Guilherme Dutra Diniz de Freitas

Integração do GitHub com uma Ferramenta de Gerenciamento de Projetos de Software

Picos - PI
10 de novembro de 2017

Guilherme Dutra Diniz de Freitas

Integração do GitHub com uma Ferramenta de Gerenciamento de Projetos de Software

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como parte dos requisitos para obtenção do Grau de Bacharel em Sistemas de Informação, sob orientação do Professor Dennis Sávio Martins da Silva.

Universidade Federal do Piauí
Campus Senador Helvídio Nunes de Barros
Bacharelado em Sistemas de Informação

Picos - PI
10 de novembro de 2017

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

F866i Freitas, Guilherme Dutra Diniz de.

Integração do GitHub com uma ferramenta de gerenciamento de projetos de softwares / Guilherme Dutra Diniz de Freitas.– 2017.

CD-ROM : il.; 4 ¾ pol. (47 f.)

Trabalho de Conclusão de Curso (Curso Bacharelado em Sistemas de Informação) – Universidade Federal do Piauí, Picos, 2018.
Orientador(A): Prof. Dennis Sávio Martins da Silva

1. Gerência de Projetos. 2. Sistemas de Gerenciamento de Projetos. 3. Sistemas de Controle de Versão. I. Título.

CDD 005.43

INTEGRAÇÃO DO GITHUB COM UMA FERRAMENTA DE GERENCIAMENTO DE PROJETOS DE SOFTWARE

GUILHERME DUTRA DINIZ DE FREITAS

Monografia aprovada como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Data de Aprovação

Picos – PI, 27 de Novembro de 2017

Dennis Sávio Martins da Silva
Prof. Me. Dennis Sávio Martins da Silva
Orientador

Alcemir Rodrigues Santos
Prof. Dr. Alcemir Rodrigues Santos
Membro

Deborah Maria Vieira Magalhães
Profª. Dra. Deborah Maria Vieira Magalhães
Membro

Agradecimentos

Quero agradecer a conclusão deste trabalho a todos que de alguma forma contribuíram para eu chegar até aqui, e me ajudaram a superar todas as dificuldades encontradas durante o curso e mesmo nos momentos mais difíceis, caminharam comigo até o fim. De forma especial quero agradecer minha avó D. Emília Gomes Freitas que nunca desistiu de mim em nenhum momento da minha vida mesmo que todos os outros, até mesmo grande parte da minha família houvera desistido, e nunca mediu esforços mesmo com todas as dificuldades para que eu concluísse este curso, algumas vezes passando dificuldades mas nunca deixou de me apoiar tanto financeiramente quanto moralmente.

Agradeço a minha mãe pela criação que eu tive, que embora não foi a que eu queria, foi a que eu precisava Passei muito tempo sem entender mas agora entendo que fez o melhor que pôde, então peço desculpas pelas vezes que lhe desapontei e agradeço pela ajuda que me deu durante esse tempo.

Também agradeço aos demais familiares que me ajudaram ao longo desses quatro anos, alguns deles distantes e que nunca imaginei que um dia pudesse contar com a ajuda. Aurimary Vieira, Antônio Diniz, Francisca Áurea, Cleomar Almeida, Iraci Alves, Roberto, ao meu tio avô José Gomes (in memorian), Juraci, Josemira.

Aos amigos que fiz nessa caminhada e hoje tenho como família Arnaldo Cavalcante e Andrea Alves não só pela ajuda financeira, mas pelo apoio e pela amizade, e sempre fizeram o que puderam pra ajudar de alguma forma, ao Zé Maranhão e sua esposa Maria Lindomar que foram minha família no Piauí e também sempre pude contar com eles, assim como seus filhos Leonardo e Letícia Souza.

Aos meus amigos que foram irmãos no curso Rafael Araújo, Leonardo Augusto, Kécyo Kévinny, Fabrício Sousa e aos demais amigos que encontrei no curso e vou levar pra vida.

Aos meus professores Dennis Sávio Martins da Silva, Ivenilton Alexandre de Souza Moura, Patrícia Medynna, Alcinele Dalília e a todos os outros professores não citados que me ajudaram a chegar até aqui, todo o meu respeito e gratidão, pelo apoio, o conhecimento, a paciência e compreensão. Levarei sempre comigo os ensinamentos repassados e sabedoria de todos vocês. Muito obrigado!

Enfim, à todos que de forma direta ou indiretamente fizeram parte dessa etapa da minha vida, o meu mais sincero obrigado!

Just keep me where the light is...

(John Mayer)

Resumo

Diante da demanda por qualidade e redução de prazos e custos, as ferramentas de suporte à Engenharia de Software evoluem constantemente. Ferramentas de gestão de projetos de software tem como objetivo auxiliar a equipe de desenvolvimento de projeto de software a administrar todas as etapas do projeto de forma organizada e eficiente em busca de melhores resultados. O controle de versão é uma prática da Engenharia de Software, na qual o objetivo principal está na organização de projetos, por meio do gerenciamento de diferentes versões de um documento, possibilitando acompanhar o histórico de desenvolvimento, desenvolvendo paralelamente até resgatar o sistema em um ponto estável, sem alterar a versão principal. As ferramentas de gestão disponíveis hoje no mercado não trabalham junto com as de controle de versão, dificultando o trabalho das equipes de desenvolvimento, que são obrigadas a trabalhar com duas plataformas completamente separadas. O Sistema de Gerenciamento de Projetos (SGP), desenvolvido por alunos da Universidade Federal do Piauí, é uma ferramenta web de código aberto, com suporte à metodologia ágil Scrum. Neste trabalho foi desenvolvido um módulo de integração de controle de versão para o SGP e foram utilizadas as seguintes funções: Gerenciamento de repositório, gerenciamento de membros da equipe de projeto no repositório, visualização de dados de *branch* e de *commits*. Os experimentos foram realizados com 34 participantes voluntários, estudantes do curso de Bacharelado em Ciência da Computação e Sistemas de Informação em Fortaleza - Ceará, para avaliar o esforço, tempo e satisfação dos usuários ao utilizarem as ferramentas SGP - que integra nativamente um sistema de controle de versão - e Redmine - que não utiliza essa função. Em todos os quesitos avaliados do esforço e tempo gasto, o estudo apontou que a integração do Github no SGP é mais eficiente, em comparação ao Redmine.

Palavras-chaves: gerência de projetos, sistema de gerenciamento de projetos, sistemas de controle de versão.

Abstract

Faced with the demand for quality and reduction of deadlines and costs, software engineering support tools are constantly evolving. Software project management tools aims to assist the software project development team to manage all project steps in an organized and efficient manner in order to always obtain the best results. Version control is a Software Engineering practice, in which the main objective is to organize projects through the management of different versions of a document, making it possible to keep track of the development history, developing in parallel until the system is rescued at a stable point, without changing the main version. The management tools available on the market today do not work together with version control tools, making it difficult for development teams to work with two completely separate platforms. The SGP project management system, developed by students of the Federal University of Piauí, is an open source web tool with support for the agile Scrum methodology. In this work, a version control integration module was developed for the SGP, the following functions were used: Repository management, project team member management in the repository, visualization of branch and commits data. The experiments were carried out with 34 volunteer participants, who were students of the Bachelor's Degree in Computer Science and Information Systems in Fortaleza - Ceará, to evaluate the effort, time and satisfaction of the users when using the SGP tools that natively integrates a system of and Redmine that does not use this function. In all the evaluated aspects of the effort and time spent, a study pointed out that the integration of Github in the SGP is superior to Redmine.

Lista de ilustrações

Lista de abreviaturas e siglas

CSS	Cascading Style Sheets
DDS	Desenvolvimento distribuído de Software
ES	Engenharia de Software
HTML	HyperText Markup Language
JSON	JavaScript Object Notation
MIT	Massachusetts Institute of Technology
SGP	Sistema de Gerenciamento de Projeto

Sumário

1 Introdução

O gerenciamento de projetos tem como objetivo aplicar técnicas, conhecimentos e habilidades para tornar a execução de um projeto mais efetiva e eficaz, permitindo que as organizações utilizem os resultados dos projetos como objetivos do negócio (XAVIER, 2009).

Um projeto de software normalmente passa por um processo de gerenciamento de recursos, pessoas e de tempo. Com o código fonte do projeto não deve ser diferente. O código fonte de um software em desenvolvimento passa por várias alterações a todo momento durante o processo de desenvolvimento, e quanto maior a equipe de desenvolvimento, maior o número de alterações. Alterações estas, que no decorrer do projeto, são passíveis de serem revertidas ou integradas. Fazer isso de forma manual é bastante complicado, visto que pode gerar muitos erros e em caso de equipe geograficamente distribuída torna-se impossível, devido as pessoas do time de desenvolvimento estarem afastadas. Nestes casos, o mais indicado é a utilização de alguma ferramenta que permita esse gerenciamento de forma automatizada.

Um dos recursos criados para auxiliar nesse gerenciamento foram os sistemas de controle de versão, que são ferramentas que ajudam a gerenciar o desenvolvimento colaborativo de software, facilitando que uma equipe de desenvolvedores trabalhem no mesmo arquivo, permitindo que as alterações sejam integradas e guardadas e um repositório (YUILL, 2008).

O Sistema de Gerenciamento de Projetos (SGP) é uma ferramenta web, de código aberto, desenvolvida por alunos da Universidade Federal do Piauí. A ferramenta atualmente auxilia o projeto de desenvolvimento de software utilizando a metodologia ágil Scrum, apoiando o Desenvolvimento Distribuído de Software (DDS) e nela foi possível aplicar o uso do controle de versionamento através da API (sigla em inglês para Interface de Programação de Aplicativos) Octokit, responsável por fazer toda a comunicação entre o Rails e a plataforma GitHub.

1.1 Contexto e Problema

Em uma pesquisa realizada no Brasil no ano de 2017 (CABRAL, 2017), foram entrevistadas diversas empresas de desenvolvimento de software, onde foi constatado que todas as empresas utilizavam algum controle de versão na produção de software e que 77,8% delas utilizavam o Git como sistema de controle de versão. Outro estudo realizado em empresas de desenvolvimento de software de grande porte (RADAIESKI; FROES; BANDEIRA, 2015) constata que em todas elas são utilizadas ferramentas de gerenciamento de projetos para controle e/ou planejamento de projetos de software.

