



Universidade Estadual de Campinas



Trabalho de Conclusão de Curso

Gestão de Projetos de Desenvolvimento de Software em Grupos de Pesquisa

ALUNO: Thomaz Fernando Ribeiro de Faria

ORIENTADOR: Prof. Dr. Plínio Roberto Souza Vilela

Limeira, SP, Brasil
2019

AGRADECIMENTOS

Agradeço primeiramente a Deus por me dar forças e saúde em todos os momentos, a minha família por sempre ter acreditado, em especial aos meus pais que independentemente da situação me apoiaram e incentivaram a nunca desistir. Ao meu orientador Prof. Dr. Plínio Roberto Souza Vilela que me orientou com excelência sempre que precisei. A todos da Faculdade de Tecnologia da Unicamp que direta ou indiretamente estiveram presentes nesta jornada, sendo todos essenciais. Aos meus amigos que sempre estiveram ao meu lado, me apoiando sempre que precisei durante todos esses anos. E a todos que participaram da minha vida, até mesmo os que hoje não fazem mais parte dela.

RESUMO

Este trabalho foi conduzido na UNICAMP (Universidade Estadual de Campinas), com o apoio de colaboradores do GEICon (Grupo de Engenharia da Informação e Conhecimento), um grupo de pesquisa formado por professores, mestrandos, doutorandos e graduandos. Foi proposto um processo de gestão dos desenvolvimentos de software para utilização em grupos de pesquisa acadêmica. A partir de um levantamento do atual cenário de desenvolvimento de software oriundos de pesquisas acadêmicas, foram constatadas necessidades e oportunidades de melhoria. Entre as carências encontradas estão a falta de sinergia dos alunos com os demais trabalhos que estão sendo executados ou já foram concluídos anteriormente por membros do grupo de pesquisa acadêmica, dificuldade de evolução em softwares desenvolvidos em pesquisas passadas e a dificuldade encontrada por parte dos orientadores em gerir e acompanhar os diversos projetos de software que são desenvolvidos em paralelo. Além da definição do processo de gerenciamento dos projetos de desenvolvimento de software, foram sugeridas ferramentas de apoio ao processo, sendo uma de gestão dos projetos de software, um repositório de documentos e uma ferramenta de repositório de código.

LISTA DE FIGURAS

Figura 1 - Exemplo de projeto com o modelo kanban utilizando o ASANA.	16
Figura 2 - Exemplo de atividade sendo criada no ASANA.	17
Figura 3 - Exemplo de projeto com o modelo kanban utilizando o TRELLO.	18
Figura 4 – Exemplo de atividade criada no TRELLO.	18
Figura 5 – Exemplo de quadro kanban dentro do ARTIA	19
Figura 6 – Exemplo de atividade criada no ARTIA	20
Figura 7 – Opções disponíveis em documento armazenado no GOOGLE DRIVE.	22
Figura 8 – Opções disponíveis ao clicar em detalhes.	23
Figura 9 – Opções disponíveis ao clicar em atividades.	23
Figura 10 – Opções disponíveis ao clicar gerenciar versões.	24
Figura 11 – Opções disponíveis em documento armazenado no ONE DRIVE.	25
Figura 12 – Opções disponíveis ao clicar em histórico de versão no ONE DRIVE.	25
Figura 13 – Tela do repositório do BITBUCKET	27
Figura 14 – Tela do repositório do GITHUB	28
Figura 15 – Fluxo macro de desenvolvimento de Software em grupos de pesquisa.	31
Figura 16 – Fluxo detalhado de desenvolvimento de Software para grupos de pesquisa universitários.	31
Figura 17 - Quadro exemplo a ser utilizado dentro da ferramenta de gestão de projetos ASANA.	36
Figura 18 – Tarefa de exemplo a ser utilizado dentro da ferramenta de gestão de projetos ASANA.	38
Figura 19 – Visualização por lista, com as opções de personalização.	39
Figura 20 – Visualização das tarefas no modo calendário.	39
Figura 21 – Funcionalidade “Progresso” para o registro do andamento do projeto como um todo.	40

- Figura 22** – Funcionalidade “Conversas”, meio disponível para discussões gerais do projeto. 40
- Figura 23**– Funcionalidade “arquivos”, centralizador dos documentos anexados nas atividades. 41

SUMÁRIO

RESUMO	3
1 INTRODUÇÃO.....	7
1.1 CONTEXTO.....	8
1.2 OBJETIVOS.....	9
1.3 METODOLOGIA	10
1.4 ORGANIZAÇÃO DO TRABALHO	11
2 MATERIAIS E MÉTODOS	13
3 PROCESSO PROPOSTO	29
3.1 INSTRUMENTALIZAÇÃO DO PROCESSO	34
4 CONCLUSÃO	43
5 REFERÊNCIAS	44

1 INTRODUÇÃO

O constante crescimento apresentado pelo setor tecnológico nos últimos anos vem se tornando pervasivo tanto no dia a dia das pessoas quanto para as empresas que buscam soluções inovadoras. Segundo números da ABES (Associação brasileira das empresas de software), o mercado de TI (Tecnologia da Informação) nacional movimentou 39,5 bilhões de dólares em 2017 e o segmento de *software* teve um crescimento de 2,8% sobre 2016. Em 2018 o mercado manteve o crescimento movimentando cerca de 47 bilhões de dólares, dados que ressaltam a importância do setor para o país ¹.

O termo *software* engloba desde aplicativos para smartphones, sistemas executáveis em computadores até aplicações *web*. Segundo a definição de Pressman (2011)², *softwares* são instruções (programas de computador) escritas com bases lógicas, que quando executadas fornecem características, funções e desempenho de acordo o código escrito. Os programas são executados de forma que seja possível realizar a interação entre humano e computador ².

No processo de desenvolvimento de *software* tanto em ambiente acadêmico como em desenvolvimentos de mercado, existem fatores semelhantes utilizados para medir a satisfação do cliente com o produto final, entre eles estão o atendimento do escopo definido com o cliente, o projeto ser entregue dentro do cronograma planejado, o software ser receptivo a novas melhorias sem um grande esforço e o gerenciamento de recursos de forma simples.

A fim de auxiliar na melhoria desses fatores nos sistemas desenvolvidos dentro de grupos de pesquisa acadêmicas, foi proposto um processo de gestão de projetos de desenvolvimento de *software*, com base em modelos atuais de mercado e auxiliado por ferramentas.

¹ ABES, **Portal Abes :: Estudo 2019 / Dados 2018**, disponível em: <<http://www.abessoftware.com.br/dados-do-setor/estudo-2019--dados-2018>>, acesso em: 25 nov. 2019.

² Pressman (2011)

1.1 CONTEXTO

Um projeto pode ser considerado como esforço gasto para criar determinado produto ou serviço dentro de um prazo limitado para entrega ¹. O gerenciamento dos recursos pode ser aplicado em diversos cenários e em todo tipo de projeto nas mais diversas áreas, com atividades gerenciadas por um gerente de projetos em desenvolvimentos de mercado ou do orientador em pesquisas acadêmicas.

A Gestão de Projetos é focada em aprimorar e facilitar o acompanhamento do andamento do desenvolvimento, com uma contínua revisão das fases, desde o momento da coleta de requisitos até a entrega final. A fim de definir a proposta do processo para gerenciamento dos projetos de *software* foram utilizadas metodologias que têm como prioridade a qualidade, definindo um conjunto de métodos práticos com um leque de ferramentas que possibilitam profissionais desenvolverem *software* de alto nível ².

Existem diferentes tipos de pesquisas acadêmicas que podem gerar um *software*, entre elas estão o trabalho de conclusão de curso, iniciação científica, iniciação tecnológica, mestrado e doutorado. Esses tipos de pesquisa têm diferentes níveis de dificuldade e também uma duração particular em cada um dos casos. Existem também projetos híbridos, ou seja, projetos integrados que podem envolver mais de uma pesquisa focada no mesmo *software*.

O alto volume de pesquisas em paralelo e a grande rotatividade de alunos são fatores que, quando tratados sem um processo definido e com a falta de ferramentas de apoio, podem prejudicar o gerenciamento por parte do orientador que está envolvido em diversos projetos simultaneamente, dificultando a visibilidade da real situação do desenvolvimento do software, do cronograma a ser realizado e a comunicação com os membros.

¹ KEELING, Ralph; BRANCO, Renato H F, **Gestão de Projetos - Uma abordagem global 3ª ED**, [s.l.]: Saraiva, 2014.

² PRESSMAN, **Engenharia de Software uma abordagem profissional**.

A dificuldade de encontrar a documentação após a entrega do software é outro ponto considerado crítico que foi encontrado no cenário atual do grupo de pesquisas acadêmicas GEICon. Parte do conhecimento gerado dentro da universidade fica com o aluno, que ao finalizar o projeto muitas vezes não terá mais envolvimento acadêmico. Com isso, o incremento de novos módulos ou a implementação de melhorias tem um alto nível de dificuldade. Sendo possível observar algumas pesquisas sendo realizadas com temas semelhantes e que já foram desenvolvidos dentro do grupo de pesquisas, porém com suas particularidades.

A fim de sanar as carências dos grupos de pesquisa no desenvolvimento de software acadêmico e aumentar o nível de satisfação com o produto final gerado, será proposta uma metodologia de gestão de projetos que auxiliará no aumento da qualidade da entrega. Segundo Pressman (2011)¹, para considerarmos um software de sucesso ele precisa ser de fácil utilização, baixa complexidade para realizar modificações e simples implementação de novas melhorias no código, operar sem perder performance por um longo período de tempo e de fato melhorar e/ou facilitar o processo em que foi aplicado.

1.2 OBJETIVOS

Neste trabalho foi definido um processo de gestão onde possam se encaixar as particularidades encontradas nos desenvolvimentos de software oriundos de pesquisas acadêmicas, permitindo ao orientador ter controle sobre diversos desenvolvimentos realizados em paralelo, com uma visão macro do andamento dos projetos e organizando todas as atividades sob sua responsabilidade ou dos demais envolvidos com datas claras de entrega, possibilitando um acompanhamento fidedigno.

