UTFPR-UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Bacharelado em Engenharia de Software - 6º Período

DISCIPLINA: *Gerência de Configuração - ES61*
PROFESSOR: *Alexandre L' Erario*

Documento de Processo

Empresa Software Supimpa Tecnologia (2ST)

Caio Vannucci

Cintia Nunes dos Santos

Mariana Felício

Cornélio Procópio
2018

Sumário

1. Introdução

A empresa Software Supimpa Tecnologia (2ST) atua no ramo de software e seu produto de maior destaque é o software Klassic, que muitos investidores apontam ser a maior fonte de captação de recursos da empresa. No entanto, a empresa possui um nicho de clientes com necessidades específicas e o software mencionado precisa ser customizado para atender todos os clientes, além de não possuir uma documentação sólida que sirva de base para novos desenvolvedores.

O objetivo desse trabalho é elaborar e documentar um processo para o gerenciamento de versões, que seja de fácil entendimento para todos os atuais e futuros funcionários da empresa e consequentemente a configuração do ambiente para tal procedimento.

2. Estado atual

Atualmente a empresa conta com 10 desenvolvedores, no entanto esse quadro de funcionários pode sofrer alterações, devido a grande rotatividade do mercado, tornando essencial o desenvolvimento de uma documentação consistente e de fácil entendimento para possíveis novos desenvolvedores que venham a se tornar colaboradores da empresa.

3. Papéis

3.1. Gerente de Projeto

O gerente de projeto é a via de comunicação principal com o cliente. Quaisquer modificações solicitadas pelo cliente será feita por intermédio dele.

Ele possui a responsabilidade de elaboração de documento dos Projetos, além de participar de todas as fases desde a elaboração até a manutenção do produto entregue. Vale frisar que é ele quem delega as funções dentro da equipe.

3.2. Chefe de desenvolvimento

Papel principal dentro do circuito de desenvolvimento, é responsável por criar as branchs e fazer a distribuição de branchs aos desenvolvedores.

Ele também pode participar ativamente do desenvolvimento do produto, além de ser responsável pela validação e documentação do código desenvolvido.

3.3. Desenvolvedor

Tem papel de executar os processos de desenvolvimento, como codificação, manutenção, realizar o controle de versão.

3.4. Tester

Tem o papel de validar o projeto desenvolvido antes que ele seja entregue ao cliente.

4. Requisitos

Para manter o sucesso do software Klassic os seguintes requisitos foram levantados:

- Deve ser possível o lançamento de novas versões do software;
- Novas funcionalidades e otimização devem estar disponíveis a todos os clientes;
- Deve ser possível a customização do software de acordo com a necessidade do cliente;
- Versões anteriores do software devem estar disponíveis aos clientes;
- Todas as alterações no software devem ser documentadas pela equipe;

5. Processo

Para que a empresa obtenha êxito, primeiro devemos estabelecer um processo de software para que os desenvolvedores sigam e também elaborar e documentar um processo que vise gerenciar a versões dos produtos por eles desenvolvidos, uma vez que o mercado de software possui grande rotatividade, logo essa documentação precisa ser consistente e de fácil entendimento para novos funcionários e também para a consulta dos envolvidos no projeto.

Nessa documentação precisa ser armazenado as ferramentas que foram utilizadas para o desenvolvimento do software.

Como processo de software, escolhemos o Scrum, por se tratar de uma metodologia ágil para gestão e planejamento de projetos de software, que pode ser encaixada com facilidade a equipe.

4.1 Processo de Software

O Scrum é uma metodologia ágil para a gestão e planejamento de projetos de software, onde os projetos são divididos em ciclos (2 a 4 semanas geralmente) chamados de Sprints. Nessas Sprints um conjunto de atividades deve ser executado

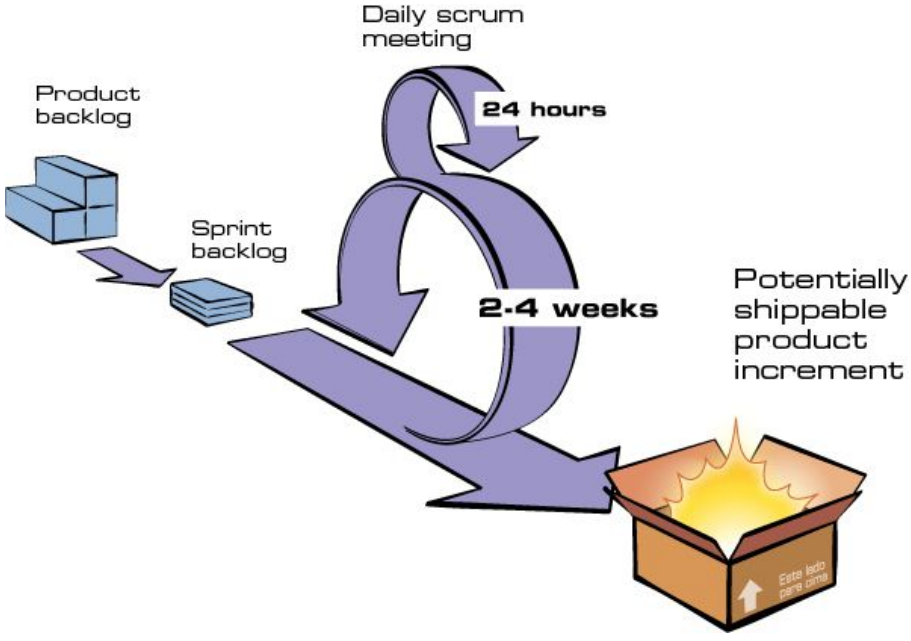
seguindo um Time Box. Ao término desse período uma versão funcional do produto deve ser entregue.