Poucas ferramentas de gerenciamento de projetos de software *open source* disponíveis no mercado hoje contam com sistema de controle de versão integrado, e as que possuem esse recurso as vezes não trazem nativamente na ferramenta, sendo necessária a instalação de componentes externos e tornando sua aplicabilidade pouco prática e difícil, como é o caso da ferramenta Redmine; e quando possuem, se limitam a funcionalidades de acompanhamento de informações, não permitindo alterar informações de repositório, informações de membros de equipes, dentre outras funcionalidades propostas neste trabalho.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo desse trabalho é propor a integração de funcionalidades do sistema de controle de versão GitHub ao SGP (Sistema de Gerenciamento de Projetos).

1.2.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Fazer o levantamento das ferramentas e esforços necessários para realização da integração com sistema SGP;
- Definir quais funções do GitHub integrar ao SGP;
- Validar a integração do GitHub ao sistema SGP, comparando o SGP com sistemas de gerenciamento de projetos que possuem essa funcionalidade;
- Analisar o tempo gasto pelos usuários ao manusear sistemas de controle de versão com e sem a utilização da integração com o Github;
- Analisar o esforço gasto pelos usuários com a integração do GitHub na ferramenta SGP e em outra similar.

1.3 Estrutura do Trabalho

O capítulo 2, o Referencial Teórico, são detalhados todos os assuntos relevantes para o entendimento correto deste trabalho, já no capítulo 3 descreve os trabalhos relacionados, fazendo uma comparação a este trabalho, no capítulo 4 é descrita todos os materiais e metodologias aplicadas na elaboração deste trabalho, no capítulo 5 são descritos todos os passos de condução deste experimento desde a seleção dos participantes até os resultados obtidos, e por ultimo temos o capítulo 6, onde concluímos nosso estudo.

2 Referencial Teórico

Este capítulo está organizado da seguinte forma. Na Seção 2.1, apresentamos as definições, características e usabilidade dos Sistemas de Controle de Versão e na Seção 2.2 trata da Gestão de Projetos com suas definições, finalidades, ferramentas de gestão e metodologias ágeis de gestão.

2.1 Sistemas de Controle de Versão, Git e Github

O controle de versão é um recurso da Engenharia de Software que tem como princípio a organização de projetos mediante o gerenciamento de diferentes versões de um documento. Esse recurso possibilita acompanhar o histórico de desenvolvimento, e até mesmo retomar o sistema em um ponto em que estava funcional, tudo isso sem modificar a versão principal. Muito usado por desenvolvedores, é o modo mais confiável de atestar a qualidade do código-fonte, principalmente quando se trata de uma equipe de grande porte (PALESTINO, 2015).

Há várias soluções livres de sistemas de controle de versão. Uma pesquisa realizada pelo *Stack Overflow*[1], site utilizado por desenvolvedores reconhecido mundialmente, elege o Git como ferramenta de controle de versão mais utilizada e popular dentre os desenvolvedores de todo o mundo, seguida do Subversion e o *Concurrent Version System* (CVS).

O Git é um software de controle de versões de documentos, que torna possível o trabalho em um único diretório, permitindo fazer alterações no projeto, gravando a documentação e os comentários, e ainda registra tudo que foi salvo ou aprovado. O Git também possibilita criar espaços separados para testes ou para projetos diferentes; desfazer modificações que estão com problemas, voltando para a versão em que o sistema estava funcional; e trabalhar em equipe de forma mais simples e segura, sendo capaz de criar e editar documentos de projetos onde vários integrantes de uma equipe podem contribuir simultaneamente sem correr o risco de terem alterações sobrescritas.

Um grande diferencial do Git em relação à maioria dos sistemas de controle de versão é que ele não armazena as informações como uma lista de mudanças por arquivo, tratando as informações e mantendo-as como um conjunto de arquivos. Ao invés disso o Git considera os dados como um conjunto de *snapshots* (espécie de fotografia). Para isso, no Git há como criar a qualquer momento várias ramificações do mesmo projeto, chamadas de *branches*, que são cópias do estado do projeto em determinado momento. Por exemplo, imagine que seja necessário realizar uma determinada alteração no código fonte, mas alteração não deve estar disponível para ninguém além do desenvolvedor responsável até que tudo

[1] https://insights.stackoverflow.com/survey/2017#technology

esteja funcionando. Então, criar essa *branch* é uma boa escolha, pois esse desenvolvedor poderá trabalhar até acertar todos os detalhes até que o sistema esteja funcional, para então fazer um *merge* (mesclagem dos dados para o projeto original). Toda vez que é salva alguma alteração (*commit*) de um projeto, é como se uma fotografia de todos os arquivos fosse tirada e armazenada.

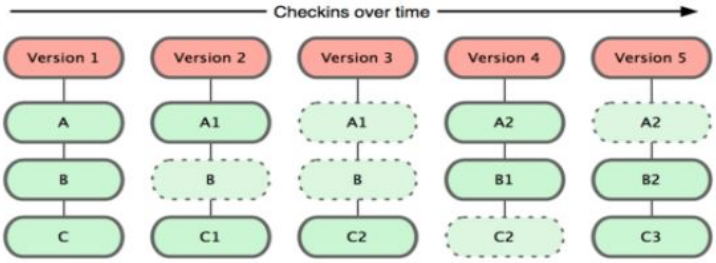


Figura 1: Tratamento dos dados - Git. Fonte: Git

Na Figura 1 podemos ver como o Git armazena os dados em forma de versões, através de *branch* e *commit*. Dentre os locais de armazenamentos em nuvem com arquivos enviados via Git, destaca-se o GitHub[2], que possui a maior comunidade de desenvolvedores da plataforma Git e não exige a instalação em vários computadores sempre que for necessário acessar algum arquivo. O GitHub é um serviço web que disponibiliza diversas funcionalidades aplicadas ao Git de gerenciamento de projetos, diversas possibilidades de integrações, gerenciamento da equipe, e documentação abrangente. De maneira sucinta, pode-se usar gratuitamente o GitHub para armazenar projetos pessoais ou profissionais. Pode-se dizer usando outras palavras que o GitHub é um complemento do Git. Além disso, a maioria dos projetos sobre desenvolvimento *open-source* estão nele, o que possibilita acompanhá-los, ajudando ao informar *bugs* ou até mesmo enviar correções.

2.2 Gestão de Projetos

A Gestão de Projetos (GP) como ciência surgiu no início do século XX, especificamente na década de 1920, com a invenção do diagrama de Gantt. Este diagrama, criado pelo engenheiro americano Henry L. Gantt, tentava resolver o problema da programação das atividades, através da distribuição das mesmas em um gráfico onde se podia visualizar rapidamente a duração, o início, o fim das atividades (SILVA, 2007).

Com o passar do tempo, reconheceu-se que a abordagem tradicional da gestão de projetos não era suficiente para dar solução aos problemas surgidos na gestão de esforços. Na segunda metade do século XX verificou-se um acréscimo significativo na complexidade dos processos na área da indústria e serviços, necessitando de um apoio maior da área de Gestão de Projeto. Como resposta a esta necessidade, foram surgindo ao longo do tempo

[2] https://github.com/features#documentation

várias organizações profissionais dedicadas à área de Gestão de Projeto, contribuindo para sua expansão em diversas áreas do mercado (SILVA, 2007).

Uma das áreas de gestão de desenvolvimento de software que tem crescido nos últimos anos, e é objeto de muitas pesquisas, é o Desenvolvimento Distribuído de Software (DDS). Esse crescimento deve-se ao fato que tem se tornado gradualmente custoso e menos competitivo desenvolver software no mesmo espaço físico, na mesma empresa e até mesmo no mesmo país. O DDS é tipificado sempre que um ou vários recursos humanos que estiverem envolvidos em um mesmo projeto estiverem fisicamente longe dos demais (AUDY, 2007). Já Carmel (1999) conceitua DDS como um modelo de desenvolvimento onde as pessoas envolvidas em um determinado projeto estão espalhados geograficamente, temporalmente ou até mesmo com diferenças culturais.

Além do DDS, a Gestão de Projetos de Software conta com diversos outros recursos para facilitar o trabalho da equipe de desenvolvimento de projetos, como por exemplo as inúmeras ferramentas que auxiliam na gestão de projetos com diversas abordagens de Engenharia de Software (ES) e funcionalidades diferentes, dentre as quais podemos destacar a Redmine, a ProjectLibre e a Open Project.

2.2.1 Redmine

Redmine[3] é uma ferramenta web de código aberto, desenvolvida na linguagem em Ruby on Rails, multiplataforma, compatível com diversos sistemas gerenciadores de banco de dados (como o SQLite, MySQL, PostgresSQL, dentre outros), e que pode ser considerada uma das mais influentes soluções *open source* de gerenciamento de projetos no mundo.

Redmine apoia todas as necessidades para um gerenciamento eficaz de projeto, e tem a vantagem de possuir vários recursos online que contribuem para a qualidade dos projetos executados, como por exemplo: o suporte a múltiplos projetos, gráfico de Gantt e calendário. Juntamente com recursos básicos de gerenciamento de projetos, o Redmine possui uma documentação abrangente, dispondo de uma *wiki* e um repositório de informações.(LESYUK, 2013).

É importante salientar que o Redmine possui integração com controle de versões por meio da instalação de um *plugin* externo, mas essa integração é bastante limitada e não aborda todas as funções propostas neste trabalho, como gerenciamento do repositório e gestão dos membros do projeto.

A figura 2 apresenta a interface do Redmine com o nome do projeto, seus menus de opções disponíveis e um exemplo de lista de gerenciamento de *issues* de um projeto.

[3] http://www.redmine.org/

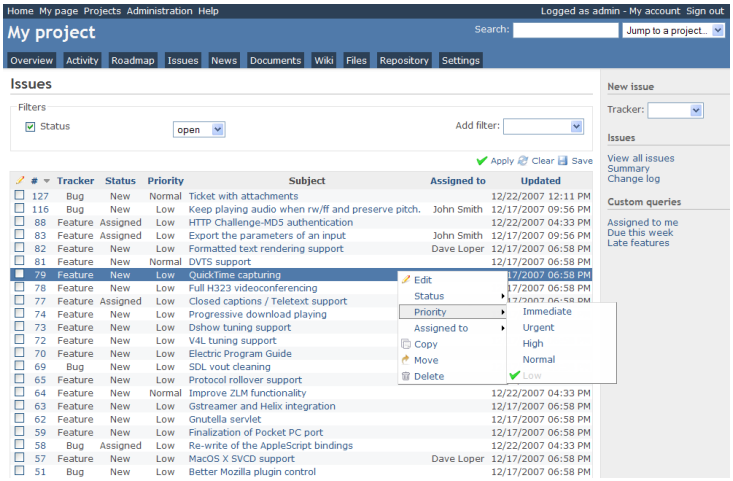


Figura 2: Tela com a lista de issues do Redmine. Fonte: Redmine

2.2.2 ProjectLibre

ProjectLibre[4] é uma ferramenta desenvolvida em Java, de código aberto, que ajuda no gerenciamento do projeto. Foi lançada em 2012, é multiplataforma, e constitui uma conhecida alternativa de código aberto ao Microsoft Project[5]. A ferramenta é principalmente focada em longas fases e interações, e ajuda na visualização do projeto como um todo durante o seu ciclo de vida, além de propiciar um ambiente colaborativo online que pode ser usado e acessado de qualquer lugar (ABRAMOVA; PIRES; BERNARDINO, 2016), mas não possui qualquer integração com sistemas de controle de versão.