O objetivo é facilitar a gestão de projetos de desenvolvimento de software integrados entre diferentes alunos ou grupo de alunos, visto que dentro do cenário acadêmico temos uma

¹ Pressman (2011)

alta rotatividade. O processo deve simplificar a integração de novos alunos, mesmo que ingressem no projeto durante o seu andamento a fim de agregar funcionalidades, além de possibilitar o registro dos históricos do desenvolvimento e a centralização das documentações geradas em um repositório onde o conhecimento gerado nas pesquisas ficará disponível aos membros do grupo. Tornando cada vez mais possíveis projetos de melhorias contínua e incremento de novos módulos em *softwares* já existentes.

Com a possibilidade de projetos incrementais e com interação de diversos alunos pensando em diferentes partes do produto final temos um ganho no valor agregado do produto entregue, permitindo atender a pedidos de evolução do sistema após a entrega do mesmo sem a dependência do aluno que conduziu o primeiro desenvolvimento, os principais objetivos da pesquisa podem ser elencados em:

- Facilitar a gestão dos projetos de software por parte do orientador com o apoio de ferramentas.
- Tornar possível a integração de novos alunos colaboradores, durante o andamento do projeto.
- Aumentar a disponibilidade dos documentos gerados.
- Possibilitar que o software esteja receptível a novos módulos para sua evolução em outras pesquisas.

1.3 METODOLOGIA

Este trabalho foi conduzido com a contribuição de professores e alunos que já fizeram ou fazem parte do grupo de pesquisas GEICon, que de forma livre explicitaram como foram suas experiências com desenvolvimentos dentro do grupo. A fim de identificar o atual cenário e mapear necessidades, foi entendido que atualmente não havia nenhum processo de gestão do

desenvolvimento de *software* definido e utilizado pelos membros do grupo, devido a uma necessidade de adaptação ao modelo de pesquisa acadêmico.

A partir do estudo das metodologias de desenvolvimento de software existentes, procurou-se identificar quais poderiam ser adaptadas de forma a serem utilizadas no ambiente acadêmico e sanar as necessidades mapeadas. Verificou-se que, atualmente, as metodologias ágeis são uma boa opção para desenvolvimento de *software* e que com algumas modificações poderiam se encaixar no processo desejado. Assim sendo, utilizamos de metodologias já validadas e utilizadas mundialmente, como o *Scrum*, para compor a proposta do processo de gestão.

Constatou-se a necessidade de ferramentas para apoio e otimização das atividades, sendo necessário uma ferramenta de auxílio à gestão de projetos, um repositório de documentos e repositório de código fonte. Dentre os tipos identificados como necessárias, foram estudadas opções de mercado que podem suprir as necessidades.

A fim de aprimorar e estruturar a gestão do desenvolvimento no ambiente acadêmico e com base em estudos e análises, foi proposto um novo processo apoiado por ferramentas e que se adapta às necessidades de um grupo de pesquisa. A proposta foi estruturada com base nas pesquisas e revisões bibliográficas. Nesse trabalho não será realizado nenhum teste ou experimento.

1.4 ORGANIZAÇÃO DO TRABALHO

O presente trabalho encontra-se organizado de maneira em que neste capítulo é explicitado ao leitor o contexto em que a pesquisa foi realizada, junto da problemática encontrada nos desenvolvimentos de software realizados nos grupos de pesquisa, os objetivos desejados com a pesquisa e as metodologias utilizadas durante o trabalho.

No capítulo 2 foi realizado o estudo de três tipos de ferramentas necessárias para o processo, sendo elas gestão de projetos, repositório de documentos e repositório de código

fonte. Para cada tipo foram analisadas diferentes ferramentas que podem ser utilizadas com tal finalidade. Além disso, foram analisados os métodos que embasam o processo proposto.

O capítulo 3 é composto pela descrição de todo o processo proposto para a gestão dos desenvolvimentos de software em grupos de pesquisa acadêmica desde a descrição dos papéis de cada integrante da pesquisa, até a definição das fases que compõem o processo. Ressaltando também a documentação mínima obrigatória que deverá ser entregue.

No capítulo 4 foi realizada a análise de como o processo proposto deverá ser executado com as ferramentas sugeridas, realizando uma análise mais completa das ferramentas, referente as funcionalidades presentes e a instrução de preenchimento dos campos que podem ser utilizados no processo.

No capítulo 5 foi realizada a conclusão que se obteve durante o desenvolvimento da presente pesquisa.

2 MATERIAIS E MÉTODOS

Na definição do processo de gestão de projetos proposto neste trabalho foram estudadas diferentes práticas atuais de mercado. Primeiramente foram analisadas as metodologias com os princípios ágeis que ganharam força no ano de 2001, com a publicação do manifesto ágil¹, o qual é um documento onde são mencionados os princípios e fundamentos de um desenvolvimento considerado ágil. Entre os 4 valores da metodologia estão o aprimoramento da relação entre os indivíduos envolvidos no projeto, a preocupação com a entrega em que o software como produto final esteja em funcionamento, a colaboração com o solicitante do software durante o processo de desenvolvimento, e a rápida resposta a mudanças desejadas pelo solicitante com o passar das fases do projeto. O modelo é focado em entregas faseadas, e apresenta facilidade em alterações durante o desenvolvimento.

A grande maioria das metodologias ágeis para o gerenciamento de desenvolvimento de software utiliza-se do quadro *kanban*, modelo proposto inicialmente pela empresa japonesa Toyota em suas linhas de produção. O controle das atividades é de forma visual, onde temos um quadro em que as colunas são as etapas de um projeto e nelas temos os cartões que correspondem as atividades, as quais transitam nas colunas de acordo com o seu andamento facilitando a visibilidade e o controle do projeto. O *kanban* é focado em entregas, é um método que se comporta com eficiência quando necessitamos de mudanças e incrementos já no decorrer da pesquisa².

Além das metodologias ágeis de desenvolvimento de software, é possível encontrar projetos que são gerenciados com base em metodologias denominadas cascata ou linear. Nessas metodologias as fases de desenvolvimentos são inflexíveis, de maneira que ao se concretizar uma fase não há retorno, o que dificulta projetos onde as fases não são bem

¹ BECK, Kent *et al*, **Manifesto para Desenvolvimento Ágil de Software**, disponível em: <<http://agilemanifesto.org/iso/ptbr/manifesto.html>>, acesso em: 10 set. 2019.

² KNIBERG, Henrik; SKARIN, Mattias, **Kanban and Scrum - Making the Most of Both**, [s.l.: s.n.], 2010.

planejadas no início ou projetos nos quais o solicitante deseje alterações durante o seu andamento. Qualquer alteração de escopo solicitada durante o andamento do projeto ou erro encontrado na fase de testes, se torna custoso e gera impacto na entrega do produto final ¹. Segundo Pressman (1995)², os projetos reais dificilmente seguem um fluxo sequencial proposto na metodologia cascata. Um dos motivos é a dificuldade encontrada pelo cliente em declarar todas as exigências no início do desenvolvimento.

Para o nosso processo, foi escolhido o embasamento em metodologias ágeis, devido à necessidade de gerarmos softwares aptos a novos módulos e receptivos a mudanças tanto de escopo quanto de membros da equipe, visto que as pesquisas acadêmicas são divididas em semestres e gerando uma alta rotatividade de integrantes nos grupos de pesquisa.

Dentre as diversas metodologias ágeis podemos destacar o *Scrum*, focado em organizar projetos de desenvolvimento de software em entregas faseadas, possibilitando rápidas respostas a mudança. A metodologia com entregas em pequenos módulos auxiliam o time na absorção de novos integrantes no início de cada ciclo de desenvolvimento, visto que os membros do projeto terão um período para a passagem de conhecimento e explicação do que já foi feito, e em caso de dúvidas o novo integrante poderá recorrer aos alunos com mais tempo de projeto, obtendo uma passagem de conhecimento contínua ³.

Com a definição de um processo de gestão de projetos de desenvolvimento de software no ambiente acadêmico, foi identificadas as necessidades de uma ferramenta de auxílio ao gerenciamento do projeto, possibilitando o controle e acompanhamento das atividades, e a atribuição de responsáveis, definição de datas de término e o acompanhamento sobre o andamento das mesmas. Além de centralizar a comunicação com todos os membros da equipe.

¹ LESSA, Rafael Orivaldo; ORIVALDO LESSA JUNIOR, Edson, **Modelos de Processos de Engenharia de Software**, Santa Catarina: [s.n.], 2006.

² PRESSMAN, Roger S., **Engenharia de Software- (3ª edição)**, São Paulo: [s.n.], 1995.

³ SCHWABER, Ken; SUTHERLAND, Jeff, Guia do Scrum MR, 2017.

A partir da necessidade encontrada foram escolhidas e analisadas três possibilidades de ferramentas de gestão de projetos disponíveis atualmente no mercado. Para se adaptar ao processo proposto, a ferramenta deveria ter obrigatoriamente as funcionalidades a seguir na sua versão gratuita:

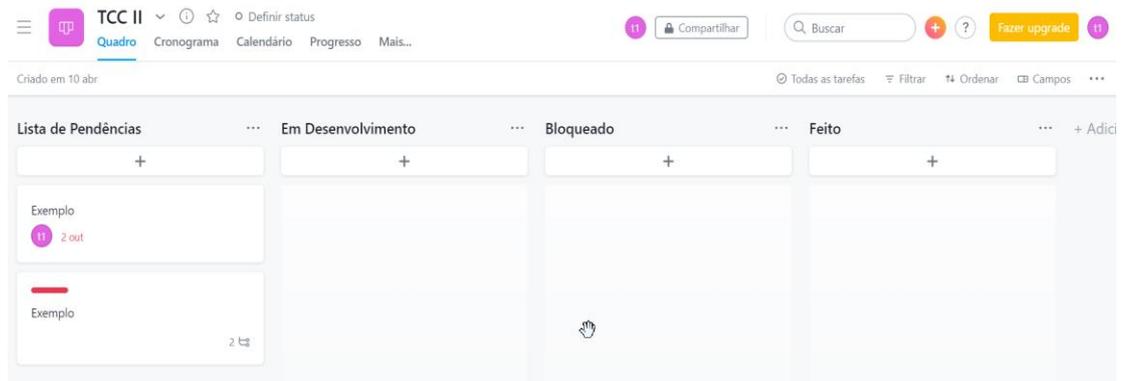
- Possibilidade de utilização do modelo *kanban*.
- Criação e atribuição de atividades a seus responsáveis.
- Criação de inúmeros projetos.
- Definição da data de entrega nas atividades, com aviso automático por e-mail ao responsável.
- Inclusão de comentários e anexos nas atividades.