As funcionalidades que serão implementadas no projeto são mantidas em uma lista conhecida como Product Backlog. No início de cada Sprint é realizada uma reunião de planejamento (Sprint Planning Meeting), no qual o Product Owner prioriza os itens do Product Backlog e a equipe (Dev team) seleciona as atividades que será capaz de implementar durante o Sprint que irá se iniciar, essas atividades são transferidas do Product Backlog para o Sprint Backlog.

A cada dia de uma Sprint é realizada uma reunião diária (Daily Scrum), com o objetivo informar o que foi feito no dia anterior, o que será feito no dia atual e identificar possíveis impedimentos para que a tarefa seja cumprida.

Ao final de um Sprint, a equipe deve apresentar as funcionalidades que foram implementadas em uma Sprint Review Meeting e fazer uma Sprint Retrospective com a finalidade de analisar as práticas de adaptação do processo, ou seja, aqui o time irá sentar e verificar o que foi feito de positivo e negativo, e o que deve ser feito nas próximas Sprints e o que não deve ser feito.

Figura 1 - Ilustração dos ciclos do Scrum



Fonte: https://www.desenvolvimentoagil.com.br/images/scrum/ciclo_scrum.gif

4.1.1 Ferramentas

Para monitorar o progresso das atividades de um Sprint sugerimos o uso do Trello, que utiliza os paradigmas do Kanban para o gerenciamento de projetos. Nessa ferramenta todos os envolvidos no desenvolvimento do projeto poderão visualizar o andamento das tarefas.

Os projetos são representados por quadros (boards), que contêm listas de tarefas. As tarefas são representadas por cartões, que são criadas dentro dos quadros. Esses cartões podem ser movidos de uma lista para outra para representar o progresso da tarefa, também é possível adicionar membros aos cartões e listar, bem como comentar o que precisa ser feito em determinado cartão.

4.3 Gerenciamento de Versões

Todas as versões do produto serão armazenados em um repositório para que quando necessário possa-se voltar ao estado anterior ou adicionar novas funcionalidades de acordo com as necessidades dos clientes.

O gerenciamento de versões seguirá o seguinte fluxo:

Correções de Bugs:

1. Seleciona a versão que deve ser aplicada correções.
2. Aplica as correções.
3. Salva uma nova versão no repositório onde estão contidas todas as versões.
4. Valida a versão, para saber se o problema foi corrigido.
5. Caso seja aprovada, essa versão será disponibilizada a todos os clientes, com uma breve descrição do que foi corrigido/adicionado.
6. Caso a nova versão não seja aprovada, deve se voltar ao passo 1 e seguir todo o fluxo de correção até que a nova versão seja aprovada.