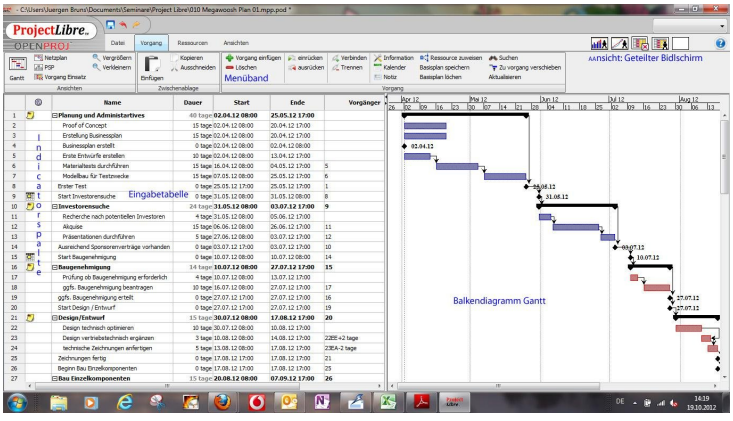


Figura 3: Interface do ProjectLibre, com sua lista de tarefas e o Gráfico de Gantt. Fonte: ProjectLibre

A figura 3 apresenta a interface do ProjectLibre no estilo *Ribbon*, bastante similar ao estilo da Microsoft Project.

4 https://www.projectlibre.com/
5 https://products.office.com/pt-br/project

2.2.3 Open Project

OpenProject é uma ferramenta de gerenciamento de projetos, de código fonte aberto, que permite definir tarefas em forma de lista ordenada, alocar os recursos disponíveis (pessoas, materiais e máquinas) que serão usados em cada tarefa e estimar seus custos.

A ferramenta oferece a possibilidade de visualizar o gráfico de Gantt, PERT (*Program Evaluation and Review Technique* ou Programa de avaliação e técnica de revisão), e diagramas para gerenciamento das tarefas do projeto. Diferentemente de algumas alternativas *open source*, o Open Project inclui o gerenciamento de recursos humanos, permitindo um melhor gerenciamento de tarefas e da equipe, entretanto não possui nenhuma integração com sistemas de controle de versão.

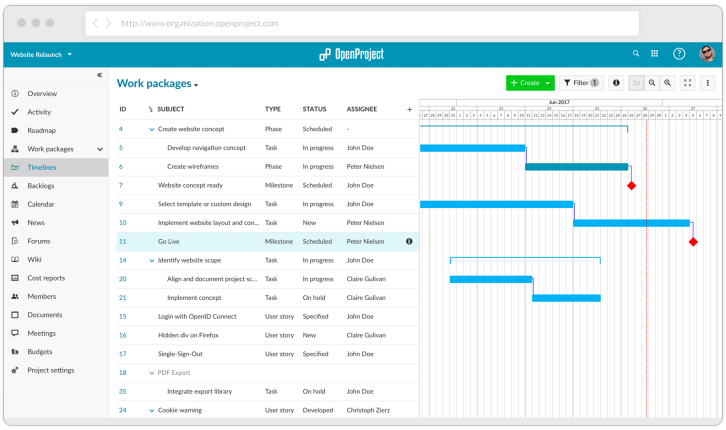


Figura 4: Interface do Open Project. Fonte: Open Project

A figura 4 mostra a tela de tarefas do Open Project, com a lista de menus do aplicativo, a lista de tarefas e o gráfico de Gantt com a estimativa de tempo para realização das tarefas.

2.2.4 Metodologias Ágeis

Dos anos 80 ao início dos anos 90, havia uma concepção geral de que a melhor forma para se obter um software de qualidade era por meio de um planejamento elaborado, do uso de métodos de análise e projeto apoiado por ferramentas CASE (*Computer-Aided Software Engineering* ou Engenharia de Software Assistida por Computador), e de um processo de desenvolvimento de software controlado e rigoroso (SOMMERVILLE et al., 2011).

Esse tipo de abordagem de desenvolvimento de software, conhecida como prescritiva, trouxe uma grande contribuição no trabalho de ES e um roteiro de desenvolvimento satisfatoriamente eficaz para as equipes de desenvolvimento de software (PRESSMAN, 2016). Em contrapartida, essas abordagens trazem também uma sobrecarga no planejamento, projeto e documentação do sistema, sendo justificado o uso das abordagens prescritivas

quando se trata de um grande número de equipes de desenvolvimento ou quando o sistema é crítico e complexo de ser trabalhado (SOMMERVILLE et al., 2011).

Em um mercado moderno, as condições de desenvolvimento mudam velozmente e novos desafios aparecem de forma inesperada. Aprender sobre o mercado e construir um produto para satisfazê-lo são práticas importantes para o desenvolvimento de soluções de software bem sucedidas (POPPENDIECK; POPPENDIECK, 2009). Entretanto, quando abordagens "pesadas"de desenvolvimento são aplicadas aos sistemas que os requisitos mudam constantemente, a sobrecarga do desenvolvimento torna-se grande a ponto de dominá-lo e dificulta que a equipe entregue o software que o cliente precisa e quando ele precisa.

Devido à popularização e ao grande número de menções aos processos "leves", que surgiam como alternativa para sanar as fraquezas dos projetos utilizando abordagens prescritivas de ES (PRESSMAN, 2016). Em fevereiro de 2001, um grupo de dezessete pensadores da área de software - dentre eles desenvolvedores, autores e consultores praticantes dessas abordagens "leves- se uniram e deram origem ao Manifesto Ágil[6], declaração de princípios que fundamenta o desenvolvimento ágil de software, e originou o termo "métodos ágeis", oficializando a partir de então, uma nova maneira de se desenvolver software.

O Manifesto Ágil consolidou valores e princípios de métodos e abordagens ágeis existentes e criou um movimento ágil mais forte e organizado na industria de desenvolvimento de software (BECK et al., 2013).

Os valores ágeis são:

- Valorização dos **indivíduos e interações** mais que processos e ferramentas;
- O **software em funcionamento** possui importância maior do que uma documentação abrangente;
- Foco na **colaboração com o cliente** maior que a negociação de contratos;
- Priorização na **resposta às mudanças**, maior do que na obediência ao plano do projeto.

Os princípios Ágeis são:

- A prioridade é satisfazer o cliente, por meio da entrega adiantada e contínua de software de valor.
- Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento, pois os processos ágeis devem se adequar a mudanças para que o cliente possa tirar vantagens competitivas.
- Entregar o software funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos.

[6] http://agilemanifesto.org/

- Pessoas relacionadas ao negócio devem trabalhar em conjunto com os desenvolvedores, durante todo o curso do projeto.
- Construir projetos ao redor de indivíduos motivados, dando a eles o ambiente e suporte necessário e confiando que farão seu trabalho.
- O método mais eficiente e eficaz de transmitir informações para uma equipe de desenvolvimento é a conversa cara a cara.
- Software funcional é a medida primária de progresso.
- Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter os passos do projeto constantes.
- Contínua atenção à excelência técnica e bom design aumenta a agilidade.
- Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito.
- As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis.
- Em intervalos regulares a equipe reflete sobre sua eficiência, ajusta e otimiza seu comportamento adequadamente.

2.2.5 Scrum

O Scrum é um *framework* dinâmico para o gerenciamento de projetos a partir de práticas iterativas e incrementais que buscam propiciar mais valor ao negócio (SCHWABWER; SUTHERLAND, 2016). Utilizado com mais frequência por organizações que desenvolvem software, o Scrum é um *framework* do qual as pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível. Scrum não é um processo ou técnica em específico, e sim uma proposta de trabalho onde podem ser aplicados vários processos e várias técnicas com o propósito da eficácia no desenvolvimento de sistemas (CRUZ, 2013).

O Scrum baseia-se na objetividade, com os papéis bem definidos e a facilidade no aprendizado. É um modelo aberto e não previsível, visto que o *framework* não detalha o que deve ser feito e muito menos resolve de forma instantânea os problemas da empresa, mas dá visibilidade aos problemas e serve como uma espécie de guia para sua resolução. Ele oferece um conjunto de práticas que tem como objetivo manter o gerenciamento do projeto visível aos usuários.

Alguns elementos são chave para a implementação do Scrum em qualquer empresa:

- A disposição de equipes auto-gerenciáveis e bem organizadas;
- Progresso de desenvolvimento por meio de *Sprints*;

- Os requisitos de produto estarem organizados em uma lista chamada de *Product Backlog*;
- Conceito de tempo fechado (*timebox*) onde todas as tarefas dentro do Scrum tem um tempo máximo para serem cumpridas (SABBAGH, 2014).

Todo o progresso do Scrum tem como base os *sprints*, que são ciclos de desenvolvimento com curto período, onde a equipe foca em atingir pequenas metas específicas. A primeira etapa de um *Sprint* é a reunião de planejamento, chamada de *Sprint Planning*, e realizada pela a equipe de desenvolvimento (*Scrum Team*), em conjunto com o *Product Owner*, pessoa que define e prioriza os itens que compõem o *Product Backlog*.

A próxima etapa é a de execução, onde a equipe Scrum detalha as tarefas necessárias para a implementação do que foi solicitado pelo *Product Owner*. Durante a *Sprint*, o Time Scrum realiza reuniões diárias (*Daily Meetings*) onde é averiguado o progresso do desenvolvimento. Ao final do *Sprint* é realizada uma reunião de validação da entrega chamada de *Sprint Review*, onde o *Product Owner* e outros *Stakeholders* do projeto podem verificar se o objetivo do *Sprint* foi atingido. Logo em seguida, é realizada apenas pela equipe do projeto uma reunião chamada de *Sprint Retrospective*, onde o *Sprint* é avaliado sob a perspectiva do processo, equipe ou produto a respeito de quais foram os acertos e os erros com o propósito de melhorar o processo de trabalho (SCHWABWER; SUTHERLAND, 2016).

