.3.2.3 Orientadora

Caracterizado como um consultor que conhece a fundo o processo. Este possui uma visão ampla da tecnologia utilizada para o desenvolvimento do produto e do escopo do projeto.

3.3.2.4 Grupo de validação

Possíveis usuários do produto gerado. Este deve participar da validação do produto, a qual deve constar na ata (artefato do processo).

Essa metodologia para desenvolvedores individuais será usada como o processo para a criação da Gine com o intuito de trazer uma maior qualidade e, ao mesmo tempo, agilidade ao processo de desenvolvimento.

Artefatos do *Scrum* Solo (Pagotto, Tiago et al., 2016)

3.3.2.5 Escopo

Caracteriza o escopo do processo e os aspectos inerentes ao mapeamento dos problemas do *product owner*. Descreve os principais pontos do *software* (aplicados na solução de problemas), o perfil do cliente e os itens da *product backlog* (requisitos funcionais).

3.3.2.6 Protótipo de *software*

Coleciona as telas para acesso e manipulação de dados do *software*, além das *interfaces* dos relatórios. Deve-se ressaltar que todos os arquivos inseridos devem estar no formato .PNG e que é importante apontar o nome do item da *product backlog* que representa a tela ou a *interface* de relatório – vide artefato na nuvem do processo. A nuvem que contém os artefatos do processo pode ser acessada através do endereço: <https://goo.gl/pf7NU2>. A partir deste ponto, todos os artefatos apresentados e referenciados neste item podem ser encontrados na subpasta “*TEMPLATES*” com seus respectivos indicadores.

3.3.2.7 Product backlog

Caracterizado como uma lista de funcionalidades que devem ser implementadas no *software*. Deve ser limitada ao código da funcionalidade, a descrição, a data da inserção e a data de seleção para a *sprint backlog*. Nota-se que a data da inserção só será preenchida quando a funcionalidade for selecionada.

3.3.2.8 Repositório do processo

Caracterizado como um serviço de armazenamento de arquivos na nuvem, este objetiva armazenar todos os artefatos gerados durante a execução do processo. Este repositório deve ser organizado em diretórios e o conjunto de regras para a organização dos diretórios é de responsabilidade do desenvolvedor.

3.3.2.9 Sprint backlog

Armazena o conjunto de funcionalidades que devem ser implementadas durante aquela *sprint*. O *sprint backlog* armazena o código da funcionalidade (mesmo do *product backlog*) e o *link* para acesso à planta de especificação daquela funcionalidade. Esta planta é caracterizada pelos diagramas de: casos de uso, sequência, classes, e entidade e relacionamento. É válido destacar que todos os diagramas devem ser armazenados no repositório do processo. O referido artefato possui data de inserção da funcionalidade no *sprint backlog*, tempo de construção (orçado ou previsto e realizado ou real), data de validação e, por fim, se a funcionalidade é fruto de um retrabalho.

3.3.2.10 Produto ou parte dele em funcionamento

Versão do produto que possibilite ao cliente obter um retorno sobre o investimento feito na compra do *software*.

3.3.2.11 Ata

Objetiva registrar a validação da implantação de uma funcionalidade. É utilizada no *sprint* e na entrega. No *sprint*, a funcionalidade é validada pelo orientador; já na entrega, a mesma é validada pelo cliente.

3.3.2.12 Planta de desenvolvimento

Objetiva aglutinar os artefatos que foram utilizados na especificação das funcionalidades. O *Scrum* Solo sugere que sejam utilizados os diagramas de: casos de uso, sequência, classes, e entidade e relacionamento. Porém, de acordo com a especificidade do projeto, esta planta pode ser customizada. Uma ferramenta que possibilita a construção dos referidos diagramas é o Astah.

3.3.2.13 Estrutura Analítica do Projeto (EAP)

Organograma que objetiva apresentar o escopo do projeto. No topo deste organograma é encontrado o nome do projeto, logo abaixo, as atividades e, posteriormente, os pacotes de trabalho. Estes últimos caracterizam formalmente todo o trabalho que será feito durante a execução do projeto.

3.3.2.14 Cronograma

Organizar sequencialmente os pacotes de trabalho dentro de um determinado espaço de tempo determinado. Nele, pode-se apontar o responsável pela execução de cada atividade. O *Scrum* Solo sugere que se utilize o cronograma no formato de diagrama de Gantt.

3.3.2.15 Planilha de custo

Mapeia o custo efetivo gerado durante a execução do projeto. Ao final, tem-se, de forma consistente, a visão de uma comparação entre o orçamento realizado previamente e os custos e tempos reais gastos no projeto.

4 PROCEDIMENTOS METODOLÓGICOS

Nesta Seção serão apresentadas as etapas dos procedimentos metodológicos, que são as coletas de informações para montar escopo do projeto, e do desenvolvimento que são baseadas nas fases do *Scrum* Solo (Pagotto, Tiago et al., 2016), que foi o processo usado para o desenvolvimento do Gine e a avaliação qualitativa do projeto.

4.1 Desenvolvimento

Essa etapa foi feita aplicando o *Scrum Solo* e executando as *sprints* estabelecidas na estrutura do processo e, desse modo, ocorre o desenvolvimento das funcionalidades propostas em cada ciclo de desenvolvimento.

A aplicação foi desenvolvida para a plataforma *Web*.

4.2 Avaliação qualitativa do projeto

A segunda parte trata-se da elaboração de um questionário referente à atuação do entrevistado em times de desenvolvimento de *software*, tendo 9 questões com objetivo de filtrar uma amostragem de pessoas que atuam como gerentes de projetos, supervisores, líderes técnicos, desenvolvedores e demais profissionais que exercem cargos em times de desenvolvimento de *software* de empresas que atuam à distância. O questionário foi aplicado através da plataforma gratuita Google Forms.

Na segunda etapa as pessoas selecionadas na etapa anterior passaram por um questionário com análise qualitativa sobre a usabilidade e os recursos existentes na ferramenta desenvolvida para equipe de desenvolvimento de software que atua de forma distribuída. Os profissionais foram submetidos a uma navegação completa no *software* onde puderam ver as funcionalidades da plataforma Gine.

5 DESENVOLVIMENTO

Nesta Seção serão apresentados os passos para o desenvolvimento da ferramenta.

5.1 Análise de ferramentas

Nesta Seção são analisadas as ferramentas já existentes no mercado. Essas ferramentas foram escolhidas levando-se em consideração sua contribuição para melhorar a comunicação e a gestão de tarefas de projetos de desenvolvimento de *software* e que

pudessem servir como referência de recursos para ser implementado no projeto em desenvolvimento.