A primeira ferramenta analisada foi o ASANA¹ que é uma ferramenta *web* e oferece seu aplicativo *mobile*, é de origem norte americana e foi criada em 2008 com foco em auxiliar equipes a organizar e gerenciar seus projetos. A ferramenta é flexível para times que se gerenciam de diferentes formas, entre elas a possibilidade de utilização do método de quadros *kanban* ou o método tradicional de lista, os quais são totalmente personalizáveis para a necessidade do projeto, criação de tarefas e atribuição a membros do time com a data de entrega definida. Dentro das tarefas é possível anexar documentos e realizar comentários.

A Figura 1 apresenta um projeto criado como exemplo no ASANA utilizando o modelo *kanban*, com colunas personalizadas escolhidas para esse projeto em especial.

¹ ASANA, disponível em: <<https://asana.com>>.

Figura 1 - Exemplo de projeto com o modelo kanban utilizando o ASANA.

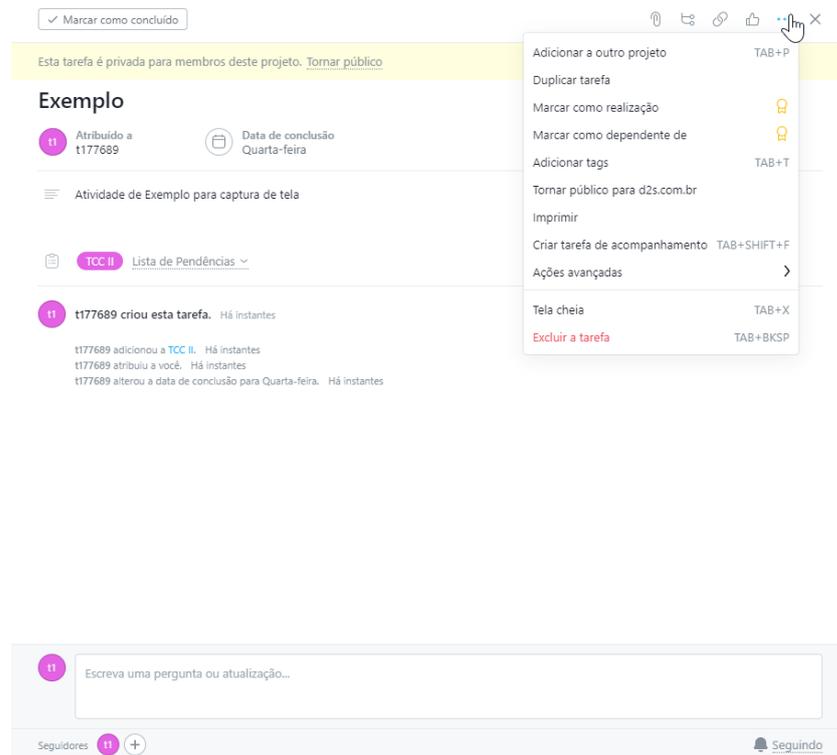


Fonte: ASANA¹

A Figura 2 apresenta a atividade sendo criada no ASANA. Entre os atributos disponíveis podemos observar alguns pontos importantes para a nossa pesquisa, tais como possibilidade de atribuir responsável à tarefa, a inclusão da data de conclusão, possibilidade de incluir comentários e anexos e o registro histórico de modificações na tarefa. E funcionalidades extras que podem ser utilizadas para facilitar a gestão como adicionar *tags* que facilitam a visualização dentro do quadro e a criação de subtarefas, desmembrando a atividade.

¹ *Ibid.*

Figura 2 - Exemplo de atividade sendo criada no ASANA.



Fonte: ASANA¹

A segunda ferramenta analisada foi o TRELLO², ferramenta criada em 2011 e que utiliza do modelo de negócio que disponibiliza uma versão gratuita e algumas funcionalidades extras que, se necessário, devem ser contratadas. Possui tanto a versão *web* quanto a versão *mobile*. O TRELLO foi criado para auxiliar as equipes a se organizarem durante o andamento do projeto, de forma simples e intuitiva.

Dentre os recursos necessários para o processo proposto, o TRELLO oferece a possibilidade de criar quadros com o modelo *kanban*, criação e atribuição de tarefas com data de entrega pré-definida, marcadores que facilitam a visualização dentro do quadro, inclusão de comentários e anexos dentro das atividades. Funcionalidades similares às apresentadas pelo ASANA.

¹ *Ibid.*

² **Trello**, disponível em: <<https://trello.com>>.,

Na Figura 3 é possível observar um projeto criado de exemplo utilizando do modelo quadro *kanban* dentro da ferramenta TRELLO.

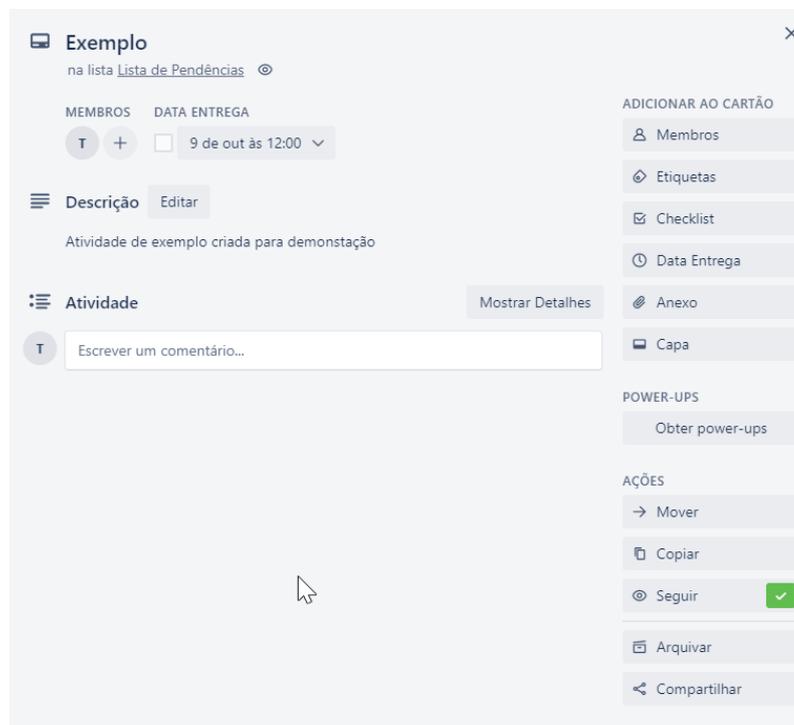
Figura 3 - Exemplo de projeto com o modelo kanban utilizando o TRELLO.



Fonte:Trello ¹

Na Figura 4 temos o exemplo de criação de uma atividade dentro do TRELLO, onde conseguimos observar os atributos disponíveis e essenciais para a cobertura do processo proposto tais como possibilidade de atribuição de responsáveis à tarefa no campo membros, a inclusão da data de conclusão e a possibilidade de incluir anexos e comentários.

Figura 4– Exemplo de atividade criada no TRELLO.



Fonte:Trello ¹

¹ *Ibid.*

Outra ferramenta utilizada para análise foi o ARTIA¹. Assim como as ferramentas anteriores, também foi criado com o intuito de facilitar a gestão de projetos o ARTIA está disponível apenas na versão *web*. Além das funcionalidades básicas de gestão, possui uma variedade de relatórios gerados automaticamente a partir dos indicadores do projeto. Devido a sua variedade de funcionalidades o ARTIA tem uma interface um pouco mais complexa que as duas ferramentas analisadas anteriormente, exigindo que o usuário iniciante assista a alguns tutoriais disponíveis no próprio site da ferramenta.

A análise de compatibilidade com os requisitos necessários na ferramenta de gestão para o processo criado, encontramos as seguintes funcionalidades criar quadros *kanban* e a criação de tarefas com atribuição de responsáveis e data de término, inclusão de comentários e anexos. Uma funcionalidade extra é o controle de tempo real gasto na tarefa, um indicador que é voltado para o controle de recursos dentro do ambiente de desenvolvimento empresarial, onde na maioria dos casos os projetos são custeados com base em horas estimadas.

Na Figura 5 podemos observar uma visualização de quadro *kanban* dentro do ARTIA.

Figura 5– Exemplo de quadro *kanban* dentro do ARTIA



Fonte: <https://artia.com> ¹

Na Figura 6 temos a criação de uma atividade dentro da ferramenta, e podemos observar as funcionalidades listadas como essenciais para o nosso processo, sendo elas possibilidade

¹ **Artia**, disponível em: <<https://artia.com>>.,

de atribuição a responsáveis, definição da data de conclusão, inserção de anexos e comentários.

Figura 6 – Exemplo de atividade criada no ARTIA

Atividade | [Editar](#) | [Pendente](#) | TF | 0 | 1 | 0 x

Título
Exemplo

Projeto / Pasta
Grupo de Trabalho > TCC2

Categorias

Descrição
Tarefa de exemplo

Início Estimado 30/09/2019	Término Estimado 30/10/2019
Esforço Estimado 0,00	% Completo 0,00%

[Fechar](#)

Fonte: <https://artia.com>¹

Uma das principais carências dos grupos de pesquisa é a descentralização dos documentos gerados, visto que temos diferentes pesquisas sendo orientadas por diferentes professores do grupo. Como os documentos anexados nas ferramentas de gestão de projetos analisadas acima ficam disponíveis apenas para os membros do time de desenvolvimento, foi sugerido um repositório de documentos, a fim de aumentar a disponibilidade do conhecimento gerado e unir os documentos das pesquisas em uma pasta compartilhada na qual os orientadores serão responsáveis por conceder permissões de acesso. A seguir será apresentada uma análise de 2 possíveis ferramentas para executar essa função, sendo elas o GOOGLE DRIVE e ONE DRIVE. As ferramentas deverão ter como requisitos mínimos:

- Tecnologia de nuvem, a fim de aumentar a disponibilidade e possibilitar o acesso de qualquer lugar.

