ST Programming

Camila Carvalho[1], Eduardo Lucas[2], Mailson Dennis [3]

Instituto de Federal de Ciência e Tecnologia da Paraíba
Campus Cajazeiras
Rua José Antônio da Silva, 300 – Bairro jardim Oásis – Cajazeiras-PB

Curso de Análise e Desenvolvimento de Sistemas.

{camilacarwalho3,elucas.crispim,mailssondennis},@gmail.com

Abstract. *This article aims to describe a process thought to guide and facilitate the development and elaboration of software.*

Resumo. *Este artigo tem o objetivo de descrever um processo pensado para guiar e facilitar o desenvolvimento e elaboração de software.*

1. Introdução

O desenvolvimento e a definição de um processo de implementação de projeto é um grande desafio, pois é necessário conhecer com razoável profundidade artefatos e métodos para tal ato. Também é necessário discernimento e criatividade para combinar essas teorias de forma adequada para solucionar os problemas encontrados durante o processo.

Dessa forma, a motivação para a equipe iniciar esse artigo é divulgar a aplicação do nosso processo, evitando que projetos futuros sofram com atrasos, falhas e imprevistos.

O processo definido pela equipe se chama ST Programming, que pode ser usado como base para implementar projetos que exigem um longo período de trabalho, mas que contam apenas com uma equipe pequena (Small Team).

2. Referencial Teórico

O ST Programming usa como base artefatos de Scrum, uma metodologia mais ágil para a gestão e planejamento de projetos. Além disso, houve a inspiração em princípios definidos do Manifesto Ágil, enfatizando a motivação dos indivíduos envolvidos e uma boa comunicação entre os mesmos e o cliente. Por fim, a Extreme Programming (XP) também foi usada como base, já que se trata de uma metodologia para equipes pequenas e médias, que é o caso do processo apresentado nesse trabalho.

De Scrum, nos inspiramos principalmente nos eventos, como o Daily Scrum, uma reunião diária entre membros do time de desenvolvimento. Também há o uso de Sprints, iterações que visam dividir o projeto em partes para serem desenvolvidas de forma mais detalhada e transparente. Na categoria de divisão de equipe, usamos o Scrum Master e Product Owner. Essas duas definições contam com um facilitador responsável por divulgar o framework Scrum e um representante do cliente, respectivamente.

Os artefatos e métodos de XP presentes nesse processo envolvem valores, práticas como User Stories e uma área de trabalho aberta (Open Workspace) e o ritmo sustentável de trabalho. Essas práticas inspiram um trabalho menos cansativo e uma melhor comunicação da equipe de desenvolvimento. Além de facilitar a compreensão do produto solicitado pelo cliente.

Na questão de fases e atividades, nos inspiramos no Processo Unificado de desenvolvimento orientado a objetos. A razão disso é que esse processo facilita a divisão de atividades e análise de requisitos.

3. Papéis

A definição dos papéis numa metodologia é essencial para facilitar o processo como um todo, já que garante que as práticas e os artefatos declarados sejam seguidos com sucesso, criando um sistema de qualidade e no prazo estabelecido. É pensando nisso que o ST Programming exige a formação da equipe por: Product Owner, Scrum Master, e desenvolvedores.

3.1. Product Owner

O papel do Product Owner é representar e priorizar o cliente, garantindo quais requisitos são mais importantes e agregam mais valor, e que devem ser trabalhados nos primeiros momentos do projeto. Consequentemente acaba possuindo uma visão mais geral do produto, o que é uma vantagem, já que ajuda a manter foco no que for mais importante de ser entregue com rapidez. Essa rapidez é importante.

3.2. Scrum Master

Numa maneira generalizada, o Scrum Master é um líder que facilita e prioriza as práticas do Scrum. Seu papel é a divulgação das premissas da metodologia, protegendo o processo de sofrer com problemas que possam comprometer o desenvolvimento. Sua interação com o Product Owner é importante para manter os níveis de importância do que deve ser feito.

3.3. Desenvolvedores

Por último, os desenvolvedores do projeto que formam o time. Seu papel é desenvolver de fato o sistema. Não existe um papel fixo para cada desenvolvedor do tipo, os mesmos podem alternar e revezar papéis para não ficarem presos a sua especialidade. É importante que todos da equipe sejam multifuncionais e se mantenham num ritmo para não comprometer e atrasar o processo.

4. Atividades

Para mapear um desenvolvimento de software, é necessário dividir o processo em etapas essenciais. Pensando nisso, e com inspiração nas fases do Processo Unificado, Scrum e Xp, dividimos as fases em 4.

4.1. Levantamento de requisitos

Primeira etapa e essencial na criação de qualquer sistema. Primeiro há a coleta de requisitos por meio de User Stories, escritas pelo cliente que simulam cenários de casos

de uso. Essa fase é importante para verificar o quão viável é o projeto (econômica, financeira e tecnicamente, por exemplo), assim como destacar fatores mais importantes.

4.2. Projeto

Nessa fase há a especificação dos requisitos coletados na anterior. Aqui é onde o escopo do projeto é pensado e a equipe se prepara para a construção do sistema e são pensadas as primeiras Sprints.

4.3. Construção

Etapa onde o sistema é implementado e acompanhado, é interessante que a equipe se reúna diariamente para acompanhar o processo e o andamento das Sprints. O time é guiado em relação às etapas que devem ser priorizadas e os prazos a serem atendidos, testes, entre outros fatores.

4.4. Transição

Onde o processo é monitorado e garante-se que os requisitos foram atendidos corretamente. É interessante que seja liberada uma versão beta para o cliente. Nessa fase há uma reunião entre a equipe para levantar o que pode ser melhorado para próximos projetos.