3 Trabalhos Relacionados

Neste capítulo, muitas pesquisas podem ser encontradas na literatura abordando a utilização de ferramentas de gerenciamento de projetos como grande auxiliar no processo de desenvolvimento de software.

Mello (2013), desenvolveram uma biblioteca chamada de Gitabs para a plataforma Git, com o objetivo de solucionar o problema de gestão de software de forma bem geral, podendo assim abranger diferentes tipos de projetos. A biblioteca funciona com base em quatro requisitos, que são: a definição dos dados a serem armazenados, a inserção dos dados, a execução e a entrega de tarefas. Possibilitando, assim, um gerente de projetos registrar junto ao repositório de um projeto informações externas à codificação do software propriamente dito, e um colaborador executar tarefas e entrega-las partindo de informações inseridas pelo gerente de projetos.

Oliveira et al. (2013), desenvolveram uma ferramenta que também visa realizar o controle das etapas do desenvolvimento de software, o SGPS (Sistema de Gerenciamento de Projeto de Software), sendo dividido em vários módulos, sendo eles: Gestão de requisitos, de tempo, de arquitetura, de mudanças, de testes, de riscos, e de implantação.

Okada e Sass (2015), projetaram uma ferramenta de gerenciamento de projeto com a metodologia baseada no PMBOK, que tem como funções: realizar cadastro do projeto, planejar os recursos de tarefas e objetivos, realizar cadastro de recursos e custos necessários no projeto, revisões, seleção e avaliação de pessoal, elaboração de relatórios e o gráfico de Gantt.

Petersen (2016) desenvolveram uma ferramenta capaz de gerenciar os artefatos de projetos, os membros deste projeto, usuários que não são alocados ainda em nenhum projeto, designa tarefas para a equipe de trabalho, define os status das tarefas e subtarefas. A ferramenta é baseada na metodologia ágil Scrum e possui recursos de gráficos para melhor acompanhamento do projeto.

Este trabalho se difere dos outros por fazer a união entre funcionalidades que eles possuem separadamente e acrescentar funções de integração inexistentes. Enquanto em Mello (2013) foi desenvolvida uma biblioteca integrada ao Git para fazer a gestão de projetos, a experiência de gestão é bastante comprometida pelo fato de não possuir uma gama completa de recursos que vise facilitar a vida do gerente de projetos. Já os outros trabalhos como os de Oliveira et al. (2013), Okada e Sass (2015) e de Petersen (2016) oferecem uma boa quantidade de recursos que auxiliam o gerente de projetos, mas nelas não há qualquer tipo de integração com nenhum sistema de controle de versão.

4 Materiais e Métodos

Neste capítulo, será apresentada a ferramenta SGP e o seu módulo que aplica a integração do sistema de controle de versão GitHub, bem como a Gem Octokit responsável pela integração e as funções do Github aplicadas, que é a proposta central desse trabalho.

4.1 O Sistema de Gerenciamento de Projetos SGP

O SGP (Sistema de Gerenciamento de Projetos) foi desenvolvido na Universidade Federal do Piauí por estudantes do curso de Sistemas de Informação e serviu de base para esse trabalho. A ferramenta contém um módulo que dá suporte ao gerenciamento de projetos utilizando a metodologia Scrum pra suportar outras metodologias no futuro.

O Sistema SGP foi projetado para funcionar na plataforma Web, suportado em praticamente qualquer dispositivo computacional, e necessitando apenas de um navegador Web para poder utilizá-lo. Esta ferramenta foi construída utilizando a linguagem de programação Ruby com o *framework Rails* e das tecnologias de *Front-end* da wWb HTML (HyperText Markup Language), CSS (Cascading Style Sheets) e JavaScript (LIMA, 2017). O código fonte do sistema SGP pode ser obtido, através da internet no repositório disponibilizado no GitHub[1]

4.2 Ferramentas e Tecnologias Utilizadas

O Ruby[2] é uma linguagem de programação multiparadigma, criada por Yukihiro Matsumoto no ano de 1995, inspirada em suas linguagens favoritas Lisp, Smalltalk, Perl e Python. Essa linguagem foi escolhida por ter foco na produtividade, exibilidade e velocidade de entrega oferecida pelos diversos paradigmas que a linguagem oferece, além de ter uma sintaxe enxuta e ser multiplataforma.

O Rails é um *meta-framework* de desenvolvimento de aplicações Web disponível sob a licença MIT (Massachusetts Institute of Technology). O Rails tem seu foco no seu modo simples e produtivo, tendo o aprendizado muito reduzido se comparado a outros *frameworks*.

O Rails foi escolhido por oferecer diversos recursos que facilitaram a produção da aplicação como: convenções; estrutura pré-definida de configurações da aplicação que poupa tempo ao fazê-las; geração automática de código, facilitando o processo de desenvolvimento e ameniza o esforço por parte do programador; suporte a diversos bancos de dados e sistemas operacionais (HARTL, 2016).

[1] www.github.com/guilhermeddf/SGP
[2] https://www.ruby-lang.org/pt/about

Ao se desenvolver uma aplicação web, é fundamental a utilização de HTML e CSS na construção do *front-end* da aplicação, pois são as linguagens comumente suportadas pelo navegador web para apresentação de conteúdo. O HTML é empregado na marcação e definição dos elementos que compõem a página. O CSS é a linguagem de folha de estilo, que serve para definir a aparência dos elementos existentes na página (GOODMAN, 2002).

Octokit é uma API (*Application Programming Interface*) do GitHub para o Ruby com licença do MIT que permite o acesso a informações de contas e de repositórios do GitHub mediante a autenticação. Com ele foi possível agregar todas as funcionalidades desejadas ao SGP.

4.3 Recursos do GitHub aplicados ao SGP

Na implementação da integração do Github com o SGP, as seguintes funcionalidades foram adicionadas à ferramenta:

- Gerenciamento de repositórios;
- Gerenciamento de membros com acesso ao repositório do projeto;
- Visualização de *branches* criadas no repositório;
- Visualização de *commits* realizados no repositório do projeto.

Foram dadas ao usuário as opções de criar, excluir e visualizar os dados do repositório, tais como nome, tipo do repositório (público ou privado), se possui ou não lista de *issues*, etc. Ao gerenciar os membros do projeto o usuário poderá adicionar ou remover um membro do projeto do repositório, revogando seus privilégios, além de visualizar os seus dados. O usuário também poderá visualizar os dados de *branches* criadas, como os seus *commits*, os membros do projeto que estão trabalhando nela, hora e data de criação entre outros. Já na visualização dos *commits*, o usuário poderá ver a quantidade de *commits* realizados, o membro da equipe que realizou o *commit*, a data e a hora.

5 Estudo Experimental

O propósito desse estudo foi avaliar o esforço desempenhado e o tempo gasto pelos membros de uma equipe de desenvolvimento de software com uma ferramenta integrada a um sistema de controle de versão e o SGP, na gestão de projetos de software. As hipóteses levantadas nesse estudo são:

- O esforço empenhado para gerenciar os processos do desenvolvimento de software com a utilização de ferramentas de gerenciamento integradas a um sistema de controle de versão é menor que com a utilização da integração do Git com o SGP.
- O tempo gasto para gerenciar os processos de desenvolvimento de software tanto na ferramenta de gerenciamento quanto no repositório do projeto é menor quando se usa o SGP integrado a um controle de versão.
- A satisfação para gerenciar os processos de desenvolvimento de software com a utilização de ferramenta integrada a um sistema de controle de versão é menor que com a ferramenta SGP.

Para a realização desse estudo, comparamos a ferramenta SGP integrada ao GitHub com uma ferramenta também com integração ao controle de versão Redmine[1]. Por ter alta aceitação no mercado de desenvolvimento de software, ser de código aberto e gratuita, funcionar na plataforma web, não utilizar integração com sistema de controle de versão nativamente, além de ter sido desenvolvida com a mesma tecnologia que o SGP (Ruby on Rails).

5.1 Seleção de Participantes

Para realizar o estudo das ferramentas foram selecionados grupos de participantes voluntários. Foram selecionados e participaram do estudo alunos dos cursos de graduação em Bacharelado em Ciência da Computação e de Bacharelado em Sistemas de Informação em 2 instituições de ensino superior em Fortaleza - Ceará. Os alunos convidados participaram de forma espontânea do experimento, concordando com os termos apresentados no termo de consentimento livre e esclarecimento disposto no APÊNDICE A. Foram levantados os perfis dos participantes da amostra, que representam a população de usuários das ferramentas que são objeto de estudo, por meio do questionário do perfil do participante, que se encontra no APÊNDICE B. Foram selecionados um total de 34 estudantes, cujo conhecimento em relação os requisitos necessários para participar de uma equipe de

[1] http://www.redmine.org/

desenvolvimento de projetos de software está disposto conforme demonstrado na Figura 5.

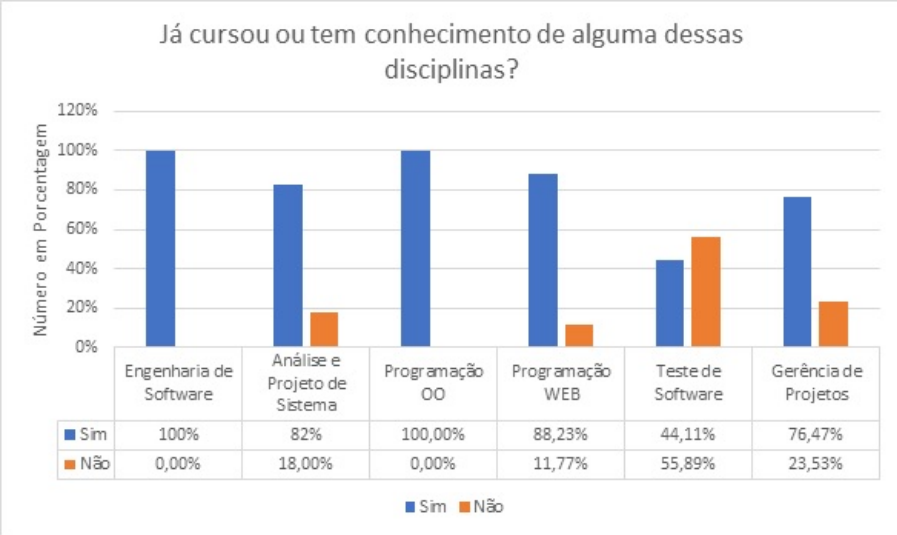


Figura 5: Habilidade dos Participantes do Estudo.