¹ **Artia**, disponível em: <<https://artia.com>>.,.

- Capacidade ilimitada de armazenamento.
- Criação de pastas compartilhadas.
- Ilimitados colaboradores nas pastas.
- Versionamento de documentos.

A primeira ferramenta analisada foi o GOOGLE DRIVE¹, uma ferramenta desenvolvida pelo Google em 2012 com a tecnologia de armazenamento em nuvem. A computação em nuvem é um modelo que busca aumentar a disponibilidade dos documentos e facilitar a escalabilidade dos serviços, onde qualquer dispositivo com acesso à internet pode visualizar ou manipular os documentos armazenados. A alusão ao termo nuvem faz referência a facilidade de acesso aos arquivos, podendo ser acessados de qualquer local do mundo. ².

Os requisitos mínimos necessários para utilização no processo sugerido para as pesquisas acadêmicas, estão presentes na ferramenta, sendo eles versionamento automático de documentos carregados, com o histórico de cada modificação, criação de pastas compartilhadas sem limite de colaboradores na pasta. O único requisito para acessar ao GOOGLE DRIVE é ter uma conta Google, fator que favorece a sugestão da ferramenta no processo proposto nessa pesquisa, visto que todo colaborador ao ingressar na Universidade Estadual de Campinas (UNICAMP) automaticamente ganha uma conta Google com um espaço de armazenamento ilimitado, eliminando futuros problemas com capacidade de armazenamento.

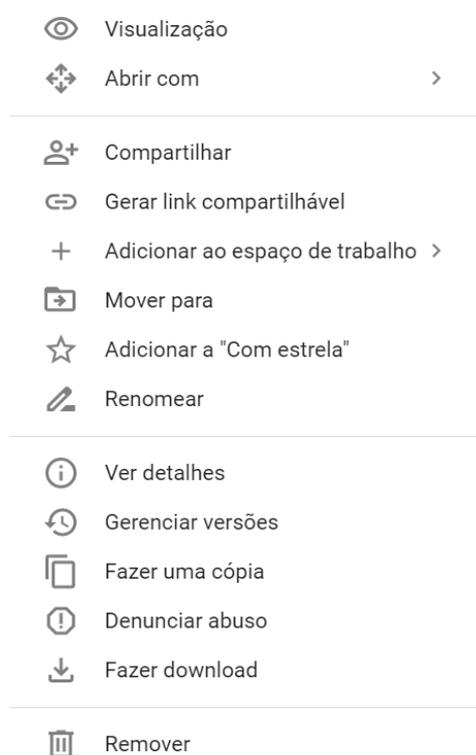
Na Figura 7 é possível observar as opções que temos com um documento já alocado no GOOGLE DRIVE e os requisitos que se fazem necessários para a utilização no processo proposto de gestão. Nas figuras 8 e 9 temos as informações que são mostradas. Ao clicar em detalhes. É possível observar todo o histórico de alterações e informações referente ao

¹ **Google Drive**, disponível em: <<https://drive.google.com/>>.,.

² RUSCHEL, Henrique; ZANOTTO, Mariana Susan; MOTA, Wélton Costa da, *Computação em nuvem.*, p. 15, 2010.

documento desde a sua criação na ferramenta. Na Figura 10 pode-se observar o controle de versões do documento, exibidos ao clicar na opção gerenciar versões. É importante verificar se as versões anteriores estão com a opção manter indefinidamente selecionada, essa opção já vem selecionada por padrão da ferramenta. Caso não esteja selecionada a versão poderá ser excluída em 30 dias ou quando o documento ultrapassar 100 versões.

Figura 7– Opções disponíveis em documento armazenado no GOOGLE DRIVE.



Fonte: <https://drive.google.com>¹

¹ Google Drive.

Figura 8 – Opções disponíveis ao clicar em detalhes.

The screenshot shows the 'Detalhes' (Details) tab for a file named 'Tcc 2.docx'. The file is a Word document located in 'Meu Drive' (My Drive). It was modified on October 10, 2019, by the user 'eu' (me). The file size is 530 KB (542,369 bytes), and it has used 1,008 KB (1,032,695 bytes) of storage. The file was opened on October 10, 2019, by 'eu', and was created on October 2, 2019, using Google Drive Web. There is an option to 'Adicionar uma descrição' (Add a description) and a note that 'Os leitores podem fazer o download' (Readers can download).

Detalhes		Atividade
Tipo	Word	
Tamanho	530 KB (542.369 bytes)	
Armazenamento usado	1.008 KB (1.032.695 bytes)	
Local	Meu Drive	
Proprietário	eu	
Modificado	em 10 de out de 2019 por mim	
Aberto	em 10 de out de 2019 por mim	
Criado em	2 de out de 2019 com o Google Drive Web	
Adicionar uma descrição		
Os leitores podem fazer o download		

Fonte: <https://drive.google.com>¹

Figura 9 – Opções disponíveis ao clicar em atividades.

The screenshot shows the 'Atividade' (Activity) tab for the file 'Tcc 2.docx'. It displays a list of recent activities under the heading 'No início desta semana' (At the beginning of this week). The first activity is from Thursday at 20:58, where the user 'Você' (You) edited 1 item, 'Tcc 2.docx'. The second activity is from Thursday at 20:56, where the user 'Você' renamed 1 item, 'Tcc 2.docx', to 'Tcc-2-V3.docx'. Below this, there is a section for 'Semana passada' (Last week) starting on October 2nd.

Detalhes		Atividade
No início desta semana		
qui 20:58	Você editou 1 item	
qui 20:56	Você renomeou 1 item	
	Tcc-2-V3.docx	
Semana passada		
2 de out		

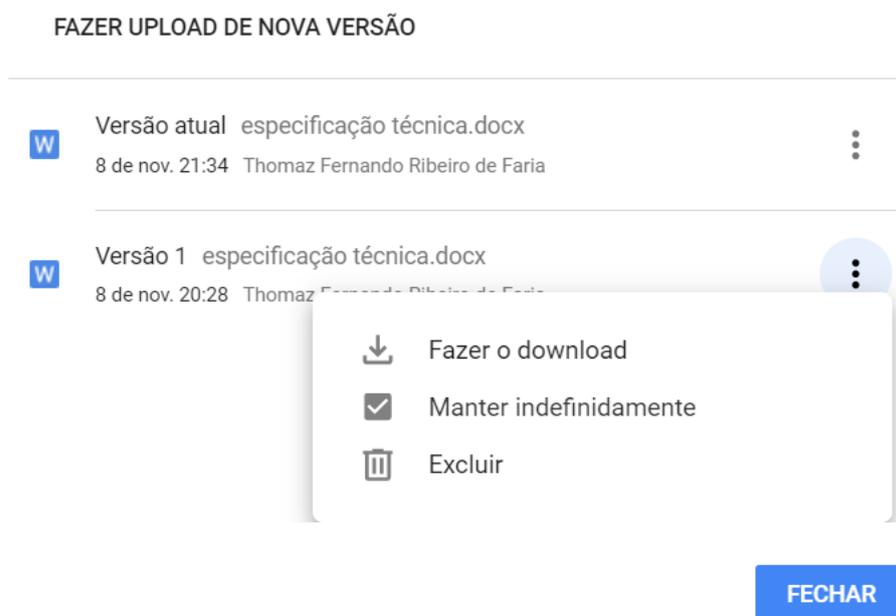
Fonte: <https://drive.google.com>¹

¹ *Ibid.*

Figura 10 – Opções disponíveis ao clicar gerenciar versões.

Gerenciar versões

As versões anteriores de "especificação técnica.docx" podem ser excluídas após 30 dias ou depois que 100 versões forem armazenadas. Para evitar a exclusão, selecione **Manter indefinidamente** no menu de contexto do arquivo. [Saiba mais](#)



Fonte: <https://drive.google.com>¹

A segunda ferramenta analisada foi o Microsoft ONE DRIVE², é uma ferramenta de armazenamento em nuvem possibilitando o acessar os documentos de qualquer dispositivo com acesso à Internet tem sua versão *web* e *mobile*. A versão gratuita do serviço oferece ao usuário 5 GB de armazenamento, um fator importante uma vez que em um repositório compartilhado não podemos ter um limite de dados, pois caso seja ultrapassado teremos futuros problemas com o armazenamento de arquivos.

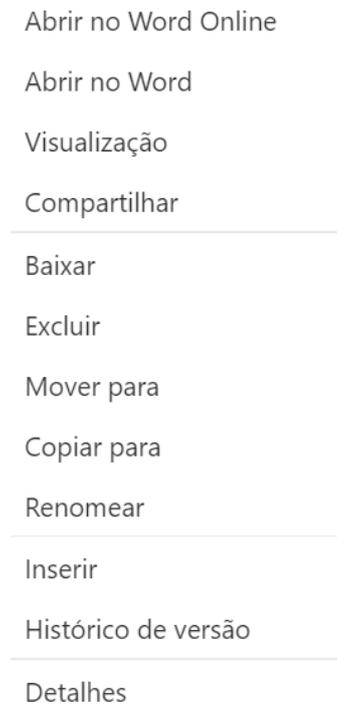
Na Figura 11 é possível observar as funcionalidades do ONE DRIVE em um arquivo que está alocado na ferramenta, A Figura 12 mostra as opções ao clicar no histórico de

¹ *Ibid.*

² **One Drive**, disponível em: <<https://onedrive.com>>..

versionamento com o registro de todas as modificações no documento. É possível criar pastas compartilhadas e integrar com outras ferramentas da Microsoft.

Figura 11 – Opções disponíveis em documento armazenado no ONE DRIVE.



Fonte: <https://onedrive.com>¹

Figura 12 – Opções disponíveis ao clicar em histórico de versão no ONE DRIVE.



