UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JOSÉ BRENO DE SOUZA

SteinS:

SISTEMA DE GERENCIAMENTO DE REUNIÕES SCRUM

CAMPINA GRANDE - PB

2019

JOSÉ BRENO DE SOUZA

SteinS:

SISTEMA DE GERENCIAMENTO DE REUNIÕES SCRUM

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. Adalberto Cajueiro de Farias.

CAMPINA GRANDE - PB

2019

S729s Souza, José Breno de.
SteinS : sistema de gerenciamento de reuniões Scrum. / José Breno de Souza. – 2019.

8 f.

Orientador: Prof. Dr. Adalberto Cajueiro de Farias.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Gerenciamento de reuniões Scrum. 2. Código GitLab. 3. Scrum. 4. Sprint. 5. Desenvolvimento de código. I. Farias, Adalberto Cajueiro de. II. Título.

CDU:004(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

JOSÉ BRENO DE SOUZA

SteinS:

SISTEMA DE GERENCIAMENTO DE REUNIÕES SCRUM

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. Adalberto Cajueiro de Farias
Orientador – UASC/CEEI/UFCG

Professora Dra. Melina Mongiovi Cunha Lima Sabino
Examinadora – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni
Disciplina TCC– UASC/CEEI/UFCG

Trabalho aprovado em: 02 de julho de 2019.

CAMPINA GRANDE - PB

SteinS: Sistema de gerenciamento de reuniões Scrum

José Breno de Souza[†]

Departamento de Sistemas e Computação , Universidade Federal de Campina Grande, Campina Grande, Paraíba, Brazil, jose.souza@ccc.ufcg.edu.br

RESUMO

A falta de gerenciamento das reuniões Scrum pode impactar diretamente os prazos e os objetivos do desenvolvimento de um produto, sendo necessário uma mudança organizacional para adaptação e controle do processo. Para minimizar esses problemas, foi desenvolvido um sistema que, através do mapeamento dos artefatos e eventos da ferramenta de desenvolvimento de código GitLab e do Scrum, busca minimizar os efeitos da redundância e inconsistência de dados entre o desenvolvimento de código e o processo, e assim garantir a aplicação mais efetiva do framework.

KEYWORDS

Scrum, gitlab, Agile, reuniões

1 Introdução

A dinâmica do mercado competitivo atual de software exige que equipes de desenvolvimento sejam capazes de reagir rapidamente a mudanças em um ambiente que exige mais flexibilidade e comunicação. Nesse cenário, a aplicação de processos convencionais baseados em cascata se torna uma atividade que exige muito tempo e retrabalho, SOMMERVILLE (2011, p. 57), sendo necessário a utilização de metodologias ou processos ágeis.

O Scrum é uma framework voltada para o gerenciamento ágil de projetos, e devido ao seu caráter customizável, dominá-lo não é uma tarefa trivial, sendo fácil de entender, mas difícil de dominar, Schwaber and Sutherland (2017, p. 3). Existem diversas ferramentas, como JIRA, Trello e Tuleap que auxiliam nesse processo, mas que são direcionadas para o gerenciamento de atividades e do tempo, não permitindo a conexão direta entre os artefatos do processo e os artefatos das ferramentas utilizadas no desenvolvimento de software, e nem o gerenciamento das reuniões do processo.

A ferramenta Steins foi desenvolvida com o objetivo de permitir um mapeamento direto entre os artefatos da ferramenta de desenvolvimento de software gitlab e os artefatos do processo ágil Scrum, e assim permitir um melhor gerenciamento das reuniões, e consequentemente uma aplicação mais efetiva no processo de desenvolvimento.

2 Fundamentação teórica

O Scrum consiste em um conjunto de funções, reuniões, artefatos e regras, que unidas formam um processo baseado no método científico empírico. A partir desses componentes, a ferramenta foi construída sustentando-se nos três pilares principais do framework: transparência, inspeção e adaptação.