Além das habilidades, foi levantada a experiência dos participantes em projetos de software, a fim de avaliar o conhecimento no projeto de desenvolvimento de software. Os resultados são apresentados nas Figuras 6 e 7.

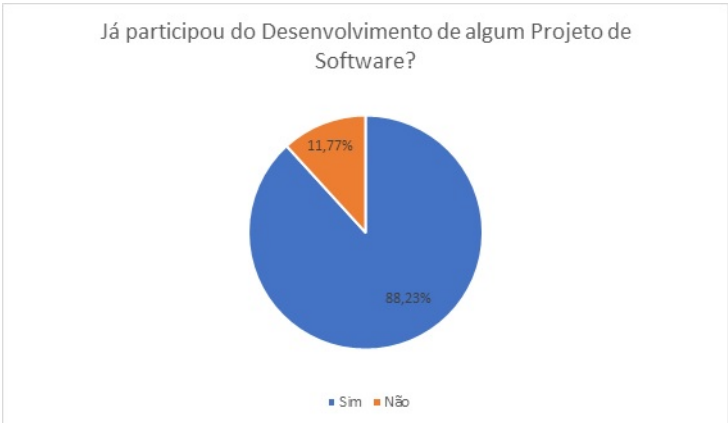


Figura 6: Participantes que já trabalharam em Equipe no Desenvolvimento de Software.

Também foi perguntado aos participantes sobre o seu nível de conhecimento no manuseio da plataforma Github, o resultado é mostrado na Figura 8.

Diante dos gráficos podemos considerar que as amostras que foram selecionadas para o estudo são representativas de equipes reais, levando em consideração as habilidades e a experiência em desenvolvimento de projetos de software dos participantes selecionados para representar reais equipes de desenvolvimento.

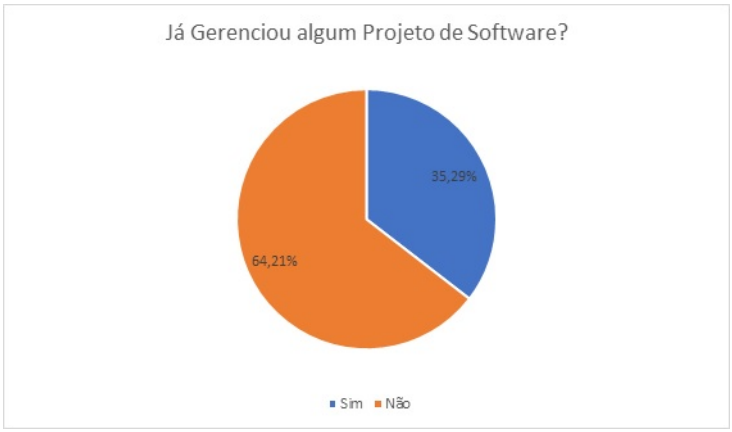


Figura 7: Participantes que já Gerenciaram Projetos de Desenvolvimento de Software.

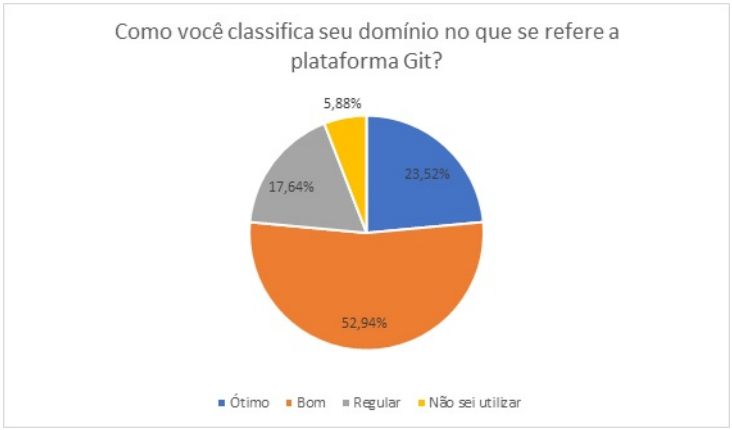


Figura 8: Nível de Conhecimento do Participante na Plataforma GitHub.

5.2 Condução do Experimento

Durante a realização do experimento, os participantes selecionados foram distribuídos de forma aleatória em 6 equipes de desenvolvimento, contendo de 5 a 6 membros. Cada equipe de desenvolvimento recebeu uma proposta de problema que pode ser encontrado no APÊNDICE C para desenvolver utilizando uma das ferramentas: *Redmine* ou SGP. Três grupos iniciaram utilizando cada ferramenta, e depois cada grupo resolveu o mesmo problema proposto empregando a outra opção. Ao final do experimento foi avaliada a experiência individual de cada um dos participantes, por meio do preenchimento do formulário de pós experimento disponível no APÊNDICE D.

5.3 Resultados

O questionário de pós-experimento foi dividido em 3 grupos de perguntas (A, B e C), que tinham como objetivo avaliar o esforço desempenhado, o tempo despendido e a satisfação dos usuários com a utilização das ferramentas utilizadas nesse estudo.

5.3.1 Grupo A (Avaliação da Sensação de Esforço)

O grupo de questões A, teve como objetivo avaliar o **esforço empenhado** pelos usuários no gerenciamento de projetos e repositórios da plataforma Git pelos dois sistemas de gerenciamento de projeto, *Redmine* e SGP. O Grupo de questões A considerava as seguintes variáveis: Gerenciamento do repositório do projeto, acompanhamento das informações do repositório do projeto, gerenciamento dos membros do projeto, acompanhamento das *branches* criadas no projeto e o acompanhamento dos *commits* do projeto e os seus responsáveis. Nesse bloco de perguntas foi solicitado que os usuários avaliassem o esforço em uma escala de 1 à 5, sendo 1 para "pouquíssimo esforço"e 5 para "muitíssimo esforço".

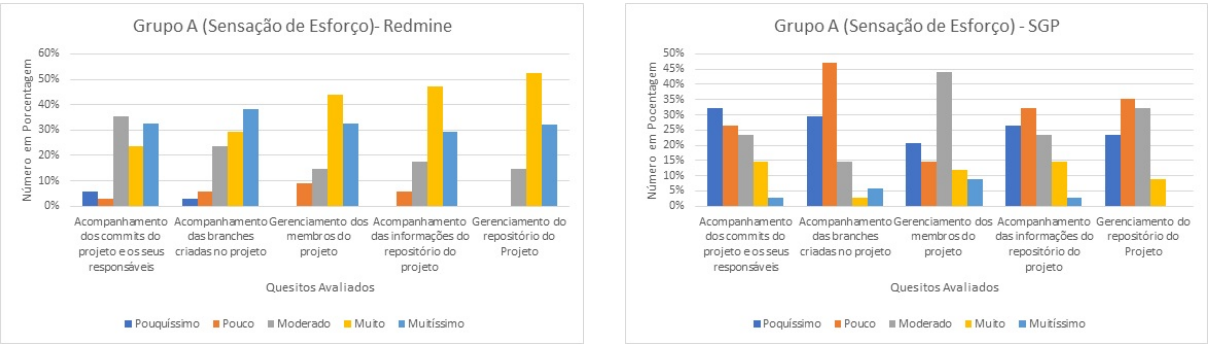


Figura 9: Resultados do Grupo A por Quesito Avaliado.

Como pode ser visto na Figura 9, em todos os quesitos avaliados pelo bloco de perguntas a integração da plataforma Git ao SGP torna o processo de gestão de projetos e a gestão do repositório Git do projeto menos cansativo, fazendo com que o esforço empenhado pelo usuário da equipe de desenvolvimento seja menor, comparado com a ferramenta *Redmine*, que segundo os dados coletados, o esforço empenhado pelos usuários foi considerado alto.

Comparando o cenário geral do bloco de questões A, notamos que segundo os dados coletados, o esforço empenhado foi reduzido drasticamente, quando utilizado a integração com o sistema de controle de versão Git no SGP.

Na **opção pouquíssimo**, enquanto houveram 45 marcações para o SGP, que significa o menor esforço para manusear as ferramentas, a ferramenta *Redmine* obteve apenas 3. Já na **opção pouco** houveram 53 marcações do SGP, contra 8 do *Redmine*, na **opção moderado** houveram 47 marcações para o SGP, contra 36 marcações do *Redmine*, na **opção muito** houveram 18 marcações para o SGP, enquanto o *Redmine* obteve 67, por último temos a **opção muitíssimo** onde o SGP obteve apenas 7 marcações enquanto o *Redmine* obteve 55, significando que, segundo os participantes a ferramenta *Redmine* exige mais esforço ao utilizá-la com o Git integrado, do que a ferramenta SGP. A Figura 10 representa o cenário geral dessa avaliação.

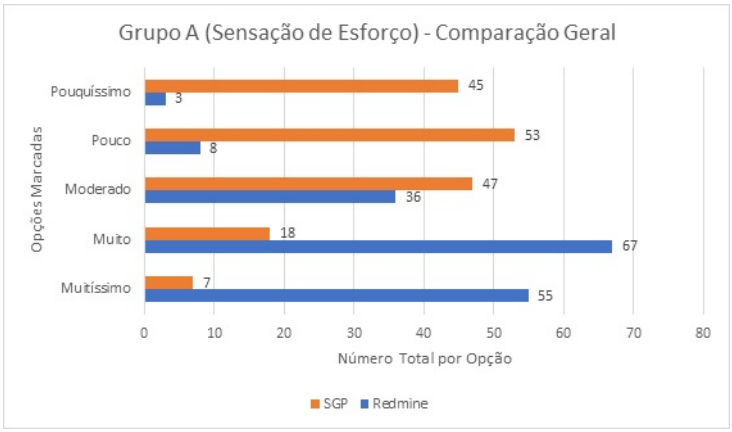


Figura 10: Cenário Geral da Avaliação do Grupo A de Perguntas.

5.3.2 Grupo B (Avaliação da Sensação de Tempo)

O grupo de questões B do questionário de pós-experimento, teve como objetivo avaliar o **tempo gasto** dos usuários ao utilizar ferramentas de gestão integradas ao sistema de controle de versão Git, *Redmine* e SGP, as questões de avaliação foram: Gerenciamento do repositório do projeto, acompanhamento das informações do repositório do projeto, gerenciamento dos membros do projeto, acompanhamento das *branches* criadas no projeto e o acompanhamento dos *commits* do projeto e os seus responsáveis. Nesse questionário também foi solicitado que os que os usuários participantes avaliassem os itens do grupo numa escala de 1 à 5, sendo 1 para "pouquíssimo tempo"e 5 para "muitíssimo esforço".