Fonte: <https://onedrive.com>¹

¹ *Ibid.*

O processo de gestão foi pensando para pesquisas que tenham desenvolvimento de software, como uma boa prática de desenvolvimento e uma forma de aumentar a vida útil do sistema. Para o armazenamento e versionamento de códigos fonte, precisamos de uma ferramenta especializada que exerça funcionalidades específicas que estão dentro de boas práticas de desenvolvimento. Ambas as ferramentas de repositório de documentos não oferecem os requisitos desejados para a gestão do código. Os requisitos desejados são:

- Possibilidade de compilar e testar módulos que estão em desenvolvimento separadamente.
- Permissões diferentes de acesso ao código
- Processo de aprovação do código antes de ser incrementado a versão atual.
- Controle de versionamento

A primeira análise foi realizada no BITBUCKET¹, uma ferramenta privada que possui um plano acadêmico gratuito, do qual todos os colaboradores da universidade poderão desfrutar. A versão acadêmica possibilita projetos com o número de colaboradores ilimitado. Os projetos são privados e podem ter acesso ao código fonte apenas as pessoas que o responsável concedeu acesso. Além disso, é possível ter níveis de autorização com diferentes ações liberadas em cada perfil.

Outro fator considerado como mínimo para a ferramenta atender às necessidades é o compilador online de código com integração a ferramentas de testes auxiliando a garantir a qualidade do código. Aliado às boas práticas, é possível que toda alteração no código antes de ser efetivada passe pela aprovação do responsável. Uma vez o código alterado todo o histórico de armazenamento fica registrado no controle de versionamento.

¹ **Bitbucket**, disponível em: <<https://bitbucket.org>>.,.

Na Figura 13 é possível observar a interface do BITBUCKET, onde no menu *Source* temos a disponibilização do código fonte armazenado. Na opção *Pipelines* é possível realizar a compilação e teste do trecho código direto na ferramenta que suporta todas as linguagens ou plataformas entre elas *java*, *javascript*, *php*, *python* entre outras. O menu *Deployments* auxilia o desenvolvedor a realizar testes em ambientes apropriados antes que o código seja implementado, garantindo a confiabilidade do projeto. *Downloads* possibilita ao time armazenar e baixar códigos fontes e arquivos. A opção de *Settings* é aonde são realizadas as configurações do repositório, entre elas a configuração de fluxo de aprovação em alterações no código, permissões de acesso a cada usuário entre outras informações.

Figura 13– Tela do repositório do BITBUCKET



Fonte: <https://bitbucket.org>¹

A segunda ferramenta analisada foi o GITHUB², uma plataforma que armazena códigos contendo controle de versionamento, utilizada por milhares de desenvolvedores. Oferece um compilador online para executar testes dos módulos facilitando para o programador e semelhante a ferramenta anterior os seus níveis de acesso podem ser definidos pelo líder do projeto, junto da configuração de aprovação necessária para alterações no código fonte.

A ferramenta ficou conhecida pelo modelo de trabalho de código aberto ou em inglês *open source*, que se trata de um movimento onde seus adeptos alocam seus códigos fontes

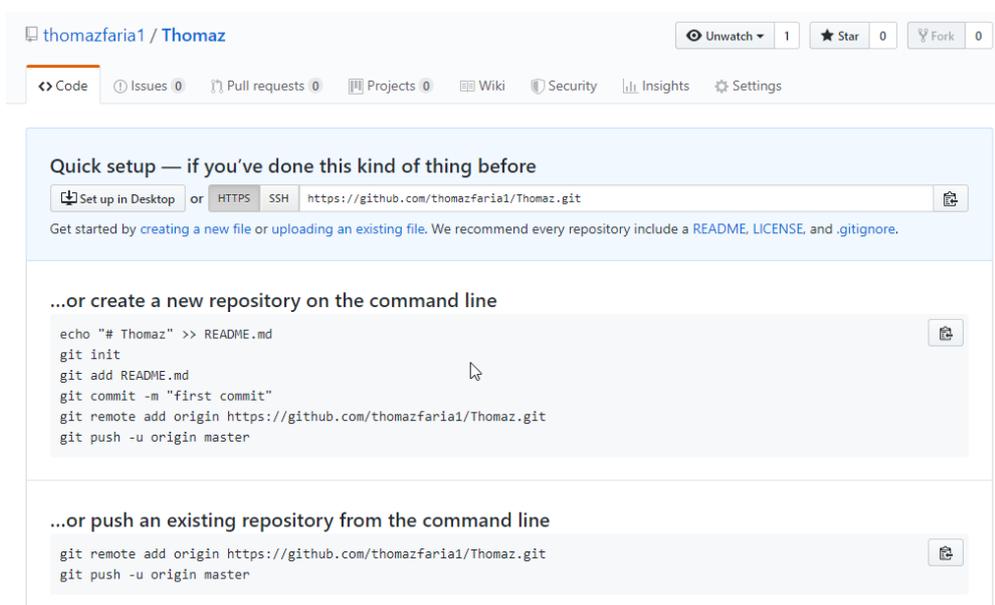
¹ *Ibid.*

² **GitHub**, disponível em: <<https://github.com>>.,.

em ferramentas semelhantes ao GITHUB e qualquer pessoa pode acessar e visualizar. Caso necessite de um projeto privado é possível na versão gratuita, porém temos um limite baixo de colaboradores nos projetos.

Na Figura 14 podemos observar um repositório que foi criado como exemplo, e não foi utilizado até o presente momento. Foi realizada uma análise das principais funcionalidades da ferramenta que aparecem na imagem abaixo. Na aba *Code* é possível visualizar o código fonte que está no repositório, na aba *Issues* a ferramenta permite que qualquer colaborador crie tarefas de correção de erros ou melhorias no software. A aba *Pull requests* permite que sejam feitas sugestões de melhorias no código, essa funcionalidade pode passar por um fluxo de aprovações antes de ser incorporada ao código fonte. Na aba *Settings* é possível a parametrização das permissões de acesso ao quadro e alterações nas funcionalidades disponíveis.

Figura 14– Tela do repositório do GITHUB



Fonte: <https://github.com>¹

¹ *Ibid.*

3 PROCESSO PROPOSTO

Neste trabalho foi definido um processo de gestão de projetos para ser aplicado nas pesquisas que envolvem desenvolvimento de software dentro de grupos acadêmicos, a fim de eliminar problemas constantes, entre eles a forte dependência do desenvolvedor original para implementação de novos módulos, já que se corre o risco do conhecimento gerado durante o desenvolvimento se perder com a saída do aluno do grupo de pesquisa. Em grande parte dos casos, após o aluno estar formado é difícil o contato com ele prejudicando o princípio de melhoria contínua. Além disso, o novo processo visa facilitar a integração de novos colaboradores durante as novas *sprints* do projeto e auxiliar o orientador no gerenciamento.

Processos podem ser definidos como um conjunto de passos a serem seguidos, que auxiliará na previsão de prazos, aumentando o controle sobre o desenvolvimento, tendo como objetivo assegurar a qualidade do produto final. Um processo adotado depende da área de atuação do software, e pode ser adaptável para se encaixar ao cenário atual do local a ser implementado ¹.

O processo foi proposto com base nos pilares de metodologias já testadas e consolidadas, proporcionando ao aluno que ainda está no ambiente acadêmico a oportunidade de ter um contato mais próximo com o cenário real encontrado no mercado de trabalho.

Foi considerado um fluxo macro que poderá ser adequado aos diferentes tipos de pesquisa. No ambiente acadêmico são conduzidas pesquisas que podem variar de 6 meses a 5 anos sendo os seguintes tipos iniciação científica, trabalho de conclusão de curso, iniciação tecnológica, mestrado e doutorado. O processo a ser adotado é ligado a métodos ágeis e com adaptações para que se encaixem as particularidades dos grupos de pesquisa, com um processo flexível podendo ser utilizado tanto em uma pesquisa com duração de 6 meses quanto em uma de 5 anos.

¹ PRESSMAN, *Engenharia de Software uma abordagem profissional*.

Serão descritos os papéis e responsabilidades de cada membro durante o desenvolvimento do projeto. Atualmente a equipe pode conter um ou mais alunos além de professores orientadores envolvidos. É comum encontrar cenários em que existam trabalhos vinculados uns aos outros, aumentando consideravelmente o volume da interação entre pessoas. O processo de gestão de projetos definido deverá diminuir a complexidade do processo atual.

Papéis:

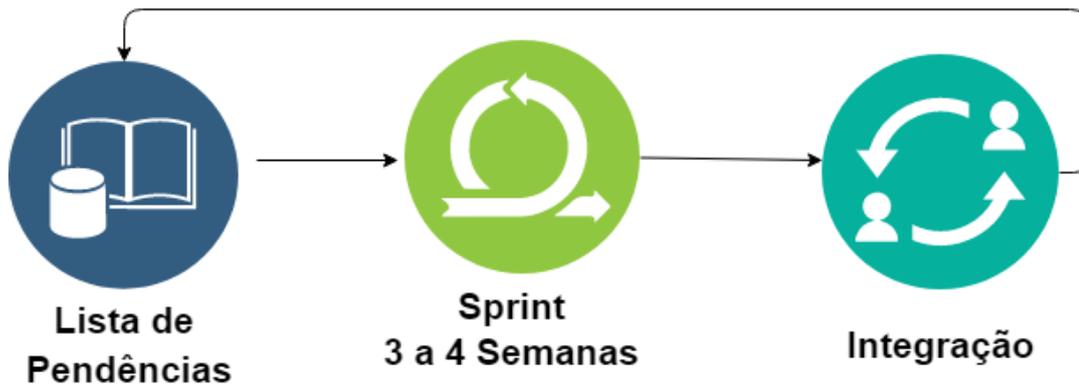
Orientador: responsável por ter a visão do produto que deve ser entregue e dos próximos passos a serem incrementados no sistema, deve fornecer as pendências que farão parte da lista de atividades, colocando prioridade e indicando quem será o responsável por executá-la, deixando claro para os envolvidos no projeto quais são os objetivos de cada *sprint*. Papel que deverá ser exercido pelo Professor (Orientador ou coorientador).

Líder: responsável por ser o líder do projeto, deverá ter a visão do andamento e realizar a integração com novos membros da equipe, mantendo foco no processo e removendo impedimentos. De forma que facilite a comunicação entre os envolvidos. Só deverá ser nomeado um líder em pesquisas que envolverem mais de um aluno. Este papel será exercido pelo integrante com mais conhecimento sobre o produto a ser desenvolvido. Idealmente deverão ser alunos de mestrado ou doutorado os quais permanecerão por mais tempo no grupo e poderão fazer a transferência de conhecimento aos novos membros com qualidade.