Abaixo estão listadas as ferramentas avaliadas

Quadro 2 - Ferramentas analisadas para a coleta de funcionalidades

FERRAMENTA AVALIADA	ENDEREÇO	FUNCIONALIDADES DESTACADAS
Assana	https://asana.com/pt	Lista de atividade Apresentação do cronograma
IDoneThis	https://idonethis.com/	Relatório automático de atividades diárias Relatório ao vivo das atividades Check-in diário do membro da equipe Relatório de presença
Trello	https://trello.com/	Apresentação das atividades em quadros Etiquetas na atividade
Basecamp	https://basecamp.com/how-it-works	Atribuir tarefas a uma ou mais pessoas Adicionar notas e anexos de arquivo Comentar diretamente em tarefas e listas
Podio	https://podio.com/	Filtro de busca de atividades
Evernote	https://evernote.com/intl/pt-br/	Lista de notas pessoais para cada membro
Slack	https://slack.com/	Conversas privadas Conversas em grupo de projeto
ScrumBrl	http://www.scrumbl.ca/?lang=en_gb	Organização e apresentação Scrum

Fonte: Elaborado pelo autor.

5.2 Desenvolvendo a ferramenta utilizando *Scrum Solo*

Nesta Seção será apresentada a aplicação do processo *Scrum Solo* para desenvolvimento da aplicação Gine, apresentando todos os artefatos gerados durante a execução do processo. O projeto baseia toda a sua documentação e artefatos nos *templates* estabelecidos no *scrum solo* (Pagotto, Tiago et al., 2016).

5.2.1 *Requirement*

De acordo com a organização do processo, a primeira atividade do *Scrum Solo* é *Requirement*. Essa fase tem por objetivo definir o escopo do produto, caracterizar o cliente do produto e definir o *product backlog*.

Como o projeto trata-se de um *software* de prateleira, o cliente se definiu como a própria equipe de desenvolvimento (desenvolvedor) que define com base do desenvolvimento suprir necessidade do DDS que, de acordo com (Herbsleb et al., 2001), é comunicação, rastreamento de tempo e respeito a estrutura hierárquica da equipe.

Com essa aplicação, o time de desenvolvimento poderá gerenciar de forma básica o desenvolvimento de software a distância, conseguindo estabelecer um time auto gerenciável e com alto nível de rastreabilidade de tempo por atividades e projetos.

Ao final dessa atividade, foram gerados o escopo do projeto: *product backlog* e protótipo do *software*.

Quadro 3 - Product Backlog da ferramenta Gine

ID	Descrição	Link para arquivo digital contendo a gravação	Data da inserção	Data de seleção para a Sprint
#001	Cadastro de responsável		01/07/2018	07/03/2022
#002	Gerenciamento de perfil		01/07/2018	07/03/2022
#003	Login		01/07/2018	07/03/2022
#004	Criação de projeto		01/07/2018	14/03/2022
#005	Gerenciamento de projeto		01/07/2018	14/03/2022
#006	Lista projetos		01/07/2018	14/03/2022
#007	Visualização dos projetos		01/07/2018	14/03/2022
#008	Criação de membro		01/07/2018	21/03/2022
#009	Alocação de membros a projetos		01/07/2018	21/03/2022
#010	Criação de tarefas		01/07/2018	28/03/2022
#011	Gerenciamento de tarefas		01/07/2018	28/03/2022
#012	Listagem de tarefas		01/07/2018	28/03/2022
#013	Listagem de tarefas por usuários		01/07/2018	28/03/2022
#014	Apresentação do progresso do projeto		01/07/2018	04/04/2022
#015	Apresentação de calendário de atividades		01/07/2018	04/04/2022
#016	Rastreamento de tempo do projeto		01/07/2018	11/04/2022

#017	Apresentar relatórios de projetos		01/07/2018	11/04/2022
#018	Check-in		01/07/2018	11/04/2022
#019	Check-out		01/07/2018	11/04/2022
#020	Chat		01/07/2018	18/04/2022

Fonte: Elaborado pelo autor.

Todos os itens listados no *product backlog* foram cadastrados com um *ID* único para sua identificação, uma descrição sobre a proposta de funcionalidade, data de inserção no documento do *product backlog* e data de seleção para a *sprint* respectivamente.

Figura 7 Protótipo de baixa fidelidade do sistema



Fonte: Elaborado pelo autor.

5.2.2 *Sprint*

Nessa segunda atividade do *Scrum Solo*, tem-se como objetivo criar o conjunto de itens que serão selecionados a partir da *product backlog* com duração de uma semana. O *product backlog* e o *protótipo* do software são usados como entrada e, os artefatos gerados nessa atividade são: *sprint backlog*, produto, ata e planta de desenvolvimento.

As *sprints* do projeto desenvolvido seguiram fielmente o prazo de uma semana. O desenvolvedor seguiu a rotina de teste unitário durante o desenvolvimento de cada funcionalidade. Durante o final de cada *sprint* se fazia um teste de usabilidade na aplicação, simulando dados e informações.

As ferramentas e tecnologias utilizadas no decorrer dos ciclos de desenvolvimento estão listadas abaixo no quadro 4.

Quadro 4 - Ferramentas e configuração da ferramenta Gine

NOME	DESCRIÇÃO
Laravel	Framework Php utilizado no desenvolvimento de back-and do projeto
Bootstrap	Framework web que foi utilizado no desenvolvimento de componentes de interface e front-end da aplicação
GitHub	Sistema de versionamento de código
Google Drive	Armazenamento na nuvem onde fica o repositório com todos os artefatos do projeto

Fonte: Elaborada pelo autor.

5.2.2.1 *Sprint 1*

Nesta *sprint* foram executadas as primeiras atividades do desenvolvimento que tiveram como artefatos de entrada para auxiliar o desenvolvimento o *product backlog* e o *protótipo*. Por ser a *Sprint* inicial, o projeto partiu do zero, tanto *back-end e front-and*, foi estabelecido o início do desenvolvimento de um painel de acordo com o que foi pensado

no protótipo, onde foi montado a ideia e organização do layout, trabalhando com elementos já existentes do framework *bootstrap*.

Quadro 5 - Sprint backlog - Sprint 1

ID	Descrição	Data de inserção	Tempo de construção (em horas)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
	Elaboração do dashboard	07/03/2022	16	16	12/03/2022	Sim
#001	Cadastro de responsável	07/03/2022	14	14	12/03/2022	Não
#002	Gerenciamento de perfil	07/03/2022	8	16	12/03/2022	Sim
#003	Login	07/03/2022	2	2	12/03/2022	Não