A Figura 11 mostra que em todos os quesitos avaliados, a integração do SGP com o Git atendeu as expectativas, diminuindo o tempo gasto pelos usuários para realizar tarefas em ambas as ferramentas se comparado ao *Redmine* que não traz todos os elementos da integração como é proposto nesse trabalho.

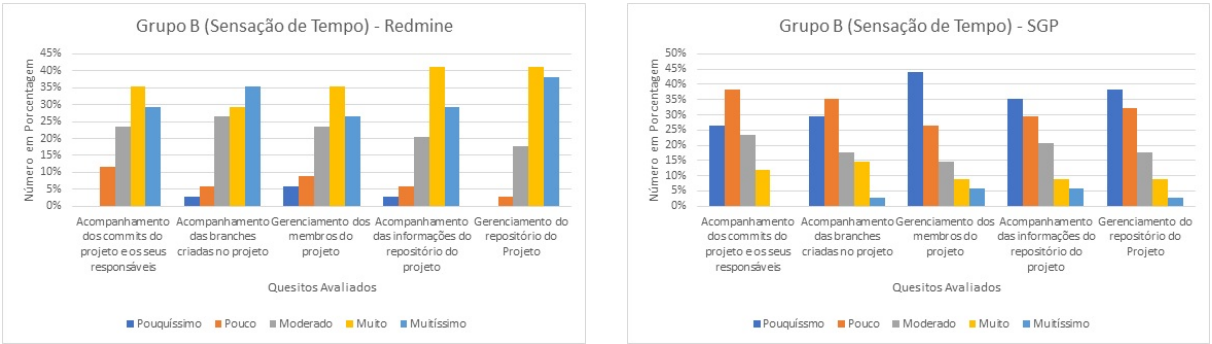


Figura 11: Resultados do Grupo B por Quesito Avaliado.

Comparando o cenário geral do bloco de questões B, notamos que o tempo gasto foi reduzido drasticamente, segundo os dados coletados, quando utilizado a integração do controle de versão Git no SGP.

Na **opção pouquíssimo** houveram 59 marcações para o SGP, que significa o menor tempo gasto para manusear as ferramentas, a ferramenta *Redmine* obteve apenas 4. Já na **opção pouco** houveram 55 marcações do SGP, contra 12 do *Redmine*, na **opção moderado** houveram 32 marcações para para o SGP, contra 36 marcações do *Redmine*, na **opção muito** houveram 18 marcações para o SGP, enquanto o *Redmine* obteve 62, por ultimo temos a **opção muitíssimo** onde o SGP obteve apenas 6 marcações enquanto o *Redmine* obteve 54, significando que segundo os dados coletados dos participantes, a ferramenta *Redmine* exige mais tempo gasto ao utilizá-la com o Git, do que a ferramenta SGP. O gráfico da figura 12 representa o cenário geral dessa avaliação.

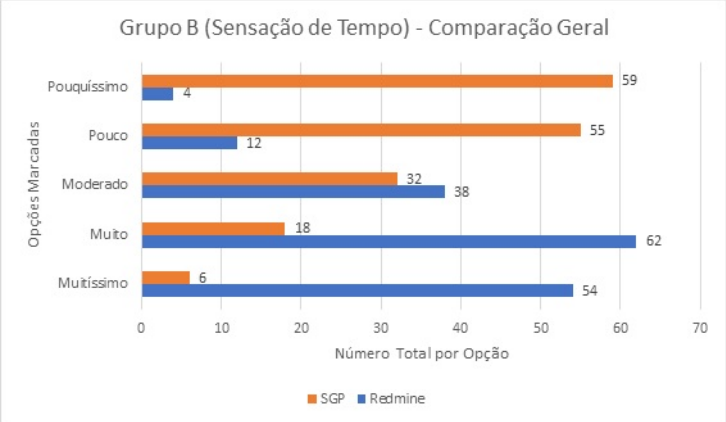


Figura 12: Cenário Geral da Avaliação do Grupo B de Perguntas.

5.3.3 Grupo C (Avaliação da Satisfação)

O grupo C de questões teve como objetivo avaliar a aceitação e satisfação com relação as ferramentas SGP e *Redmine*, ou seja, se os usuários ficaram contentes com o recurso da ferramenta e se a expectativa foi atendida, nesse bloco de questões foi solicitado que os usuários avaliassem os itens em uma escala de 1 à 5 sendo 1 para "pouquíssimo"e 5 para "muitíssimo", as seguintes questões: Como você considera a qualidade da ferramenta? O quão conveniente é utilizar a ferramenta? O quão útil você considera a ferramenta? O quanto você recomenda a ferramenta? O quão satisfeito você ficou ao utilizar esta ferramenta? A Figura 13 demostra se a integração do sistema de controle de versão torna o gerenciamento de projetos de software através de ferramentas mais satisfatório.

Comparando o cenário geral do bloco de questões C, notamos que a satisfação do usuário com o gerenciamento do seu projeto foi significantemente maior quando utilizada a integração com o sistema de controle de versão Git no SGP. Enquanto houveram 5 marcações na **opção pouquíssimo** para o SGP, a ferramenta *Redmine* obteve 26. Já na **opção pouco** houveram 17 marcações do SGP, contra 37 do *Redmine*, na **opção moderado** houveram 39 marcações para para o SGP, contra 48 marcações do *Redmine*, na **opção muito** houveram 64 marcações para o SGP, enquanto o *Redmine* obteve 34,

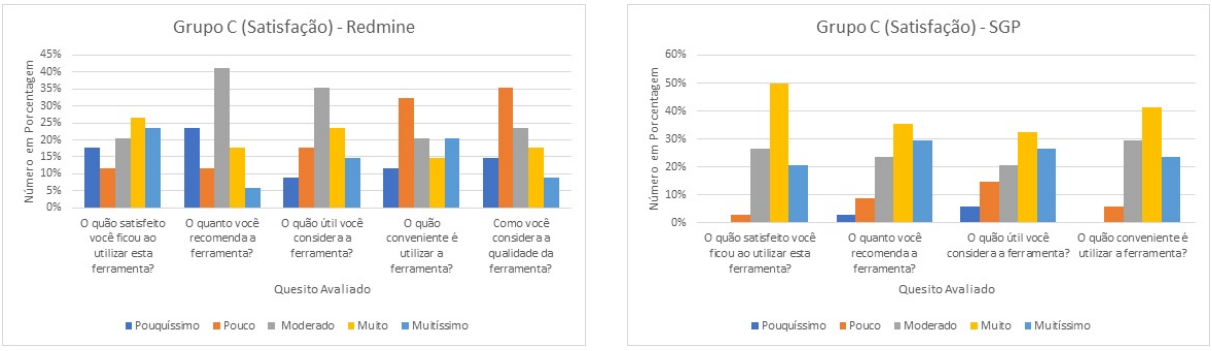


Figura 13: Resultados do Grupo C por Quesito Avaliado.

por último temos a **opção muitíssimo**, que avalia a satisfação ao utilizar a ferramenta, onde o SGP obteve 45 marcações enquanto o *Redmine* obteve 25, significando que segundo os participantes a ferramenta *Redmine* não é tão satisfatória no manuseio, do que a ferramenta SGP. O gráfico da figura 14 representa o cenário geral dessa avaliação.

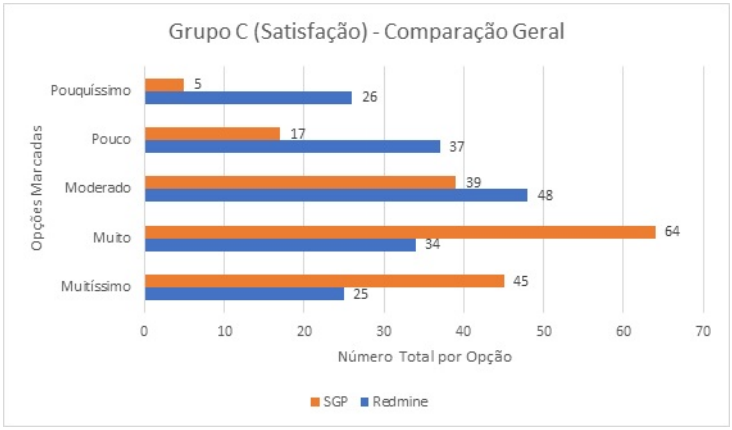


Figura 14: Cenário Geral da Avaliação do Grupo C de Perguntas.

6 Conclusão

Mesmo que já haja ferramentas de gestão de projetos de software que utilizem integração com o Github, as características e limitações levantadas revelam que ainda há espaço para o desenvolvimento de soluções mais completas para a gestão de projetos ágeis de software integrada a controle de versionamento.

A proposta de um sistema de gerenciamento de projetos que se utilize da integração com o Git, bem como a verificação se o mesmo ofereceu algum ganho ou melhoria em comparação as diversas ferramentas que já existem no mercado foi um dos assuntos que nortearam a ideia deste trabalho, visto que a maioria das ferramentas disponíveis no mercado não trazem essa funcionalidade, e quando trazem, é de forma limitada. Com relação as hipóteses levantadas, conseguimos levantas os seguintes resultados:

- O esforço empenhado para gerenciar os processos do desenvolvimento de software com a utilização de ferramentas de gerenciamento integradas a um sistema de controle de versão é menor que com a utilização da integração do Git com o SGP.

  Sim, considerando o experimento realizado, em todos as questões apresentadas aos participantes do estudo a sensação de esforço foi drasticamente reduzido se comparado com a outra ferramenta avaliada, o Redmine

- O tempo gasto para gerenciar os processos de desenvolvimento de software tanto na ferramenta de gerenciamento quanto no repositório do projeto é menor quando se usa o SGP integrado a um controle de versão.

  Sim, segundo o experimento realizado, em todas as questões apresentadas aos participantes, a sensação de tempo foi drasticamente reduzida se comparado com o Redmine.

- A satisfação para gerenciar os processos de desenvolvimento de software com a utilização de ferramenta integrada a um sistema de controle de versão é menor que com a ferramenta SGP.

  Sim, também considerando o estudo realizado, os dados apontaram que a satisfação se mostra maior quando utilizado o SGP integrado ao Github em comparação ao Redmine.