(i) Transparência é vital para o processo, pois permite que todos os envolvidos acompanhem e compreendam o que está acontecendo na sprint.

(ii) A Inspeção é realizada por todos os membros da equipe, podendo ocorrer no produto, processo ou até mesmo no planejamento, e permite identificar equívocos ou mudanças necessárias.

(iii) A adaptação está relacionada com os resultados da inspeção, onde é necessário avaliar os problemas encontrados durante a sprint e encontrar soluções.

Com base nesses conceitos, parte desses componentes foram mapeados e estruturados para artefatos e eventos do GitLab, permitindo uma maior organização e controle sobre o processo, a partir de uma ferramenta que é familiar aos integrantes do time.

2.1 Product Backlog

As user stories são criadas para refletir as necessidades dos clientes. Eles são priorizados em um Product Backlog, onde são capturadas a urgência e a ordem desejada de desenvolvimento, que são definidas com as partes interessadas, sendo constantemente refinadas durante todo o processo. No GitLab, existem task lists que são geradas dinamicamente.

2.2 User Story

Uma User Story captura um único recurso que agrega valor comercial aos usuários. No GitLab, um única issue dentro de um projeto assume esse propósito.

2.3 Task

Geralmente, uma user story é separada em tarefas individuais, e assim pode-se criar uma lista de tarefas na descrição de uma issue no GitLab.

2.4 Sprint

Um sprint representa um período de tempo no qual o trabalho deve ser concluído, o que pode ser uma semana, algumas semanas, ou talvez um mês ou mais, dependendo da experiência da equipe e das atividades que foram alocadas para a sprint. O recurso de milestones do GitLab suporta isso, atribuindo uma data de início e uma data de vencimento para capturar o período de tempo do sprint, e com base nas prioridades são definidas as issues que serão alocadas.

2.5 Exemplo

João administra uma empresa de desenvolvimento de software, que recentemente foi contratada para o desenvolvimento de um aplicativo de venda de carros. A equipe responsável pelo projeto teve que adotar o processo Scrum, devido às mudanças constantes que eram solicitadas pelo cliente, porém não tinham familiaridade com esse processo. Nas primeiras semanas, os objetivos da sprint não foram atingidos, pois, apesar de usarem uma ferramenta de gerenciamento de atividades, quando o cliente solicitava mudanças, muitas vezes elas não eram atualizadas no gitlab, e os desenvolvedores não notavam as mudanças. Além disso, devido aos prazos curtos, muitas vezes o Scrum master tinha dificuldade de acompanhar o real andamento das atividades, pois a comunicação da equipe não era tão eficiente, e visualizar no gitlab a atividade de cada usuário exige muito tempo.

Nesse cenário, nota-se o quão é difícil administrar uma equipe de desenvolvimento, principalmente quando não estão familiarizados com o Scrum, onde é necessário uma comunicação efetiva entre os membros.

3 Projeto da solução

3.1 Visão geral da Solução

O sistema desenvolvido utiliza os dados de um servidor do gitlab para consumo dos dados, sendo necessário uma conta com token válido no gitlab para autenticação e cadastro. Após realizado o cadastro, os dados do gitlab são mapeados e estruturados diretamente para o SteinS.

3.1.1 Acompanhamento de sprints

Na tela de Sprints é possível visualizar os status de todas as sprints e informações relativas aos prazos e atualizações.

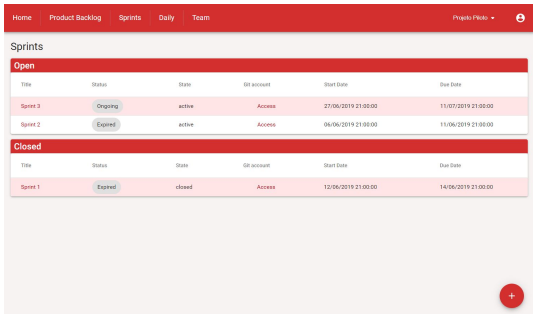

Figura 1: Tela para acompanhamento das sprints do projeto selecionado

Ao selecionar uma sprint, é direcionado para uma tela com seus detalhes, contendo informações das tasks alocadas, juntamente com seus prazos, responsáveis e merge requests associados.