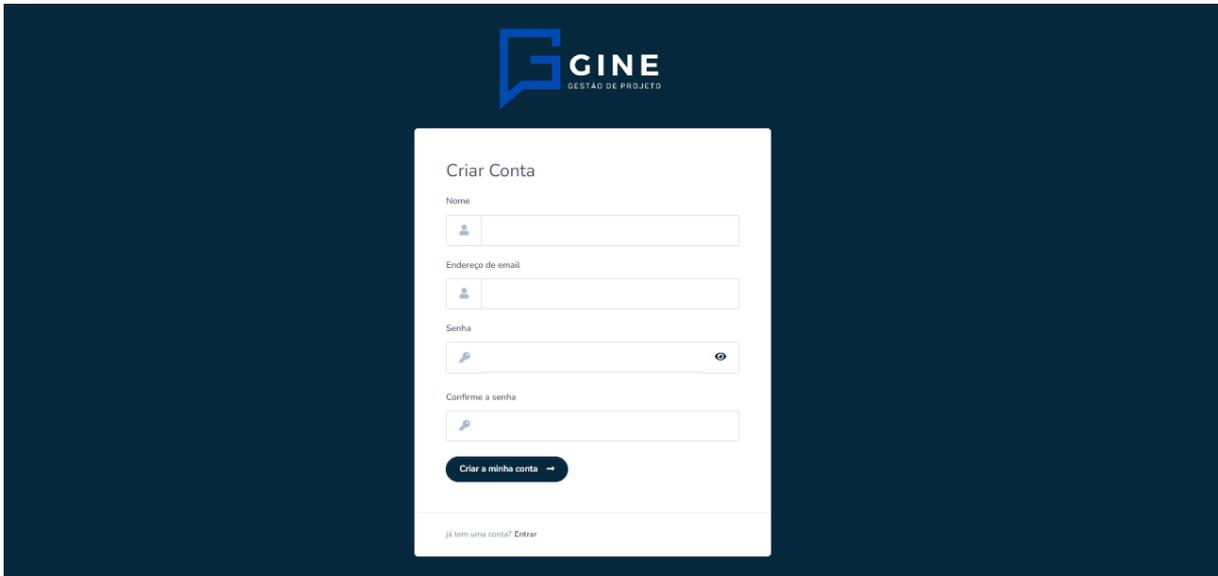
Fonte: Elaborado pelo autor.

A primeira atividade da *Sprint backlog* foi a criação de cadastro de times de desenvolvimento que é feito através do responsável pelo time inicialmente. Nesse marco também foi feito o sistema de gestão de perfil e um mecanismo de login para identificar cada usuário.

Sobre o trabalho investido na carga de trabalho dos requisitos, estimou-se um prazo de 40h, uma semana de trabalho, para a conclusão, tendo início no dia 07/03/2022

e término no dia 11/03/2022, mas foram necessárias 48h. A validação ocorreu no dia 12/03/2022, na atividade do processo *Deployment* e foi necessário passar por retrabalho.

Figura 8 - Cadastro de responsável



A imagem mostra a interface de usuário para a criação de uma conta no sistema GINE. No topo, há o logotipo da GINE, com o texto "GINE GESTÃO DE PROJETO". Abaixo, o formulário "Criar Conta" contém os seguintes campos:

- Nome: Campo de texto com ícone de pessoa.
- Endereço de email: Campo de texto com ícone de envelope.
- Senha: Campo de texto com ícone de cadeado e botão de alternância de visibilidade.
- Confirme a senha: Campo de texto com ícone de cadeado.

Abaixo dos campos, há um botão "Criar a minha conta" com uma seta para a direita. Na base do formulário, há o link "já tem uma conta? Entrar".

Fonte: Elaborado pelo autor

5.2.2.2 *Sprint 2*

Quadro 6 - Sprint backlog - Sprint 2

ID	Link para planta de especificação	Data de inserção	Tempo de construção (em horas)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#004	Criação de projeto	14/03/2022	4	4	19/03/2022	Não
#005	Gerenciamento de projeto	14/03/2022	16	20	19/03/2022	Sim
#006	Lista projetos	14/03/2022	2	2	19/03/2022	Sim
#007	Visualização dos projetos	14/03/2022	18	18	19/03/2022	Não

Fonte: Elaborado pelo Autor

Essa *Sprint* teve sua carga de trabalho estipulada em 40h, uma semana de trabalho, para a conclusão, tendo início no dia 14/03/2022 e término no dia 18/03/2022, mas só foram utilizadas 32h. A validação ocorreu no dia 19/03/2022, na atividade do processo *Deployment* e foi necessário passar por retrabalho.

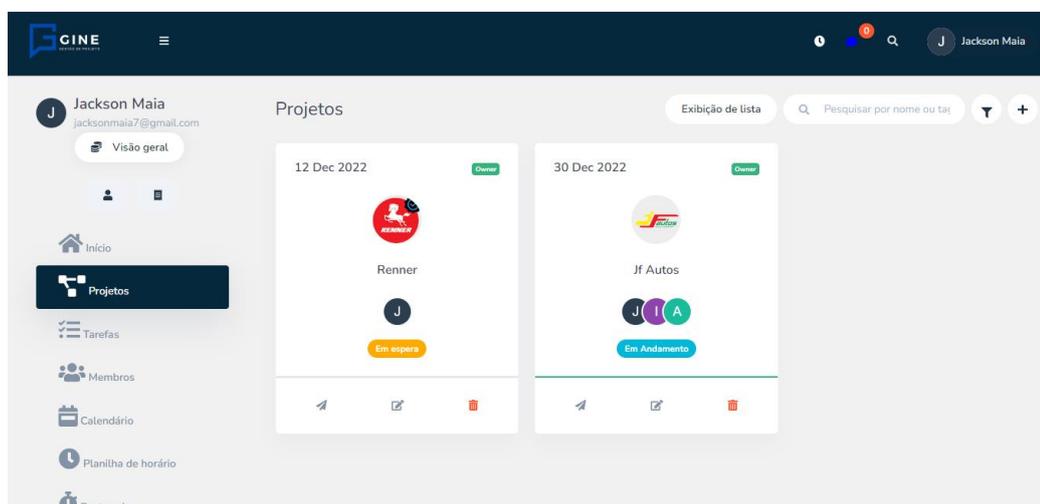
Duas funcionalidades passaram por retrabalho nessa *sprint*: as atividades de gerenciamento de projeto e a listagem de projeto, na primeira atividade de *id* #005 existia uma maior complexidade, pois foi necessário levar em consideração ações básicas de cadastro, edição, exclusão e, além disso, associar o projeto a um proprietário que seria a

pessoa responsável pelo projeto. Surgiram problemas nessa atribuição de responsabilidade, mas foram corrigidos.

Na visualização dos projetos, funcionalidade do *id* #007, havia sido elaborada uma proposta de apresentação arredondada que dificultou o entendimento do usuário quando dos recursos do módulo de projeto, então foi corrigido.

Essa *Sprint* foi planejada para uma duração de 40h, porém, a carga horária investida durou mais do que o planejado e teve 44h de tempo investido na sua execução.