Time de desenvolvimento: são os responsáveis por realizar o desenvolvimento do software, devem ser auto gerenciáveis e se preocupar em conseguir cumprir as atividades designadas pelo professor no *Sprint*. Este papel deverá ser desempenhado pelo aluno.

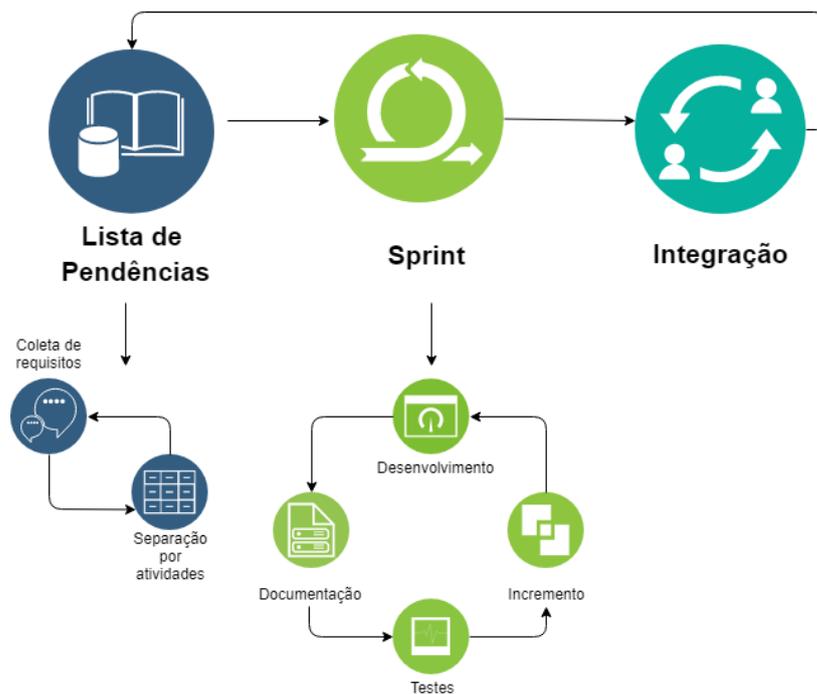
As fases do processo estão representadas na Figura 15 em sua forma macro e na Figura 16 junto das subatividades, e serão definidas a seguir:

Figura 15– Fluxo macro de desenvolvimento de Software em grupos de pesquisa.



Fonte: O próprio autor.

Figura 16 – Fluxo detalhado de desenvolvimento de Software para grupos de pesquisa universitários.



Fonte: O próprio autor.

Fases do processo de desenvolvimento:

- Lista de Pendências: essa é a macro atividade onde o Professor deverá elencar quais são as atividades pendentes para a execução do software, sendo o momento do planejamento da próxima *sprint*. Dentro dessa fase temos mais duas atividades sendo elas:

- Coleta de Requisitos: é uma subatividade da lista de pendências. Devem ser listados os requisitos necessários do sistema, revistos os requisitos desejados no início de cada *sprint*, os módulos que deverão ser desenvolvidos durante todo o projeto é aqui que serão possíveis inserir novos módulos que apareceram durante o andamento do projeto.

- Separação por atividades: é uma subatividade da lista de pendências. Tendo a coleta de requisitos finalizada, o orientador deverá dividir as atividades que sejam possíveis de se concluir no período da *sprint*. O tempo de conclusão de cada atividade deve ser discutido entre todos os integrantes da pesquisa. Com a possibilidade de definir prioridades e pesos para cada atividade.

- *Sprint*: é a macro atividade onde será realizado o desenvolvimento, em intervalos com duração de 3 a 4 semanas. No final do *sprint* deve ser entregue tudo o que foi produzido durante o período, que pode ser dividido nas 4 subatividades apresentadas a seguir

- Desenvolvimento: nesta subatividade o time estará dedicado unicamente a realizar o desenvolvimento das tarefas designadas a cada um e codificação do software.

- Documentação: é uma subatividade do *sprint* e nesse passo o time de desenvolvimento terá por obrigação realizar o versionamento do código na ferramenta escolhida, atualizar toda a documentação que estará no repositório de dados, e incluir comentários na ferramenta de gestão de projetos.

- Testes: é uma subatividade do *sprint* onde após o incremento o time deverá realizar os testes necessários e coletar evidências de que todos os módulos foram incluídos com sucesso.

- Incremento no Software: é uma subatividade do *sprint*, nesse momento o time de desenvolvimento deverá se dedicar exclusivamente em se preocupar com o acoplamento dos novos módulos desenvolvidos.

- Integração: deverá acontecer após o término dos testes e antes do início do planejamento da próxima *sprint*, onde professor e líder deverão dar uma introdução sobre o que se refere o projeto e detalhar quais serão as atividades do novo colaborador. Etapa essencial para a continuidade do software.

Como ponto importante dentro da gestão de projetos e essencial para o aumentar a receptibilidade dos softwares a novos módulos feitos por diferentes pesquisadores, à documentação tem papel fundamental, fornecendo base e conhecimento de como o sistema foi desenvolvido, com os detalhes técnicos e informações necessárias para o acoplamento de novos módulos.

Princípios ágeis defendem que softwares em funcionamento são mais importantes que documentação abrangente ¹. Entendemos que para o nosso processo documentação é essencial. Para isso definimos as documentações mínimas necessárias que deverão ser preenchidas de forma objetiva sobre o desenvolvimento do software. Os documentos definidos foram os seguintes:

- Documento de Apresentação: documento inicial que deverá ser feito durante as primeiras conversas. Deverá conter uma breve descrição do objetivo do software, o cenário que motivou o seu desenvolvimento, os requisitos mapeados

¹ BECK *et al*, **Manifesto para Desenvolvimento Ágil de Software**.

e o responsável pelo seu desenvolvimento. Deverá ser atualizado até que seja feita a entrega final do sistema.

- Especificação técnica: documento contendo as principais informações técnicas, como linguagem de desenvolvimento, metodologias, versionamento, estrutura do software, módulos e também a maneira como os demais módulos devem ser desenvolvidos para conseguirem serem implementados no software. Com uma avaliação de risco aos pontos que devem ter atenção redobrada no desenvolvimento e em futuras melhorias, esse documento é essencial para o entendimento do que foi feito e compreensão de como devem ser implementadas as futuras melhorias. Deve ser atualizado em cada entrega. O time de desenvolvimento será responsável pela escrita da sua parte e o professor terá a função de validar todas as alterações.

3.1 INSTRUMENTALIZAÇÃO DO PROCESSO

Para apoiar no processo de gerenciamento foi utilizado uma ferramenta de gestão de projetos. Nesta pesquisa sugerimos o ASANA, que foi já foi apresentada no capítulo 2. No entanto o processo desenvolvido pode ser adequado e utilizado com outras ferramentas, desde que ela contenha os requisitos mínimos elencados anteriormente.

O ASANA apresenta as funcionalidades que satisfazem as necessidades encontradas no âmbito de pesquisa acadêmica sendo elas centralização das pesquisas em um local com uma visão macro simplificada para o orientador, possibilidade de comentários e anexos no mesmo local, criação de tarefas com data de entrega fixada e responsável atribuído, acompanhamento simples em tempo real do andamento de cada pesquisa, auxiliando na gestão e na tomada de decisão e também possui a funcionalidade de alertas via e-mail para o responsável sobre o

vencimento próximo, além de sinalizar por e-mail todas as alterações dentro do projeto, também é possível criar inúmeros projetos. Essas funcionalidades estão presentes em sua versão gratuita, que apresenta apenas a limitação de 10 membros por equipe fator que não impacta no nosso processo.

Foi explicitado anteriormente neste capítulo como deverá ser o processo de gerenciamento dentro dos grupos de pesquisa acadêmica, agora vamos mostrar um pouco de como implementar o processo proposto dentro da ferramenta ASANA.

Cada projeto deverá conter seu quadro específico conforme a Figura 17, com os membros do time definidos na equipe. Ao criar o projeto deverá ser escolhido o modelo de visualização padrão de quadro e a privacidade ser pública para a equipe. Será criado um projeto em branco no qual o professor ou o líder devem criar as colunas a seguir:

- Lista de Pendências: nessa coluna estarão presentes todas as tarefas mapeadas e levantadas como requisitos do projeto durante a fase lista de pendências, e em algum momento do projeto deverão ser executadas. Não conter as atividades planejadas para a Sprint atual.

- Sprint Atual: aqui deverão estar todas as atividades que ainda não iniciaram o desenvolvimento e fazem parte do planejamento a ser executado na Sprint atual.

- Em desenvolvimento: ao iniciar o desenvolvimento, o responsável pela tarefa deverá mover para essa coluna, e manter atualizada nos comentários sobre o andamento, já deverá ter estimativa de término e responsável atribuído.

- Documentação: após finalizar o desenvolvimento da atividade, o desenvolvedor deverá mover sua atividade para a coluna de documentação onde sinalizará aos outros membros que o desenvolvimento foi concluído, porém está sendo documentado.

- Incremento: nessa coluna o desenvolvedor deverá sinalizar que concluiu o desenvolvimento e a documentação. O incremento no software poderá ser atribuído para mais de uma pessoa em ocasiões onde necessite de ajuda para realizar o incremento.

- Bloqueado: toda tarefa que estiver com alguma pendência ou impedimento além da alçada do desenvolvedor deverá ser movida para essa coluna e atribuída ao professor e/ou ao líder para que ajudem a resolver os impedimentos.

- Feito: nessa coluna deverão estar todas as atividades concluídas e já implementadas no software.

Figura 17 - Quadro exemplo a ser utilizado dentro da ferramenta de gestão de projetos ASANA.

Lista de Pendências	Sprint Atual	Em Desenvolvimento	Documentação	Incremento	Bloqueado	Feito
+	+	+	+	+	+	+

Fonte: ASANA¹

Dentro do processo existem atividades que deverão ser atualizadas pelos responsáveis, transitando entre as colunas do quadro de acordo com a sua evolução. Essas atividades podem ser criadas por qualquer membro da equipe e podem conter as seguintes informações, preenchidas de acordo com a necessidade.

Título: breve descrição do que se trata a tarefa.

Responsável: é possível atribuir ao responsável a tarefa criada. O mesmo será avisado por e-mail.