Com a realização dos experimentos com os 34 alunos de graduação na área de tecnologia da informação, que responderam questionários à respeito do esforço e do tempo gasto para gerenciar projetos em duas ferramentas de gestão de integradas à sistemas de controle de versão,SGP e *Redmine*, pudemos perceber que a ferramenta de gestão de projetos de software SGP com integração com sistemas de controle de versão, diminui esforços e

o tempo que normalmente a equipe de desenvolvimento iria ter em gerenciar duas ferramentas distintas de forma separada, e trazendo também uma satisfação maior ao usar a ferramenta. Consideramos então que o uso da ferramenta de gestão de software SGP, integrada com sistemas de controle de versão Git, trazem benefícios significativos para a equipe de desenvolvimento de projetos, se comparado com a ferramenta concorrente, *Redmine*.

6.1 Trabalhos Futuros

Embora esse trabalho tenha obtido resultados considerados satisfatórios no que se refere à integração do Git com o SGP, nem todos os recursos do Git foram integrados e avaliados nessa ferramenta. Como proposta de trabalho futuro, pretende-se aumentar o número de funções disponíveis no Git integradas ao SGP, como por exemplo: o gerenciamento de diversas equipes em um único projeto; o gerenciamento de repositórios privados; e algumas funções que estão disponíveis na versão paga da plataforma GitHub, como o gerenciamento de repositórios privados e o gerenciamento de multiplas equipes de desenvolvimento em um único repositório.

Referências

ABRAMOVA, V.; PIRES, F.; BERNARDINO, J. Open source and proprietary project management tools for smes. *Journal of Information Systems Engineering and Management*, v. 1, n. 3, p. 1–10, 2016. Citado na página 15.

AUDY, J. *Desenvolvimento Distribuído de Software: Desenvolvimento de software com equipes distribuídas*. [S.l.]: Elsevier Brasil, 2007. Citado na página 14.

BECK, K. et al. Manifesto para desenvolvimento ágil de software. 2001. *Disponível na URL:< http://agilemanifesto. org/iso/ptbr/>(03/10/2017)*, 2013. Citado na página 17.

CABRAL, F. T. *Identificando quais são as Ferramentas de Gerência de Configuração e Integração Contínua mais Utilizadas Pelo Mercado de TI de Fortaleza*. [S.l.]: Monografia, 2017. Citado na página 10.

CARMEL, E. *Global software teams: collaborating across borders and time zones*. [S.l.]: Prentice Hall PTR, 1999. Citado na página 14.

CRUZ, F. *Scrum e PMBOK unidos no Gerenciamento de Projetos*. [S.l.]: Brasport, 2013. Citado na página 18.

GOODMAN, D. *Dynamic HTML: The Definitive Reference: A Comprehensive Resource for HTML, CSS, DOM & JavaScript*. [S.l.]: "O'Reilly Media, Inc.", 2002. Citado na página 22.

HARTL, M. *Ruby on rails tutorial: learn Web development with rails*. [S.l.]: Addison-Wesley Professional, 2016. Citado na página 21.

LESYUK, A. *Mastering Redmine*. [S.l.]: Packt Publishing Ltd, 2013. Citado na página 14.

LIMA, G. F. d. S. *Uso de Estratégias de Gamifcação no Processo de Gestão e Desenvolvimento de Software*. [S.l.]: Monografia, 2017. Citado na página 21.

MELLO, E. M. d. S. Gitabs: uma extensão ao sistema git para gestão de projetos. 2013. Citado na página 20.

OKADA, K. C.; SASS, G. G. Projeto de uma ferramenta de apoio à gestão de projetos de software. *ANAIS DO ENIC*, v. 1, n. 1, 2015. Citado na página 20.

OLIVEIRA, L. C. de et al. Sistema de gestão de projetos de software-sgps. *Revista de Engenharia e Tecnologia*, v. 5, n. 4, p. Páginas–1, 2013. Citado na página 20.

PALESTINO, C. M. C. Estudo de tecnologias de controle de versões de software. 2015. Citado na página 12.

PETERSEN, J. Mean web application development with agile kanban. 2016. Citado na página 20.

POPPENDIECK, M.; POPPENDIECK, T. *Implementando o desenvolvimento Lean de Software: do conceito ao dinheiro*. [S.l.]: Bookman Editora, 2009. Citado na página 17.

PRESSMAN, R. S. *Engenharia de software*. [S.l.]: Makron books Sao Paulo, 2016. Citado 2 vezes nas páginas 16 e 17.

RADAIESKI, G. d. S.; FROES, M. F.; BANDEIRA, D. L. Fatores que influenciam o uso e a seleção de um software de gerenciamento de projetos: aplicando o modelo de liberatore e pollack-johnson em uma empresa pública brasileira de desenvolvimento de software. *ReA UFSM: Revista de Administração da UFSM. Santa Maria, RS. Vol. 8, n. 1,(mar. 2015), p. 9-25*, 2015. Citado na página 10.

SABBAGH, R. *Scrum: Gestão ágil para projetos de sucesso*. [S.l.]: Editora Casa do Código, 2014. Citado na página 19.

SCHWABWER, K.; SUTHERLAND, J. Um guia definitivo para o scrum: As regras do jogo. 2013. 2016. Citado 2 vezes nas páginas 18 e 19.

SILVA, M. *Microsoft office project 2007: depressa e bem*. [S.l.: s.n.], 2007. Citado 2 vezes nas páginas 13 e 14.

SOMMERVILLE, I. et al. *Engenharia de software*. [S.l.]: Addison Wesley São Paulo, 2011. Citado 2 vezes nas páginas 16 e 17.

XAVIER, C. M. da S. *Gerenciamento de Projetos: como definir e controlar o escopo do projeto*. [S.l.]: Saraiva, 2009. Citado na página 10.

YUILL, S. *"Concurrent Versions System."Software Studies: A Lexicon*. [S.l.: s.n.], 2008. Citado na página 10.

Apêndices

APÊNDICE A – Termo de Consentimento

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS – PICOS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Termo de Consentimento Livre e Esclarecimento
Baseado nas Diretrizes Contidas na Resolução CNS Nº466/2012, MS.

Eu, _____, convidado(a) a participar voluntariamente do estudo experimental do trabalho intitulado "Integração do GitHub com uma Ferramenta de Gerenciamento de Projetos de Software", recebi do graduando Guilherme Dutra Diniz de Freitas, responsável pela execução do experimento, as seguintes informações que me fizeram compreender sem objeções e dúvidas as seguintes informações:

- Esse estudo é destinado a avaliar o gerenciamento do desenvolvimento de sistemas com e sem a utilização do sistema de controle de versão integrado ao sistema de gerenciamento de projetos;
- Que a importância deste estudo é a de possibilitar a disponibilização de uma ferramenta que facilite o processo de gestão de desenvolvimento de sistemas de computacionais;
- Que o estudo almeja alcançar o esforço e o tempo empenhado para gerenciar o desenvolvimento de software com e sem a utilização do sistema de controle de versão integrado à ferramenta de gestão de projetos de software;
- Que, minha participação nesse estudo não implicará em nenhum risco à minha saúde física ou mental;
- Que, em qualquer momento desse estudo, eu poderei me recusar a continuar participando desse estudo;
- Que os resultados obtidos por meio de minha participação não possibilitarão a minha identificação, exceto ao responsável por este estudo;
- Que o estudo realizado iniciou-se às ____:____ na data ___/___/____ e seu término às ____:____ na data ___/___/____;

Considerando que fui informado(a) dos objetivos e da relevância do estudo proposto, de como será minha participação, dos procedimentos, e de que não há riscos decorrentes deste estudo, declaro o meu consentimento em participar da pesquisa e concordo que os dados obtidos por meio dessa sejam utilizados para fins científicos (publicações). Estou ciente de que receberei uma via desse documento.

_____
Assinatura do Participante ou Responsável Legal

APÊNDICE B – Formulário de perfil do participante

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS – PICOS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Caracterização de Participante

Nome:	
Email:	Curso:

1 - Qual seu nível de ensino? [ ] Técnico [ ] Graduação [ ] Pós-graduação

( ) Concluído ( ) Cursando ( ) Abandonei

2 - Sua atuação? [ ] Acadêmico [ ] Mercado profissional

3 - Já cursou ou tem conhecimento de alguma dessas disciplinas?

Engenharia de Software	( ) sim	( ) não
Análise e Projeto de Sistemas	( ) sim	( ) não
Programação OO	( ) sim	( ) não
Programação WEB	( ) sim	( ) não
Teste de Software	( ) sim	( ) não
Gerência de Projetos	( ) sim	( ) não

4 - Já participou do desenvolvimento de algum projeto de software?
( ) sim
( ) não

5 - Já gerenciou o desenvolvimento de algum projeto de software?
( ) sim
( ) não

6 – Como você classifica seu domínio no que se refere a plataforma Git?
( ) Ótimo () Bom () Regular ( ) Não sei utilizar

OBS: Esse questionário tem como objetivo caracterizar os participantes desse experimento e auxiliar na compreensão dos resultados desta pesquisa, então é importante que seja preenchido de maneira fiel.

APÊNDICE C – Problema

Problema

1. Definição do Problema

Com a crescente demanda de criação de páginas *web* por empresas e pessoas, muitas tecnologias foram desenvolvidas para agilizar esse processo, como a criação de CMSs (*Content Manger System* – Sistema Gerenciador de Conteúdo) populares como *Wordpress* para blogs, *Joomla* para portais, *Magento* para *web-Commerce* (Comércio eletrônico), etc. Porém essas ferramentas não dão completo apoio as necessidades dos desenvolvedores, os *layouts* são poucos personalizáveis e a curva de aprendizado para construção de *templates* é longa, outra dificuldade enfrentada é que esses CMSs vem consigo todos os recursos que o desenvolvedor precisa, e os recursos contruidos pelo desenvolvedor é de dificil integração com esses CMSs, como por exemplo: um programador tem a necessidade adicionar a sua página um sistema de sorteio e integra-lo ao site construído com algum CMS; ou um *webdesign* ao construir um *template* precisa integrar ao gerenciador de conteúdo.

Muitos desenvolvedores optam por construir seu próprio gerenciador de conteúdo para ter o total controle do site criado, apesar dos icríveis recursos desses *frameworks* que aceleram o processo de desenvolvimento construir um CMS é um processo que pode demorar, resultando na insatisfação dos clientes.

2. Descrição da Solução

O sistema gerenciador de conteúdo em *Ruby on Rails* baseado em web service permitirá a comunicação com outras aplicações através do formato de dados json. Este gerenciador possuirá um template/layout padrão de página que poderá ser usado pelo usuário, webdesign ou programador. O administrador poderá gerenciar páginas, postagens, categorias, e configurações do site, como definir título, sub-título, número de postagens por página, e se o site estará disponível ou não, na sessão de postagens o administrador terá os seguintes camops disponíveis: título, resumo, e conteúdo, onde ele poderá escrever e formatar sua postagem e também realizar o upload de imagens, e definir se a postagem será publica, VIP para os visitantes cadastrador ou como rascunho (condúdo para publicação futura). Na sessão de gerencimanto de páginas o administrador poderá definir um título e um conteúdo estático que será exibido para os visitantes.