Figura 9 - Apresentação dos projetos



Fonte: Elaborado pelo Autor

5.2.2.3 Sprint 3

Quadro 7 - Sprint backlog - Sprint 3

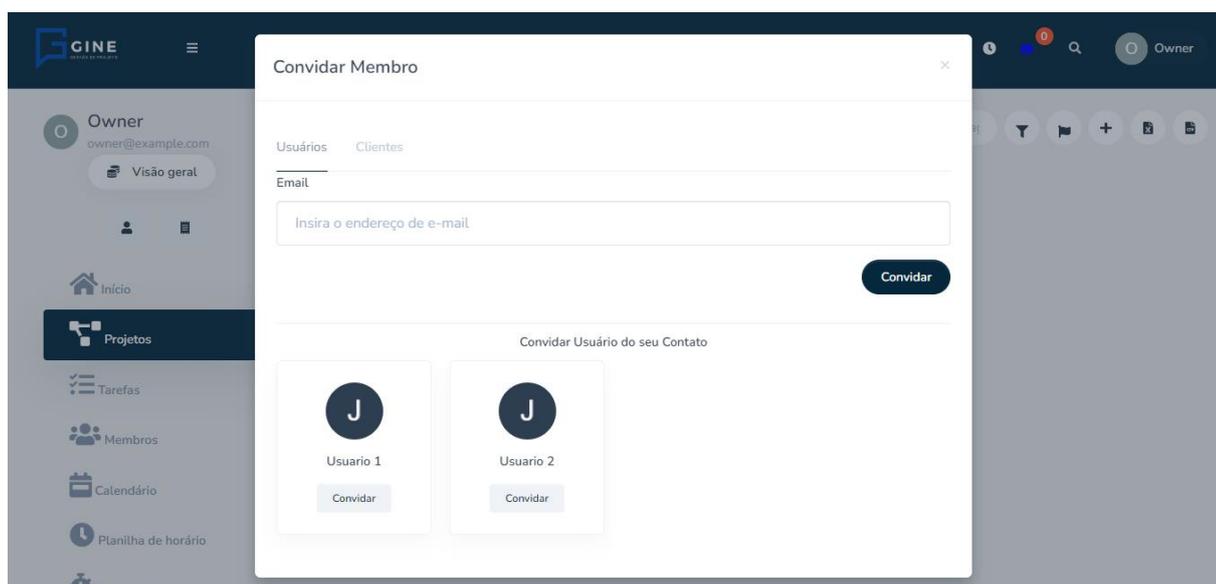
ID	Link para planta de especificação	Data de inserção	Tempo de construção (em horas)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#008	Criação de membro	21/03/2022	20	12	26/03/2022	Não
#009	Alocação de membros a projetos	21/03/2022	20	16	26/03/2022	Não

Fonte: Elaborado pelo Autor

A *sprint 3* se caracterizou pela elaboração das funcionalidades de criação dos membros do time e alocação desses membros no projeto, essa funcionalidade não teve alto grau de complexidade e durou bem menos que a quantidade de horas planejadas.

A carga de tempo necessária para a execução dessa *sprint* foi de 28h, teve duração entre os dias 21/03/2022 ao 24/03/2022.

Figura 10 - Alocação de membros



Fonte: Elaborado pelo autor

5.2.2.4 Sprint 4

Quadro 8 - Sprint backlog - Sprint 4

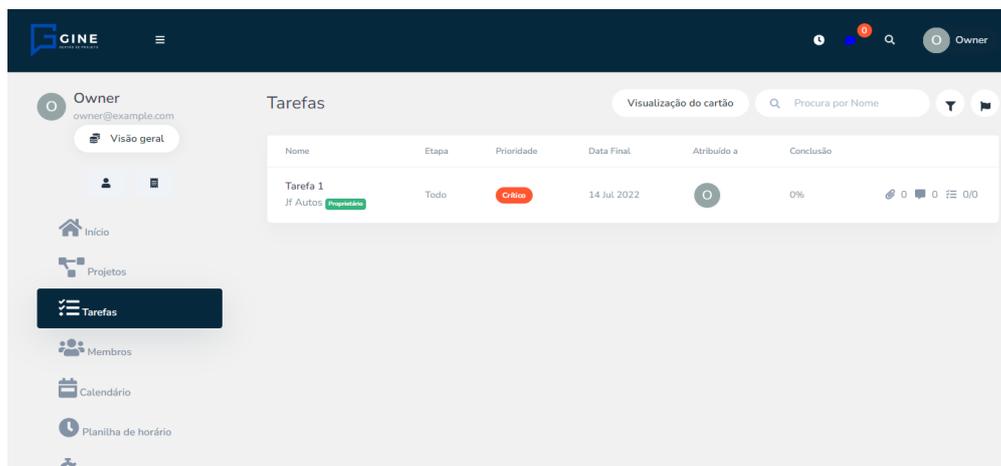
ID	Link para planta de especificação	Data de inserção	Tempo de construção (em horas)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#010	Criação de tarefas	28/03/2022	8	8	02/04/2022	Não
#011	Gerenciamento de tarefas	28/03/2022	20	16	02/04/2022	Não
#012	Listagem de tarefas	28/03/2022	4	4	02/04/2022	Sim
#013	Listagem de tarefas por usuários	28/03/2022	8	6	02/04/2022	Sim

Fonte: Elaborado pelo Autor

Essa *Sprint* teve sua carga de trabalho estipulada em 40h, para a conclusão, tendo início no dia 28/03/2022 e término no dia 02/04/2022, mas só foram utilizadas 34h.

Duas funcionalidades relacionadas à listagem de tarefas tiveram retrabalho, devido à dinâmica de acessos das tarefas gerais do proprietário do projeto e das tarefas alocadas a ele como desenvolvedor.

Figura 11 - Listagem de tarefas



Fonte: Elaborado pelo Autor

5.2.2.5 Sprint 5

Quadro 9 - Sprint backlog - Sprint 5

ID	Link para planta de especificação	Data de inserção	Tempo de construção (em horas)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#014	Apresentação do progresso do projeto	04/04/2022	20	20	09/04/2022	Não
#015	Apresentação de calendário de atividades	04/04/2022	20	18	09/04/2022	Não

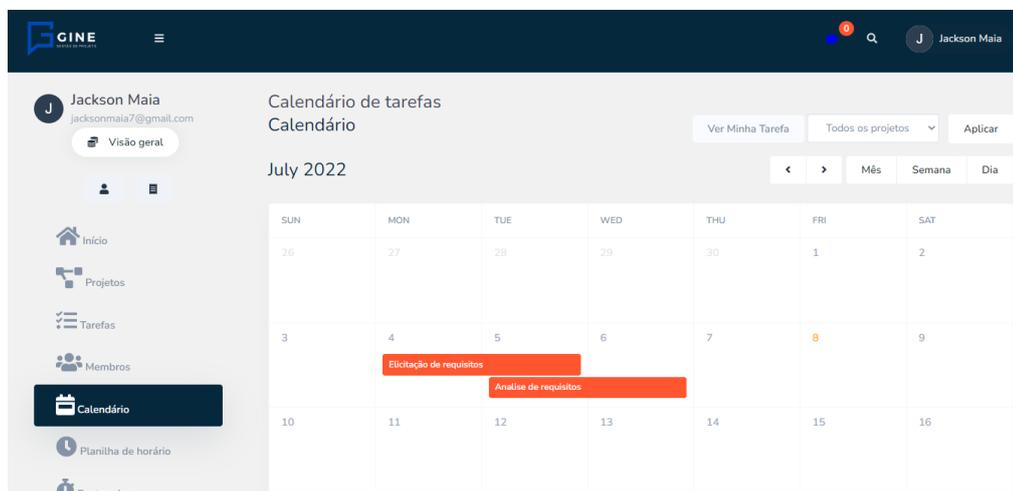
Fonte: Elaborado pelo Autor

No *sprint 5*, foram desenvolvidas as funcionalidades de apresentação do progresso do projeto e apresentação de calendário de atividades. Nessas funcionalidades, apesar do grau de dificuldade, não se teve retrabalho. Alguns recursos de funcionalidade do projeto já tinham sido desenvolvidos em projetos anteriores pelo o desenvolvedor, então foram reutilizados, como é o exemplo da apresentação do calendário.