Data de Conclusão: ao criar a tarefa é possível incluir qual a data de conclusão planejada, facilitando assim a visão do responsável sobre quais tarefas estão com a data de entrega mais próxima e também sinalizando para o professor possíveis atrasos.

Descrição: campo utilizado para que toda a equipe, ao abrir a tarefa, entenda qual é seu objetivo, e para que o responsável consiga a partir dessa descrição dar início a atividade.

Comentários: nesse campo poderão ser adicionados comentários sobre o andamento da tarefa, qualquer membro do time pode fazer um questionamento.

¹ ASANA.

Seguidores: é possível adicionar seguidores para que recebam notificações sobre quaisquer atualizações realizadas na tarefa.

Anexos: é possível incluir anexos de documentos ou imagens relacionadas a atividade, de forma a centralizar toda a documentação e histórico do projeto.

Subtarefa: é possível criar subtarefas da tarefa principal. Elas podem ser visualizadas em forma de *checklist* e pode possuir todos os atributos disponíveis para uma tarefa principal.

Copiar o *link* da tarefa: nessa opção é possível obter o *link* para ser compartilhado e levar diretamente a tarefa.

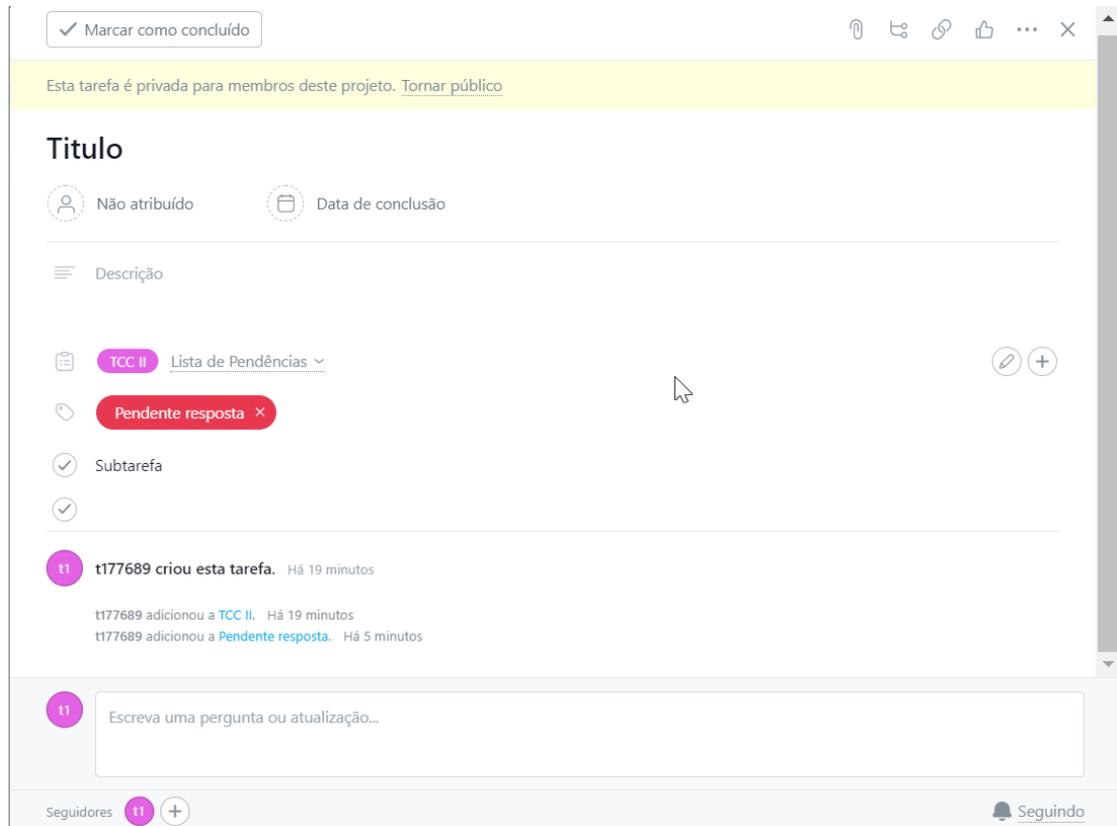
Adicionar a outro projeto: em casos específicos é possível copiar ou migrar atividades entre os projetos do grupo.

Adicionar *Tags*: essa opção é importante para auxiliar na gestão de forma prática, sendo possível criar *tags* cada uma com sua cor e significado, facilitando a visão sobre as pendências da atividade no quadro geral.

Campo de histórico: nesse campo ficam registradas todas as alterações feitas na atividade, com o registro de qual usuário alterou, o que alterou e quando alterou.

Todos os recursos mencionados estão disponíveis no ASANA para a organização das tarefas. A Figura 18 mostra esses campos na ferramenta. É sugerido que seja criada uma tarefa de informações gerais do projeto, onde no campo descrição contenha as orientações do projeto, tais como link de acesso para o repositório de documentos e link de acesso ao repositório de código fonte, entre outras informações úteis. Assim centralizando todas as informações e o acesso as demais ferramentas de forma simples.

Figura 18– Tarefa de exemplo a ser utilizado dentro da ferramenta de gestão de projetos ASANA.



Fonte: ASANA¹

Além da visualização kanban (Figura 1), a ferramenta também oferece aos usuários a possibilidade de ver as tarefas em forma de lista conforme apresentado na Figura 19. É possível realizar filtros nas opções “Todas as tarefas” e “Filtrar” e também ordenar a lista por data prevista de conclusão, responsável, curtidas e ordem alfabética, além de selecionar quais campos ficam visíveis na lista. São opções que podem ser personalizadas na ferramenta, de acordo com a sua preferência de visualização.

¹ *Ibid.*

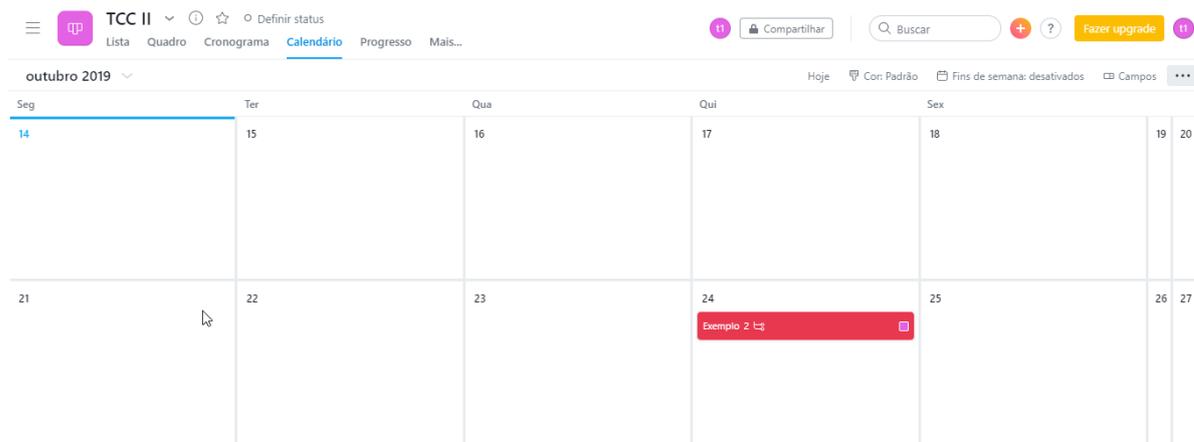
Figura 19 – Visualização por lista, com as opções de personalização.



Fonte: ASANA¹

Outra forma de visualização disponível, é no formato de calendário conforme apresentado na Figura 20, onde as tarefas apareceram distribuídas em um calendário mensal e cada uma estará de acordo com a data colocada na tarefa como “data de conclusão”.

Figura 20 – Visualização das tarefas no modo calendário.

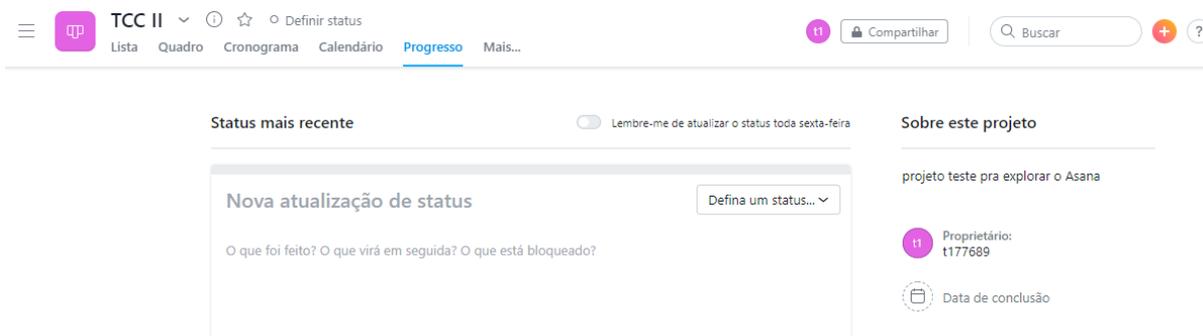


Fonte: ASANA¹

O ASANA tem funcionalidades que permitem ao time ações referentes ao projeto como um todo e não apenas sobre uma atividade específica. Uma delas é a opção “Progresso” conforme apresentado na Figura 21. Essa opção tem o objetivo de registrar todo o progresso do projeto gerando um status visível ao time todo. O professor deverá ser responsável por atualizar essa opção ao final de cada Sprint.