5. Artefatos

Um artefato é a representação do que foi produzido durante o desenvolvimento do projeto. São métodos, estratégias ou documentos que facilitam uma metodologia mais transparente e compreensível. No ST Programming, fazemos uso se user stories como documento de requisitos e plano de testes.

User stories descrevem a necessidade do usuário, vista como um requisito. A User Story em si consiste em um Cartão (que pode ser um cartão, índice ou ficha), que prioriza as informações de “Quem” (usuário), “O quê” (necessidade) e “Por quê” (benefício ao usuário). Além do Cartão, a User Story é detalhada por meio de conversas.

O plano de testes se trata de um documento com escrita e abordagem sistemática para os testes do produto. Seu responsável será o desenvolvedor que se encarregar como testador do produto. Sua entrada é a projeção, execução e validação de testes. O que será obtido no final, é o planejamento de iteração e testes futuros.

6. Ciclo de Vida

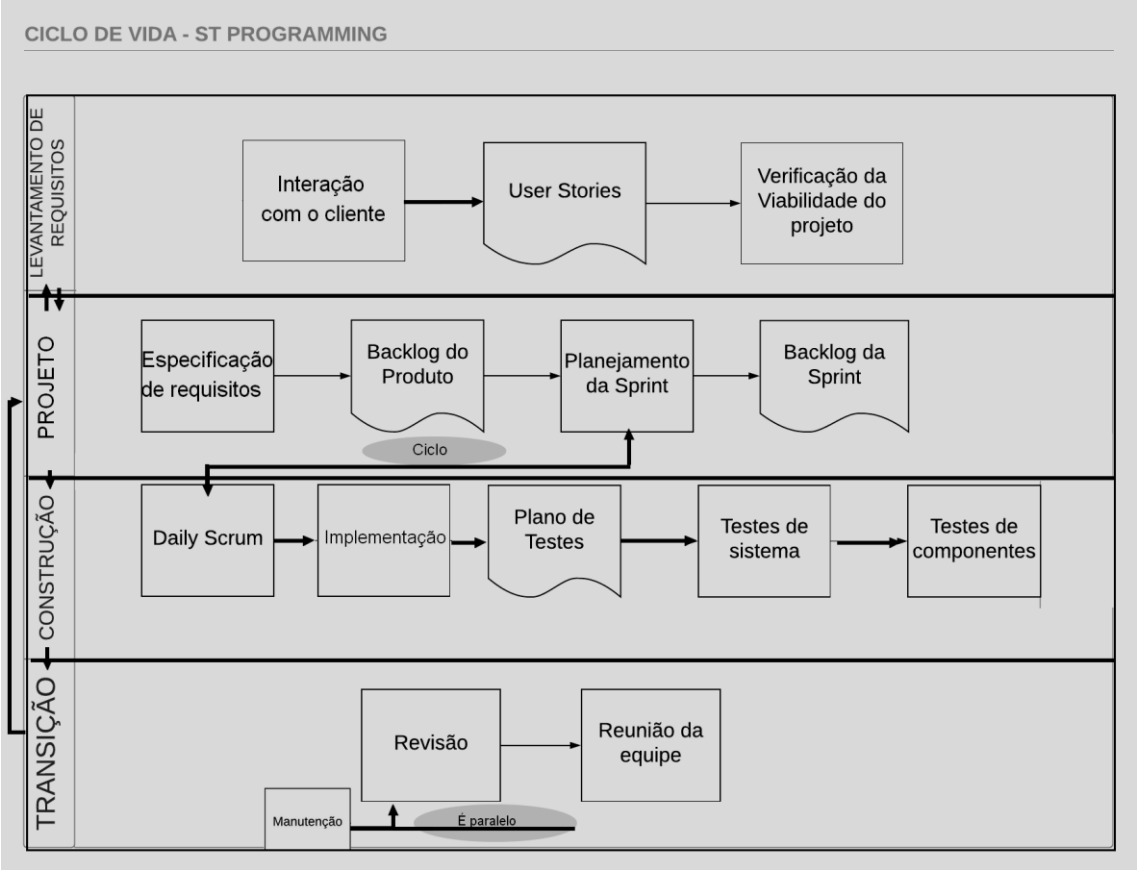

Figura 1. Ciclo de vida do ST Programming

Nosso Ciclo de Vida é Iterativo e Incremental. Quando se trata das Sprints, o ciclo é repetido várias vezes, de acordo com quantas Sprints forem feitas.

7. Indicações e Métricas

As métricas de um processo são usadas como estratégia para adaptação do fluxo de trabalho e atividades técnicas de um projeto. Ocorrem as estimativas, que ajudam na alaviação do processo de software. Nosso processo mede o projeto através da quantidade de user stories, classes programadas e páginas de documentação. Retirando do mesmo, estimativas que melhoram nosso processo, evitando erros e imprevistos que diminuam a eficácia do projeto

8. Ferramentas

Como ferramentas, usaremos o Trello e o Git. Com Trello, divide-se as tarefas entre os membros do grupo e os prazos são organizados. Usaremos o Git para melhor controle de codificação e versões do software, levando assim a melhor organização em geral do projeto. Para a codificação, são usadas as IDE’s.

9. Referências

EXTREME PROGRAMMING. Disponível em: <http://www.desenvolvimentoagil.com.br/xp/>
SCRUM. Disponível em: <https://www.desenvolvimentoagil.com.br/scrum/>
MANIFESTO ÁGIL. Disponível em: <http://www.manifestoagil.com.br/>