A carga de tempo necessária para a execução dessa *sprint* foi de 28h, teve duração

entre os dias 04/04/2022 ao 08/04/2022. A validação ocorreu no dia 09/04/2022, na atividade do processo Deployment.

Figura 12 - Calendário de atividades



Fonte: Elaborado pelo Autor

5.2.2.6 Sprint 6

Quadro 10 - Sprint backlog - Sprint 6

ID	Link para planta de especificação	Data de inserção	Tempo de construção (em dias)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#016	Rastreamento de tempo do projeto	11/04/2022	16	16	16/04/2022	não
#017	Apresentar relatórios de projetos	11/04/2022	8	16	16/04/2022	não
#018	Check-in	11/04/2022	8	10	16/04/2022	sim
#019	Check-out	11/04/2022	8	10	16/04/2022	sim

Fonte: Elaborado pelo Autor

No *sprint* 6 foram desenvolvidas as funcionalidades de Rastreamento de tempo do projeto, apresentar relatórios de projetos, check-in, check-out

A carga de tempo necessária para a execução dessa *sprint* foi de 52h, passando do planejamento, devido ao grau de complexidade. A principal dificuldade foi estabelecer um cronômetro para cada tarefa.

A validação ocorreu no dia 18/04/2022, na atividade do processo Deployment e 2 tarefas não passaram pela validação.

5.2.2.7 *Sprint* 7

Quadro 11 - Sprint backlog - Sprint 7

ID	Link para planta de especificação	Data de inserção	Tempo de construção (em dias)		Data de validação	Retrabalho (sim ou não)
			orçado	realizado		
#019	Chat	30/05/2022	24	24	05/06/2022	sim
	Teste de usabilidade e correção de bugs	30/05/2022	16	16	05/06/2022	Não

Fonte: Elaborado pelo Autor

Essa *sprint* foi iniciada depois de um intervalo de tempo da *sprint* 6. Na *sprint* 7 foi executado a atividade de construção do chat de *id* #019. O chat foi desenvolvido utilizando reuso de código de outros projetos, além disso foi realizada uma integração com serviço *pusher* que é um serviço que facilita a comunicação em tempo real.

Por essa *sprint* ser a última a ser executada, foi alocada como última atividade de teste de usabilidade e correção de bugs de integração de módulos.

A carga de tempo necessária para a execução dessa *sprint* foi de 40h. A validação ocorreu no dia 05/06/2022, na atividade do processo *Deployment*.

5.2.3 Deployment

Nesta etapa, foi apresentado o produto para o cliente para que ele fizesse a validação das funcionalidades da ferramenta, assim, gerando a ata de validação e a versão final do produto, porém, como o projeto é um *software* de prateleira e o cliente era o desenvolvedor então foi feita uma adaptação no processo e um dia após o fim da *sprint* acontecia o processo de validação que se dava através de testes e simulações nas funcionalidades desenvolvidas na *Sprint*. Foram simulados no decorrer do desenvolvimento de projetos reais para que se pudesse ver a aplicação da funcionalidade, na prática.

As validações das funcionalidades da *Sprint*, tinha como resultado a ATA de validação, apresentada na Quadro 11, onde o desenvolvedor confirmava a validação de cada funcionalidade. A planta de desenvolvimento servia como base para a validação de cada funcionalidade da *sprint*.

Quadro 12 - Ata de validação Sprint backlog - Sprint 6

ID	Item da sprint backlog	Validado:
#016	Rastreamento de tempo do projeto	Sim
#017	Apresentar relatórios de projetos	Sim
#018	Check-in	Não
#019	Check-out	Não

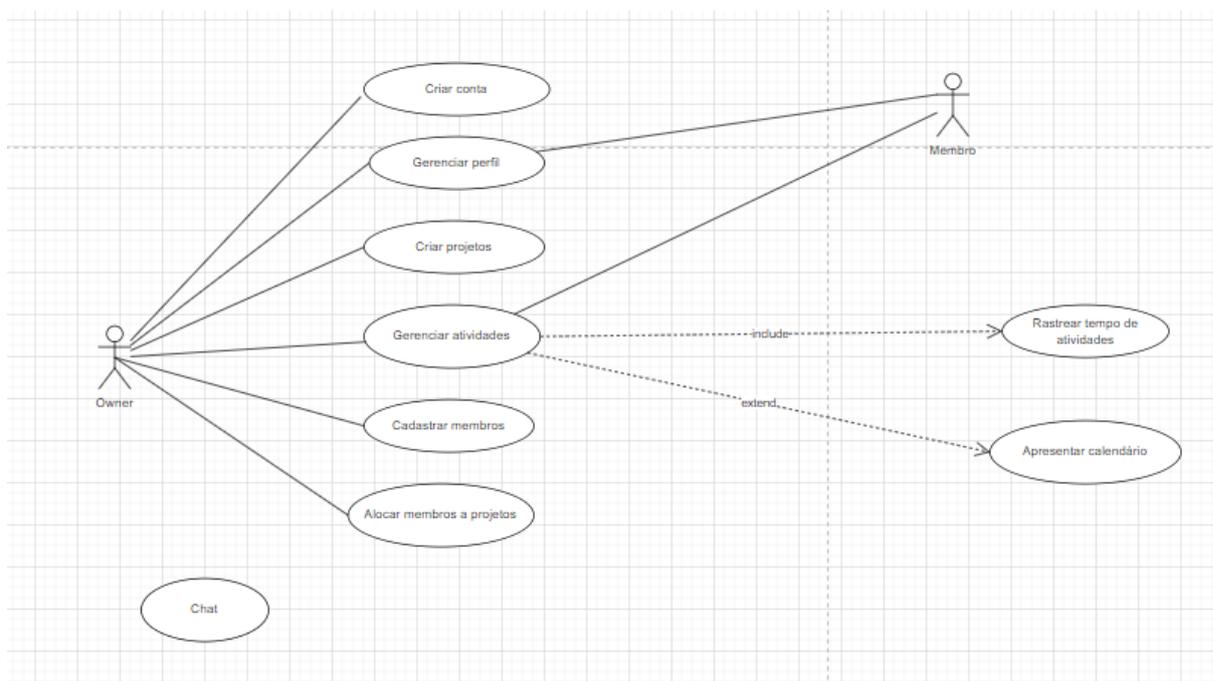
Fonte: Elaborado pelo autor.

Foi selecionado na Ata de validação as funcionalidades da *Sprint* 6 por ser a única com funcionalidades que não foram validadas, que são, *Check-in*, *Check-out* que fizeram parte da *sprint* 6 Essas funcionalidades não foram validadas por que não se conseguiu chegar a uma dinâmica comum que funcionasse de forma geral para equipes que atuassem

em fuso-horários e em horários diferentes e para que cada membro desse do time pudesse individualmente ter sua dinâmica de entrada e saída.

De forma geral, a taxa de funcionalidades válidas é bem alta, isso aconteceu pelo fato do desenvolvimento do projeto ser embasado nos requisitos definidos e na documentação estabelecida.