¹ *Ibid.*

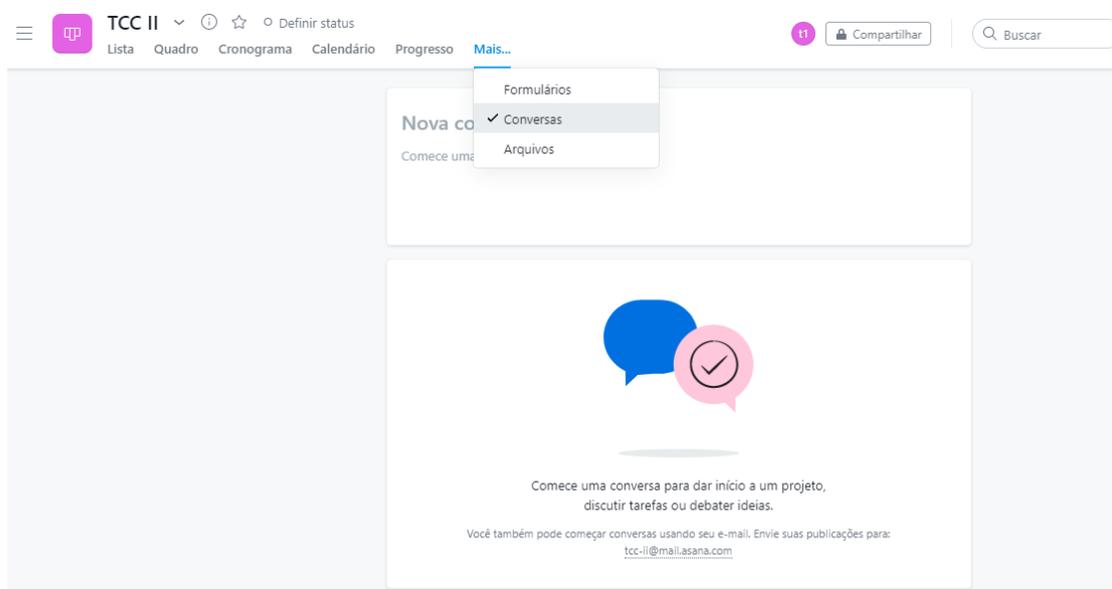
Figura 21– Funcionalidade “Progresso” para o registro do andamento do projeto como um todo.



Fonte: ASANA¹

Outra funcionalidade que poderá ser utilizada ao projeto como um todo, deverá ser o local de possíveis discussões e meio de recados gerais ao time. É a opção de “Conversas”, uma forma de chat disponibilizada para que toda equipe converse sobre assuntos de interesse mútuo, conforme apresentada na Figura 22.

Figura 22 – Funcionalidade “Conversas”, meio disponível para discussões gerais do projeto.



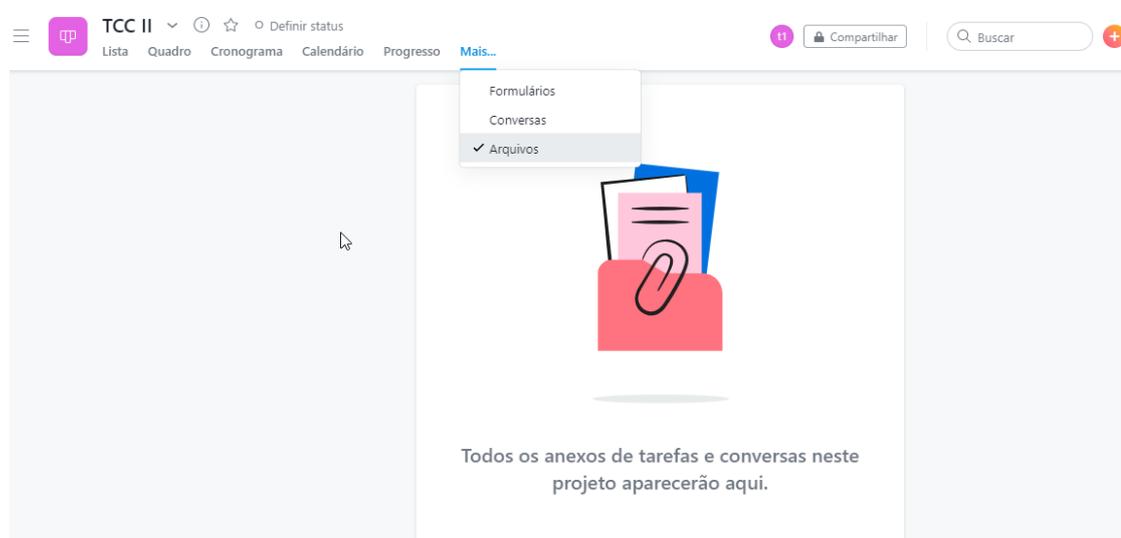
Fonte: ASANA¹

A última funcionalidade presente na versão gratuita é a de “Arquivos” conforme apresentado na Figura 23, local onde serão armazenados todos os anexos que forem incluídos

¹ *Ibid.*

nas atividades durante o projeto. Essa funcionalidade não substitui ao repositório de dados sugerido, pois os documentos ficaram disponíveis somente ao time e ao final do projeto permaneceram no ASANA somente para as pessoas com acesso ao projeto. As demais funcionalidades não apresentadas nesta pesquisa estão disponíveis apenas na versão *premium* da ferramenta.

Figura 23– Funcionalidade “arquivos”, centralizador dos documentos anexados nas atividades.



Fonte: ASANA¹

As ferramentas de repositório de documentos e repositório de código fonte auxiliarão na gestão do projeto em paralelo com a ferramenta ASANA. Como sugestão e ferramenta de repositório de documentos sugerimos o GOOGLE DRIVE que, conforme apresentado no capítulo 2, atende as necessidades básicas para exercer tal função. O motivo de cada aluno da faculdade ter um e-mail do GOOGLE, sem limite de armazenamento é um fator importante e decisivo na sugestão da ferramenta, evitando futuros problemas com memória de alocação.

Funcionando em paralelo, todos os membros da equipe devem ter acesso à pasta compartilhada na ferramenta de repositório de documentos. Devem ser anexadas toda a documentação gerada durante a pesquisa. Conforme os documentos sofrerem atualizações

¹ *Ibid.*

deverão ser novamente carregados na ferramenta com o mesmo nome, para que automaticamente seja feito o controle de versionamento.

Com o repositório de códigos fontes, a ferramenta sugerida nesta pesquisa será o BITBUCKET que, conforme apresentado no capítulo 2, atende a todas as necessidades esperadas pelo processo de uma ferramenta de repositório de códigos fontes. O professor cria o projeto, compartilha com o time e configurar as permissões que cada integrante terá, além de configurar o fluxo de aprovações necessários para as alterações serem implementadas no *software*. O incremento de novas funcionalidades deverá acontecer no final de cada *Sprint*, via ferramenta e o professor responsável deverá aprovar o incremento do novo módulo.

4 CONCLUSÃO

Com a definição do processo para gestão de projetos de desenvolvimento de software, é possível ter um desenvolvimento estruturado, tornando-o mais próximo do cenário encontrado no mercado. Todo o processo foi construído com referência a metodologias já consolidadas e utilizadas em larga escala no ambiente de desenvolvimento de sistemas, com adaptações necessárias ao cenário dos grupos de pesquisa acadêmicos, a fim de aumentar a maturidade dos sistemas gerados e proporcionar ao aluno uma experiência mais próxima do que será encontrado no mercado de trabalho.

O processo apoiado pela ferramenta de gestão de projeto ASANA proporciona ao professor maior controle e visibilidade sobre o andamento dos seus projetos, visto que acontecem diversas pesquisas de desenvolvimento de software em paralelo, as quais podem envolver diversos alunos. A ferramenta dá apoio para que o professor consiga manter o controle sobre quais atividades estão em responsabilidade de qual orientando, auxiliando nas cobranças de prazos de entregas, e centralizando os comentários e discussões.

O aumento da sinergia entre os envolvidos na pesquisa torna viável que diferentes orientadores do grupo realizem projetos integrados com orientandos não necessariamente de sua responsabilidade, mas membros do grupo como um todo. Sendo assim, será possível desenvolver sistemas complexos e que levem anos para serem construídos. Com uma metodologia de fácil integração diversos alunos poderão auxiliar contribuindo com pequenos módulos, e gerando um resultado final de alto valor agregado.

Com o processo de desenvolvimento faseado, o software tende a ser concebido com um molde receptível a futuros novos módulos, fator que aliado a alta disponibilidade e concentração dos documentos gerados no repositório definido e com o histórico da pesquisa na ferramenta de gestão, facilitam a entrada de futuras melhorias e incrementos no software por novos desenvolvedores, retirando a dependências dos criadores iniciais.

5 REFERÊNCIAS

ABES. **Portal Abes :: Estudo 2019 / Dados 2018**. Disponível em:

<<http://www.abessoftware.com.br/dados-do-setor/estudo-2019--dados-2018>>. Acesso em: 25 nov. 2019.

BECK, Kent; BEEDLE, Mike; VAN, Arie; *et al.* **Manifesto para Desenvolvimento Ágil de Software**. Disponível em: <<http://agilemanifesto.org/iso/ptbr/manifesto.html>>. Acesso em: 10 set. 2019.

KEELING, Ralph; BRANCO, Renato H F. **Gestão de Projetos - Uma abordagem global 3ª ED.** [s.l.]: Saraiva, 2014.

KNIBERG, Henrik; SKARIN, Mattias. **Kanban and Scrum - Making the Most of Both**. [s.l.: s.n.], 2010. Disponível em: <https://books.google.com.br/books?hl=pt-BR&lr=&id=Hx1KAgAAQBAJ&oi=fnd&pg=PR5&dq=kanban&ots=gmnV-vrS1F&sig=6tkzZM1N_XmTFey24bILGDnM68s&redir_esc=y#v=onepage&q=kanban&f=false>. Acesso em: 29 set. 2019.

LESSA, Rafael Orivaldo; ORIVALDO LESSA JUNIOR, Edson. **Modelos de Processos de Engenharia de Software**. Santa Catarina: [s.n.], 2006.

PRESSMAN, Roger S. **Engenharia de Software- (3ª edição)**. São Paulo: [s.n.], 1995.

PRESSMAN, Roger S. **Engenharia de Software uma abordagem profissional**. AMGH Edito. São Paulo: [s.n.], 2011.

RUSCHEL, Henrique; ZANOTTO, Mariana Susan; MOTA, Wélton Costa da. **Computação em nuvem**. p. 15, 2010. Disponível em:

<[https://www.pggia.pucpr.br/~jamhour/RSS/TCCRSS08B/Welton Costa da Mota - Artigo.pdf](https://www.pggia.pucpr.br/~jamhour/RSS/TCCRSS08B/Welton%20Costa%20da%20Mota%20-%20Artigo.pdf)>.

SCHWABER, Ken; SUTHERLAND, Jeff. **Guia do Scrum MR**. 2017. Disponível em:

<<https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese->

Brazilian.pdf>.