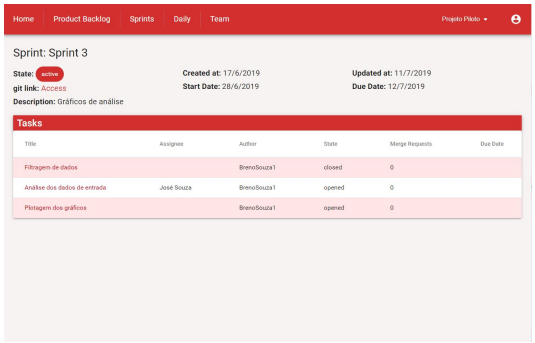

Figura 2: Tela de detalhes de uma sprint

3.1.2 Reuniões diárias

Para o auxílio das reuniões diárias, foram utilizados os dados das atividades dos usuários do time scrum no repositório do gitlab. As informações de atualizações de ações nas

branches, como pode ser observado na Figura 3, e criação e edição de tasks foram selecionadas e agrupadas por usuário.

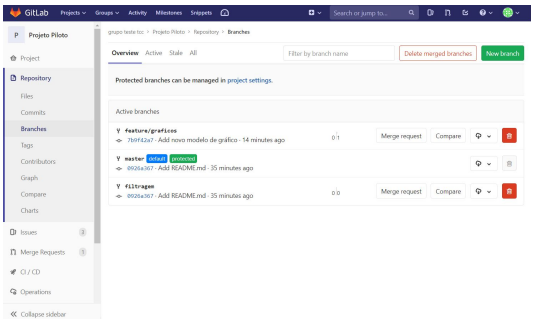

Figura 3: Tela de informações das branches de um projeto do gitlab

Como pode ser visto na figura abaixo, na tela de daily, cada membro adicionado ao projeto do gitlab, apresenta um histórico de atividades realizadas que estão relacionadas ao cumprimento do objetivo da sprint atual. Foram selecionadas as atividades do dia da daily e as do anterior, com objetivo de ajudar a equipe a responder as perguntas:

(i) O que fiz ontem que ajudou o time a atingir a meta do sprint?

(ii) O que vou fazer hoje para ajudar o time a atingir a meta do sprint?

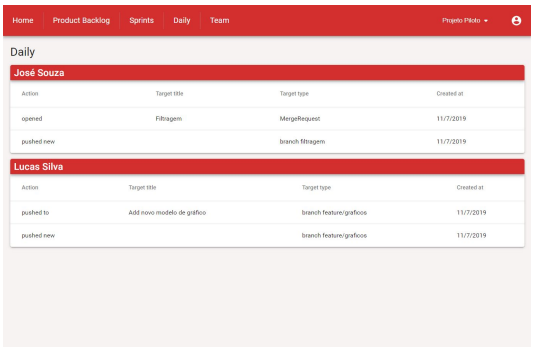

Figura 4: Tela de informações das atividades do time no gitlab

3.2 Arquitetura

A arquitetura definida segue o estilo MVC (Model-View-Controller), onde a aplicação é dividida em camadas, e levando-se em consideração que todos os dados poderão ser consumidos e produzidos através do modelo REST, independentemente da interface gráfica.