Figura 13 - Diagrama de fluxo



Fonte: Elaborado pelo autor.

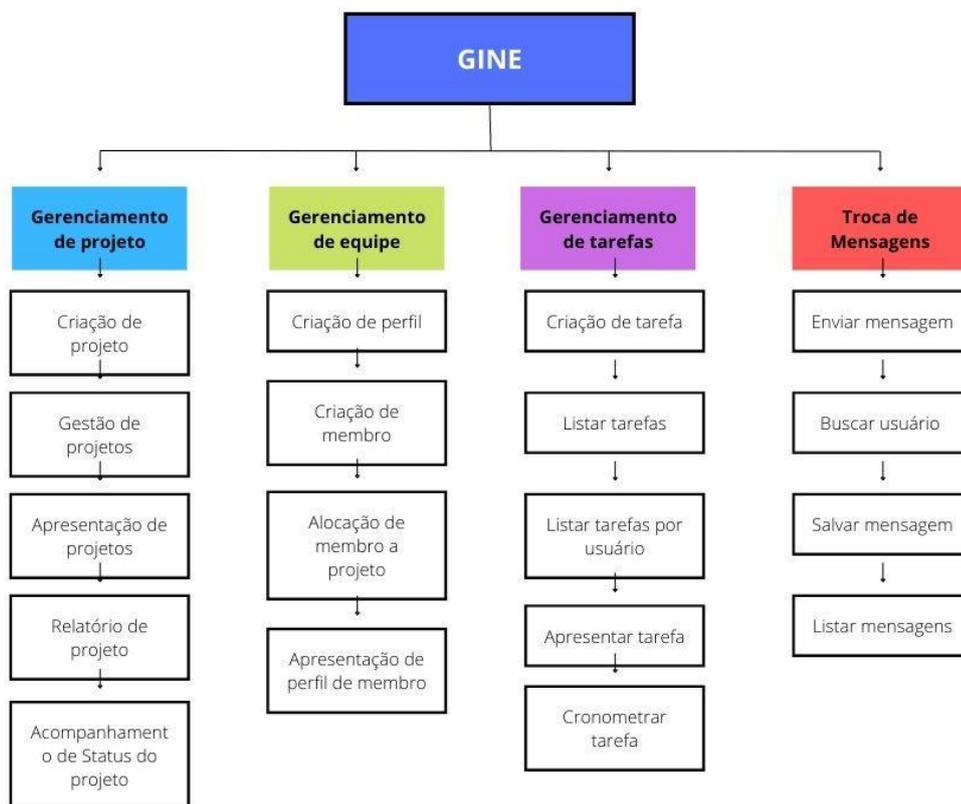
5.2.4 Management

Nessa fase do processo, teve como objetivo planejar, monitorar e controlar o desenvolvimento do produto. Essa fase durou do início até o final do desenvolvimento do software desse projeto.

A Estrutura Analítica do Projeto (EAP) foi gerada nessa fase do desenvolvimento do processo, esse artefato funciona como um mapa de entendimento do escopo do projeto

a ser desenvolvido. A EAP do projeto Gine foi está apresentada na Figura 10 e demonstra a organização das atividades do projeto.

Figura 14 - Estrutura analítica do projeto



Fonte: Elaborado pelo autor.

A estrutura estabelecida apresenta em seu primeiro nível o nome o título do projeto Gine, um pouco abaixo você pode ver as funcionalidades gerais: gerenciamento de projeto, gerenciamento de equipe, gerenciamento de tarefas, troca de mensagens e abaixo os pacotes de trabalho, a estrutura desse organograma é apresentada de forma vertical apresentando assim suas atividades e seus pacotes de trabalho.

Quadro 13 - Cronograma do projeto

Pacote de trabalho					Tempo para construção (Por Sprint)				Responsável
	1	2	3	4	5	6	7		
Cadastro de responsável	■							Jackson M	
Gerenciamento de perfil	■							Jackson M	
Login	■							Jackson M	
Criação de projeto		■						Jackson M	
Gerenciamento de projeto		■						Jackson M	
Lista projetos		■						Jackson M	
Visualização dos projetos		■						Jackson M	
Criação de membro			■					Jackson M	
Alocação de membros a projetos			■					Jackson M	
Criação de tarefas				■				Jackson M	
Gerenciamento de tarefas				■				Jackson M	
Listagem de tarefas				■				Jackson M	
Listagem de tarefas por usuários				■				Jackson M	
Apresentação do progresso do projeto					■			Jackson M	
Apresentação de calendário de atividades					■			Jackson M	
Rastreamento de tempo do projeto						■		Jackson M	
Apresentar relatórios de projetos						■		Jackson M	
Chat							■	Jackson M	

Fonte: Elaborado pelo autor.

Todo o processo teve suas atividades rastreadas e mapeadas no cronograma que apresenta o fluxo que o desenvolvimento do projeto seguiu dividido em *sprints* semanais, o *Scrum Solo* sugere ao desenvolvedor que ele utilize um diagrama de *Gantt* para representá-lo como mostrado na figura acima.

O processo também estabelece a criação de uma planilha de custo, que não se fez necessária no desenvolvimento da ferramenta Gine, já que todo o processo de desenvolvimento aconteceu remotamente, e não existiu nem um investimento financeiro por parte do desenvolvedor, a planilha também não se fez necessária considerando que o *software* é disponibilizado de forma gratuita.

5.3 Avaliação da ferramenta

5.3.1 Montagem da Amostragem

Esta etapa buscou analisar de forma qualitativa a capacidade de aplicação funcional e a usabilidade do produto gerado nesse projeto, além de *feedback* de melhorias da plataforma. Essa análise foi realizada tendo como requisito pessoas atuantes no desenvolvimento de software e que atuassem de forma remota.

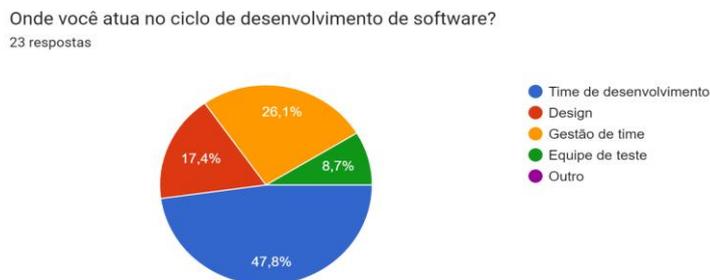
Para que a coleta da amostragem seguisse os critérios estabelecidos, foi desenvolvido um questionário utilizando o Google *Forms* que foi compartilhado em fóruns de tecnologia e em grupos de empresas de desenvolvimento com o intuito de obter uma amostragem de pessoas que atuassem com desenvolvimento de *software* de forma remota ou híbrida que é o público alvo. Esse questionário ficou disponível pelo período dos dias 20 a 30 de junho e participaram um total de 31 pessoas.

Figura 15 - Gráfico de Percentual de pessoas que atuam com desenvolvimento de *software*

Fonte: Elaborado pelo autor

Foi realizado ainda um questionário sobre atuação em time de desenvolvimento para entender melhor quais processos são mais utilizados de acordo com o objetivo de cada um.