Os usuários do site feito com o CMS em Ruby on Rails além de poderem navegar pelas postagens e páginas estáticas também poderão comentar nas publicações. Para os usuários terem acesso as postagens VIPs, estres deverão credenciar-se no site, onde poderão logar-se e ter acesso ao conteúdo exclusivo e também poder publicar comentários nas postagens.

Além do módulo do adminstrador que é utilizado para genrenciar o conteúdo na página, a aplicação web também disponibilizará seu conteúdo em formato JSON no módulo de web service, permitindo assim que desenvolvedores e webdesigns integrem suas próprias aplicações ou layouts/templates com este CMS.

3. Arquitetura do Sistema

3.1 Diagrama UML de classes:

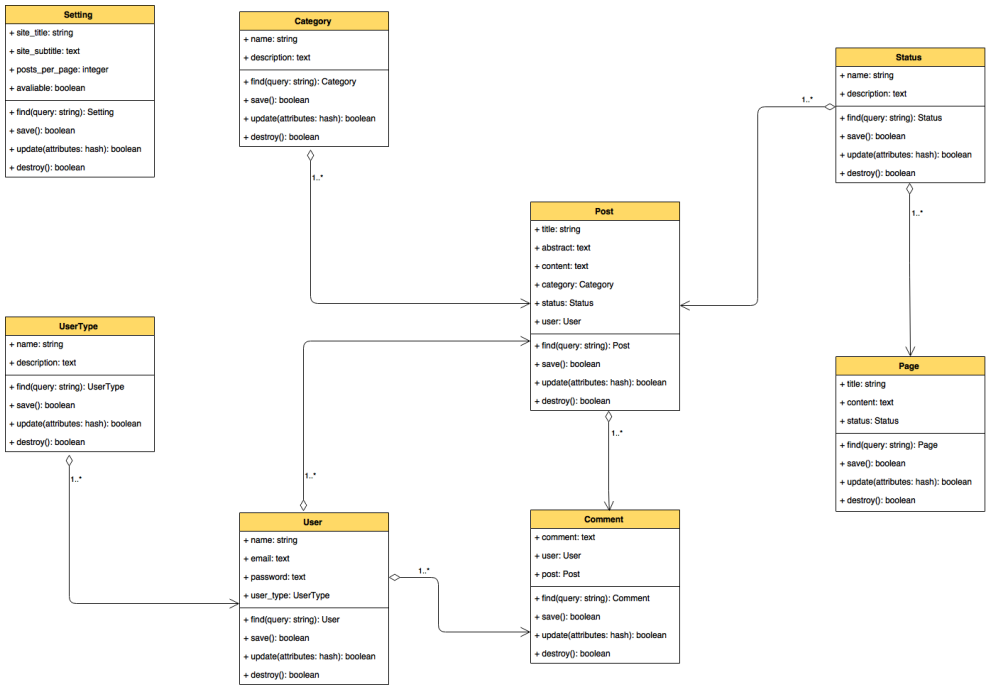


Figura 2 – Diagrama UML de classes

3.2 Product backlog e sprints backlog:

Product backlog:

IBL	TEMA	USER STORY	CRITÉRIOS DE ACEITAÇÃO	VALOR DE NEGÓCIO	STORY POINTS	ROI (VN/SP)	SPRINT
1	Visitante	Como visitante do site, gostaria de poder vizualiar as postagens publicadas.	Exibir as postagens publicadas.	100	1	100	1
2	Visitante	Como visitante do site, gostaria de posder navegar pelas páginas do site.	Exibir barra de navegação e páginas de conteúdo estático.	100	2	50	1
3	VIP	Como visitante frequente da página, gostaria de poder me cadastrar no site para me tornar um visitante VIP.	Realizar o cadastro do visitante VIP.	80	4	20	1
4	VIP	Como usuário VIP, gostaria de poder vizualizar as postagens para assiantes do site.	Exibir as postagens excluisivas para os usuários VIPs que estejam autenticados no sistema.	80	4	20	2
5	VIP	Como usuário VIP, gostaria de poder publicar comentários nas postagens.	Armazenar e exibir os comentários dos usuários autenticados no sistema.	80	4	20	2
6	Admin	Como administrador do site gostaria de poder, alterar as configurações do site, como título, números de posts por página e a disponibilidade do site.	Armarzenas as informações de configuração do site, exibir mecanismos para manipulação das configurações	100	5	20	2
7	Admin	Como administrador do site, gostaria de gerenciar as publicações de posts.	Gerenciar as publicações dos posts	200	5	40	3
8	Admin	Como adminstrador do site gostaria de poder gerenciar as páginas estáticas do site.	Gerenciar as páginas estáticas do site.	200	5	40	3
9	Admin	Como administrador do site, gostaria de poder gerenciar as categorias das publicações.	Gerenciar as categorias das publicações do site.	200	5	40	3
10	Dev	Como desenvolvedor que utiliza o CMS, gostaria de acessar as informações do CMS no formato JSON para que eu possa integrar	Disponibilizar as informações publicadas em formado JSON por meio de URLs.	300	6	50	4

		as minhas aplicações.					
11	Dev	Como desenvovedor que utiliza o CMS, gostaria de poder integrar os conteúdos publicados aos meus templates de páginas web.	Disponibilizar as informações publicadas em formado JSON por meio de URLs.	300	6	50	4

Sprints backlog:

IBL	USER STORY	CRITERIOS DE ACEITAÇÃO	STATUS (POR FAZER, EM ANDAMENTO, FEITO)	RESPONSÁVEL
1	Como visitante do site, gostaria de poder vizualiar as postagens publicadas.	Exibir as postagens publicadas.	FEITO	[GF]
2	Como visitante do site, gostaria de posder navegar pelas páginas do site.	Exibir barra de navegação e páginas de conteúdo estático.	FEITO	[RR]
3	Como visitante frequente da página, gostaria de poder me cadastrar no site para me tornar um visitante VIP.	Realizar o cadastro do visitante VIP.	FEITO	[RF]

Sprint backlog 1

IBL	USER STORY	CRITERIOS DE ACEITAÇÃO	STATUS (POR FAZER, EM ANDAMENTO, FEITO)	RESPONSÁVEL
4	Como usuário VIP, gostaria de poder vizualizar as postagens para assiantes do site.	Exibir as postagens excluisivas para os usuários VIPs que estejam autenticados no sistema.	FEITO	[GF]
5	Como usuário VIP, gostaria de poder publicar comentários nas postagens.	Armazenar e exibir os comentários dos usuários autenticados no sistema.	FEITO	[RR]
6	Como administrador do site gostaria de poder, alterar as configurações do site, como título, números de posts por página e a disponibilidade do site.	Armarzenas as informações de configuração do site, exibir mecanismos para manipulação das configurações	FEITO	[RR]

Sprint backlog 2

IBL	USER STORY	CRITERIOS DE ACEITAÇÃO	STATUS (POR FAZER, EM ANDAMENTO, FEITO)	RESPONSÁVEL
7	Como administrador do site, gostaria de gerenciar as publicações de posts.	Gerenciar as publicações dos posts	FEITO	[GF]
8	Como adminstrador do site gostaria de	Gerenciar as páginas	FEITO	[RR]

	poder gerenciar as páginas estáticas do site.	estáticas do site.		
9	Como administrador do site, gostaria de poder gerenciar as categorias das publicações.	Gerenciar as categorias das publicações do site.	FEITO	[RF]

Sprint backlog 3

IBL	USER STORY	CRITERIOS DE ACEITAÇÃO	STATUS (POR FAZER, EM ANDAMENTO, FEITO)	RESPONSÁVEL
10	Como desenvolvedor que utiliza o CMS, gostaria de acessar as informações do CMS no formato JSON para que eu possa integrar as minhas aplicações.	Disponibilizar as informações publicadas em formado JSON por meio de URLs.	FEITO	[GF]
11	Como desenvovedor que utiliza o CMS, gostaria de poder integrar os conteúdos publicados aos meus templates de páginas web.	Disponibilizar as informações publicadas em formado JSON por meio de URLs.	FEITO	[RR]

Sprint backlog 4

APÊNDICE D – Questionário Pós-Experimento

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS – PICOS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

QUESTIONÁRIO PÓS-EXPERIMENTO

Ferramenta avaliada: Redmine [ ] SGP [ ]

Avalie os seguintes aspectos da ferramenta, em uma escala de 1 (pouco esforço) a 5 (muito esforço)

Gerenciamento do repositório do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento das informações do repositório do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Gerenciamento dos membros do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento das *branches* criadas no projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento dos *commits* do projeto e os seus responsáveis	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]

Avalie o tempo demandado para cada um dos seguintes aspectos da ferramenta, em uma escala de 1 (pouco) a 5 (muito)

Gerenciamento do repositório do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento das informações do repositório do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Gerenciamento dos membros do projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento das *branches* criadas no projeto	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
Acompanhamento dos *commits* do projeto e os seus responsáveis	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]

Responda as seguintes perguntas, utilizando uma escala de 1 (pouco/baixo) a 5 (muito/alto).

Como você considera a qualidade da ferramenta?	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
O quão conveniente é utilizar a ferramenta?	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
O quão útil você considera a ferramenta?	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
O quanto você recomenda a ferramenta?	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]
O quão satisfeito você ficou ao utilizar esta ferramenta?	1: [ ] 2: [ ] 3: [ ] 4: [ ] 5: [ ]

Sugestões, críticas ou comentários sobre a ferramenta:

_____
_____
_____
_____
_____
_____

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA "JOSÉ ALBANO DE MACEDO"

Identificação do Tipo de Documento

( ) Tese

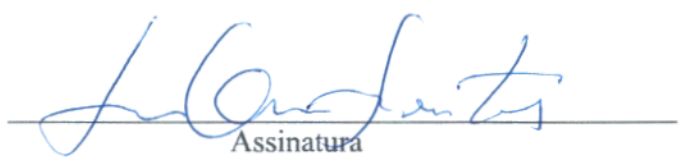
( ) Dissertação

(X) Monografia

( ) Artigo

Eu, Guilherme Dutra Diniz de Freitas, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação "Integração do GitHub com uma Ferramenta de Gerenciamento de Projetos de Software" minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI, 09 de janeiro de 2018.



Assinatura