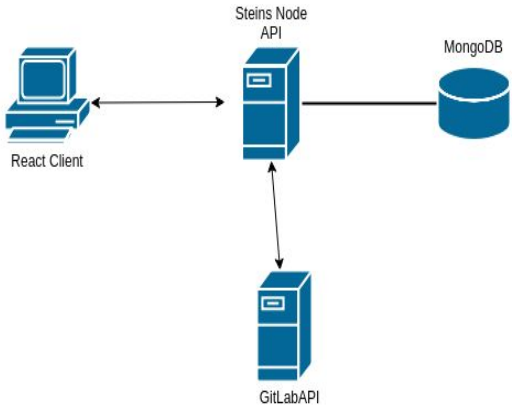

Figura 5: Modelo arquitetural da aplicação

A divisão da aplicação em camadas permite a comunicação com qualquer tipo de aplicação que seja capaz de consumir serviços REST, garantindo dessa forma uma maior flexibilidade para expansão futura.

3.3 Camada de apresentação

A camada de apresentação foi desenvolvida de forma independente das outras camadas, com as bibliotecas React e Redux, que utilizam javascript e permitem alta componentização e consequentemente, reutilização de código. Além disso, para interface do usuário foi utilizado a framework Material-UI.

3.4 Camada de negócios

A camada de negócios utiliza Node.js, juntamente com express, e permite a construção de aplicações escaláveis utilizando javascript. Além disso, ela mantém comunicação direta com a API do GITLAB, que fornece acesso aos dados dos usuários cadastrados no GitLab.

3.5 Camada de persistência

Na camada de persistência foi utilizado o banco de dados MongoDB, que é um banco não relacional orientado a documentos, que utiliza um modelo de dados flexível, de modo que o

schema possa mudar facilmente conforme a aplicação evolui. A biblioteca mongoose também é utilizada, funcionando como um ORM (Object Relational Mapping), traduzindo dados do banco de dados para javascript.

4 Experiência e lições

4.1 Desenvolvimento

O processo de desenvolvimento durou aproximadamente 5 meses, sendo dividido em cinco sprints com duração de aproximadamente um mês.

4.1.1 Sprint 01

Definição da arquitetura do sistema, tecnologias utilizadas no cliente, servidor e banco de dados, e criação dos wireframes das principais telas da aplicação.

4.1.2 Sprint 02

Criação dos repositórios e configuração do ambiente de desenvolvimento e dependências. Desenvolvimento do módulo de login e autenticação por token.

4.1.3 Sprint 03

Construção de formulários de cadastro e login de usuários e integração entre o cliente e servidor com autenticação.

4.1.4 Sprint 04

Estudo da API do GitLab e criação de estrutura para integração do servidor com a API.

4.1.5 Sprint 05

Criação de telas e controller para exibição dos dados do Gitlab.

4.2 Desafios e soluções

O principal desafio no desenvolvimento da aplicação foi construir uma ferramenta que fosse flexível o suficiente para se adaptar às mudanças que são inerentes ao processo SCRUM, e que variam de acordo com a maturidade e o tipo de equipe de desenvolvimento.
Além disso, um dos requisitos essenciais seria a facilidade de utilização, e que exigisse o mínimo trabalho possível para os usuários, tendo em vista que a maioria das soluções existentes são difíceis de utilizar e necessitam muito retrabalho. Com a utilização da API do GitLab, foi possível realizar o mapeamento dos artefatos do desenvolvimento do código para o SCRUM, evitando dessa forma que o usuário criem os artefatos separadamente.

4.3 Trabalhos futuros

Por mais que seja possível identificar e se adaptar a problemas decorrentes do processo, muitos desses problemas se tornam recorrentes e impactam diretamente nos prazos e no objetivo do produto. Identificar a probabilidade de uma sprint atingir seu objetivo dentro do prazo estabelecido com base nos resultados e impedimentos de sprints passadas, poderá auxiliar a tomada de decisões nas reuniões, e assim permitir o melhor aproveitamento do tempo e reduzir os riscos.

REFERENCES

[1]SOMMERVILLE, I. (2011). Software Engineering (9th ed.). Addison-Wesley.

Schwaber, M., Sutherland, J. (2017). “The Definitive Guide to Scrum: The Rules of the Game”.

Larman, C., Vodde, B., (2019). “Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum”. Addison-Wesley.