Figura 16 - Gráfico de Percentual Área de desenvolvimento



Fonte: Elaborado pelo autor

Figura 17 - Gráfico de Forma de atuação



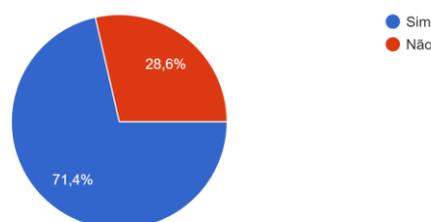
Fonte: Elaborado pelo Autor

Dessa forma, foi possível traçar o perfil dos usuários e pode-se notar uma amostragem bem segmentada no que se relaciona a atuação no ciclo de desenvolvimento de *software*.

Das 23 respostas coletadas, 56,5% atuavam de forma remota, 21,7% atuavam de forma híbrida e 21,7% de forma presencial. Esse terceiro grupo não cumpria o critério de atuação a distância, então estavam habilitadas a passar para a próxima fase apenas 78,2% das pessoas que responderam o questionário.

Figura 18 - Gráfico de Disponibilidade dos participantes para responder a 2ª fase

Você aceitaria passar por uma entrevista sobre avaliação de uma ferramenta direcionada a times de desenvolvimento de software que atuam de forma remota?
21 respostas



Fonte: Elaborado pelo autor

Do total de pessoas habilitadas a participar da segunda fase da entrevista, apenas 71,4% demonstrou disponibilidade de participar da análise qualitativa da ferramenta Gine. As pessoas que se propuseram a contribuir com a análise, deixaram seus e-mails para contato e envio da segunda etapa.

5.3.2 Avaliação Qualitativa

Essa etapa buscava junto a amostra de pessoas selecionadas na etapa anterior realizar uma análise qualitativa da ferramenta que tinha como base a usabilidade do software e as funcionalidades que o mesmo se propunha a oferecer, funcionalidades essas que intencionavam suprir as necessidades de equipe que atuam com desenvolvimento de software distribuído.

A avaliação foi realizada a distância, devido a diferença geográfica dos membros envolvidos e do momento pandêmico em que se encontravam.

Ao início da avaliação foram enviadas aos profissionais do desenvolvimento de software que se dispuseram a participar da entrevista, um e-mail contendo uma mensagem e o link para o site da aplicação.

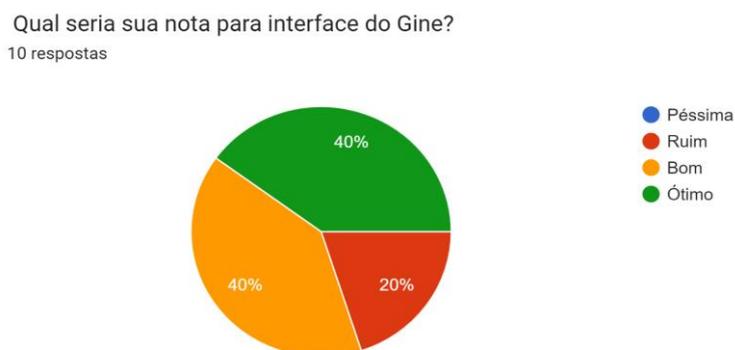
Figura 19 - Avaliação da usabilidade da ferramenta



Fonte: Elaborado pelo autor

Foi perguntado aos entrevistados sobre a qualidade da usabilidade da ferramenta, 90% das pessoas entrevistadas a definiram como boa ou ótima e apenas 10% acharam a usabilidade da ferramenta ruim e nenhum dos entrevistados a definiu com péssima.

Figura 20 - Avaliação para interface



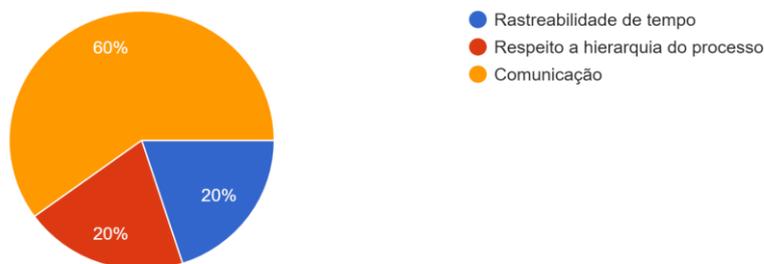
Fonte: Elaborado pelo autor

Em relação à estrutura do layout da plataforma, que define como a plataforma se apresenta visualmente, foi perguntado qual seria a nota para a interface da aplicação. De todos os participantes 80% deles definiu como ótimo e bom, 20% acreditam que a interface da plataforma é ruim e nenhum dos participantes achou que a interface se definiria com péssima.

Figura 21 - O que os usuários acreditam ser o maior suporte que a plataforma Gine

Dentre as principais necessidade do desenvolvimento de software distribuído que é: comunicação, rastreamento de tempo e respeito a... do DDS você acha que o Gine dá MAIOR suporte?

10 respostas



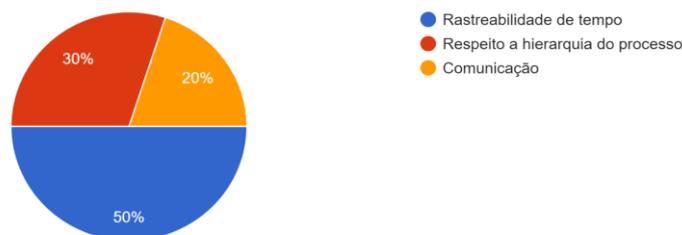
Fonte: Elaborado pelo autor

Com a intenção de estabelecer os pontos forte da ferramenta Gine, foi questionado ainda qual era o maior retorno em relação ao suporte dos recursos que são oferecidos para o usuário, onde se destacou como principal característica da plataforma com 60% de votos a comunicação entre o time, empatam com 20% cada a rastreabilidade de tempo das atividades e o respeito a hierarquia do processo utilizado pela equipe.

Figura 22 - O que os usuários acreditam ser o menor suporte que a plataforma Gine oferece

Dentre as principais necessidade do desenvolvimento de software distribuído que é: comunicação, rastreamento de tempo e respeito ... do DDS você acha que o Gine dá MENOR suporte?

10 respostas



Fonte: Elaborado pelo autor

Com a intenção de estabelecer os pontos fracos e a melhorar na ferramenta Gine, foi questionado ainda qual eram os recursos de menor suporte dos recursos que são oferecidos para o usuário, onde se destacou negativamente a rastreabilidade de tempo

com 50%, em seguida vem o respeito à hierarquia com 30% e depois a comunicação com 20%.

6 CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma solução para as problemáticas definidas referente às metodologias de desenvolvimento distribuído de *software*, onde as empresas de desenvolvimento buscam por mão de obra qualificada em lugares geograficamente distintos. Essa proposta de solução se apresenta por meio de uma ferramenta web.

Esse projeto teve como principal propósito apresentar o desenvolvimento de uma plataforma que dá suporte técnico às equipes de desenvolvedores de *software*. Para o desenvolvimento dessa ferramenta, foi utilizado um processo que se encaixasse na realidade de um time de um único desenvolvedor, então o *Scrum Solo* se mostrou a melhor alternativa por ser dinâmico, fácil de trabalhar e com fases menos burocráticas como se propõe a ser todo bom processo ágil. O *Scrum Solo* se encaixou muito bem com a proposta da ferramenta a ser desenvolvida, mas é importante destacar pontos negativos no processo que poderia dar mais fluidez e celeridade ao desenvolvimento. Primeiramente o processo peca quanto ao rastreamento de atividades durante as suas *sprints* semanais já que não existe nenhum tipo de padronização para lidar com as atividades. Um outro ponto negativo foi a falta de informações em relação aos *templates* propostos, não há material que sirva de exemplo.

Na intenção de solucionar o primeiro problema identificado no *Scrum Solo*, foi estabelecido um fluxo padrão de tratamento das atividades durante cada ciclo de desenvolvimento. As atividades eram executadas dentro de cada *sprint* de acordo com o grau de dificuldade identificado, no período de codificação de cada funcionalidade eram aplicados testes unitários. Ao final de cada *sprint* eram feitas correções de bugs. A transição de uma atividade para outra só acontecia quando a atividade atual fosse finalizada, assim facilitando o tratamento das atividades de cada *sprint*.

A segunda problemática foi solucionada adaptando os *templates* propostos a real necessidade deste projeto, então alguns tópicos foram adicionados ao *template* para que se conseguisse ter uma melhor fluidez no desenvolvimento e que ajudasse na organização

do produto final, como foi o exemplo da descrição de cada item na *Sprint Backlog* e a criação de uma só ATA de validação para todo o projeto

Durante o surgimento da proposta do projeto e do seu desenvolvimento, o mundo passou por um grande período pandêmico que fez com que a grande maioria dos profissionais fosse obrigada a atuar de casa, realidade que intensificou ainda mais a necessidade de ferramentas como Gine que propunham a gestão de projetos a distância, centralizando informações, comunicação, gestão de time, de equipe e de projeto em um só lugar já que podem ser adequadas facilmente a realidade do fluxo de trabalho de qualquer equipe.

Como trabalho futuro, foram recebidas diversas sugestões de como o software Gine poderia melhorar para que pudesse dar um maior suporte a time de desenvolvimento de todos os tamanhos. Como principais pontos a melhorar, existe a necessidade de melhoria na proposta da interface do projeto e a integração com ferramenta de versionamento de código Git e ferramentas de reuniões online como *zoom*.

Uma solução proposta para a melhoria na interface do projeto seria uma parceria com um design, que traria uma nova proposta de apresentação visual, além de acrescentar muito na usabilidade da ferramenta. Isso seria possibilitado pelo fato de o *Scrum Solo* também dar suporte a um time de 2 membros.

Sobre a proposta de integração com ferramentas de versionamento de código e ferramentas de reuniões *online*, será necessária uma nova análise na estrutura do *software* e um novo ciclo de análise de requisitos que tornaria essa ferramenta bem mais robusta com esses recursos e que traria maior suporte a equipes que atuam de forma remota.

Em relação às limitações deste trabalho, provavelmente a principal limitação está relacionada ao número de respostas obtidas. Desse modo, além das sugestões, o projeto necessita de aplicação de testes mais rigorosos que possam trazer *feedbacks* reais, provavelmente, um teste de caixa-preta e um teste de usabilidade formado por uma amostragem de pessoas bem maior traria um *feedback* mais claro.

REFERÊNCIAS

- BECK, K et al. **Manifesto for Agile Software Development**. [S.l], 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso em: 25 jul. 2022.
- BRUNHERA, Diego; ZANATTA, A. **Scrum**: Uma aplicação em um software house. [S.l: s.n], 2010.
- CARMEL, E. **Global Software Teams**: Collaborating Across Borders and Time Zones. [S.l]: Prentice Hall, 1998.
- CARMEL, E. **Global Software Teams – Collaborating Across Borders and TimeZones**. [S.l]: Prentice Hall. 269 p., 1999.
- CARMEL, E.; Tija, P. **Offshoring Information Technology**: Sourcing and Outsourcing to a Global Workforce. [S.l]: Cambridge, 2005.
- 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, Logística e operações internacionais (SIMPOI), 12., 2009, São Paulo, SP. **Anais ...** São Paulo, SP, 2009.
- CAVALCANTI, E.; MACIEL, T. M. M.; ALBUQUERQUE, J. Ferramenta OpenSource para Apoio ao Uso do Scrum por Equipes Distribuídas. In: **Simpósio Brasileiro de Engenharia de Software**. III Workshop de Desenvolvimento Distribuído de Software. Fortaleza, 2009.
- DWIVEDI, Yogesh K. et al. Research on information systems failures and successes: Status update and future directions. **Information Systems Frontiers**. [S.l], v. 17, n. 1, p. 143-157, 2015.
- HERBSLEB, James D.; MOITRA, Deependra. Global software development. **IEEE software**. [S.l], v. 18, n. 2, p. 16-20, 2001.
- HUZITA E. H. M. et. Al. Um ambiente de desenvolvimento distribuído de software – DiSE. In: **I Workshop de Desenvolvimento Distribuído de Software**, João Pessoa, 2007.
- HUZITA, Elisa Hatsue Moriya et al. Um conjunto de soluções para apoiar o desenvolvimento distribuído de software. In: **Simpósio Brasileiro de Engenharia de Software**. II Workshop de Desenvolvimento Distribuído de Software. [S.l: s.n], 2008.
- PAGOTTO, Tiago et al. Scrum solo: software process for individual development. In: **Information Systems and Technologies (CISTI)**. 11th Iberian Conference on. IEEE, [S.l], 2016. p. 1-6.
- PASCUTTI, M.C.D. **Uma proposta de arquitetura de um ambiente de desenvolvimento de software distribuído baseado em agentes**. Dissertação (Mestrado). Universidade Federal do Rio Grande do Sul – Instituto de Informática.

Porto Alegre, 2002.

PRESSMAN, R. S. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2006.

PRIKLADNICKI, R.; AUDY, J. MuNDDoS: Um Modelo de Referência para Desenvolvimento Distribuído de Software. In: **18 SBES - Simpósio Brasileiro de Engenharia de Software**. Brasília, 2004.

RADAVELLI, Alex. **Uma ferramenta para gerenciamento Scrum em ambientes de desenvolvimento geograficamente distribuídos**. [S.l: s.n], 2014.

STEINMACHER, I. F.; Wiese, I.S.; Pozza, R.S.; Huzita, E.H.M. Amorim, E.F; Pascutti, M.C. D. Uma proposta de arquitetura para ambientes de desenvolvimento distribuído de software. In: **CACIC 2005**. Workshop de Ingeniería de Software y Base de Datos (WISBD). Concórdia, 2005.

VERSIONONE. **The State of Agile Development Survey Results**. [S.l: s.n], 2008. Disponível em: http://www.versionone.com/pdf/3rdAnnualStateOfAgile_FullDataReport.pdf. Acesso em: 25 jul. 2022.