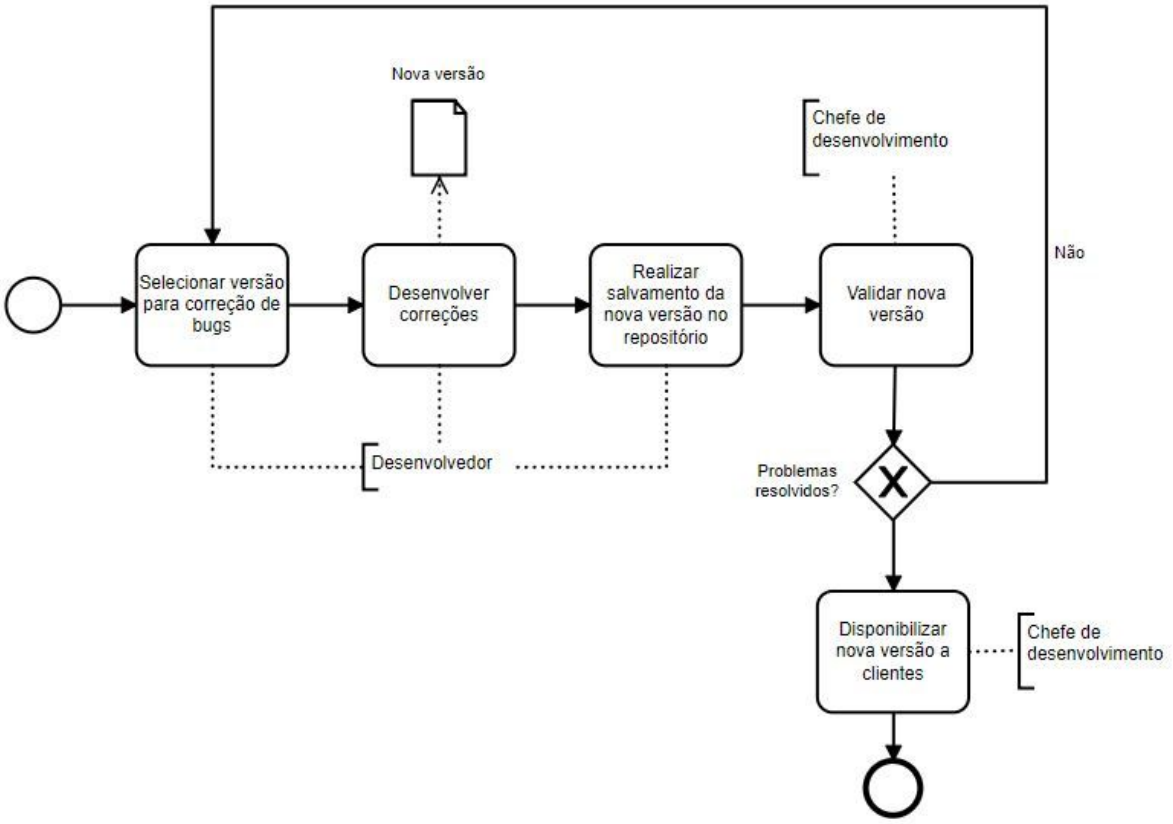


Diagrama BPNM 1 - Fluxo de correção de bugs

Adição de novas funcionalidades:

1. Seleciona versão do produto.
2. Desenvolve funcionalidades requeridas pelo cliente.
3. Adiciona as funcionalidades desenvolvidas.
4. Salva uma versão no repositório em que estão armazenadas as demais versões.
5. Valida a versão com o cliente, para saber se atende sua necessidade e especificação.
6. Caso seja aprovada, publica essa versão para o uso do cliente.
7. Caso não seja aprovada, deve se voltar ao passo 1.

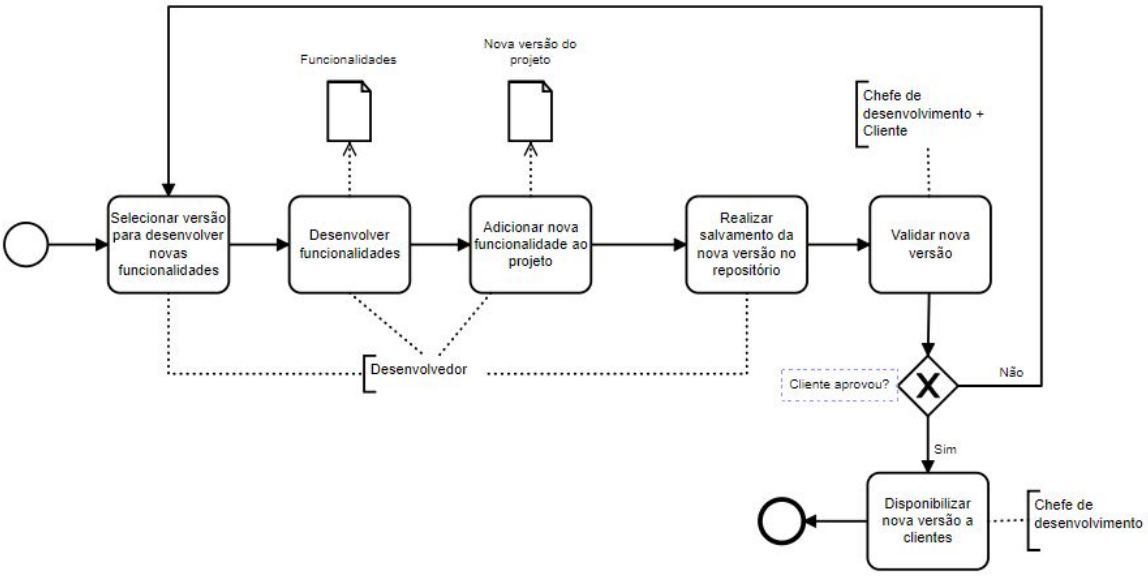


Diagrama BPNM 2 - Fluxo de adição de novas funcionalidades

Remoção funcionalidades:

1. Seleciona versão do produto.
2. Remove funcionalidade pedida pelo cliente.
3. Salva uma versão no repositório em que estão armazenadas as demais versões.
4. Valida a versão com o cliente, para saber se atende sua necessidade e especificação.
5. Caso seja aprovada, publica essa versão para o uso do cliente.
6. Caso não seja aprovada, deve se voltar ao passo 1.

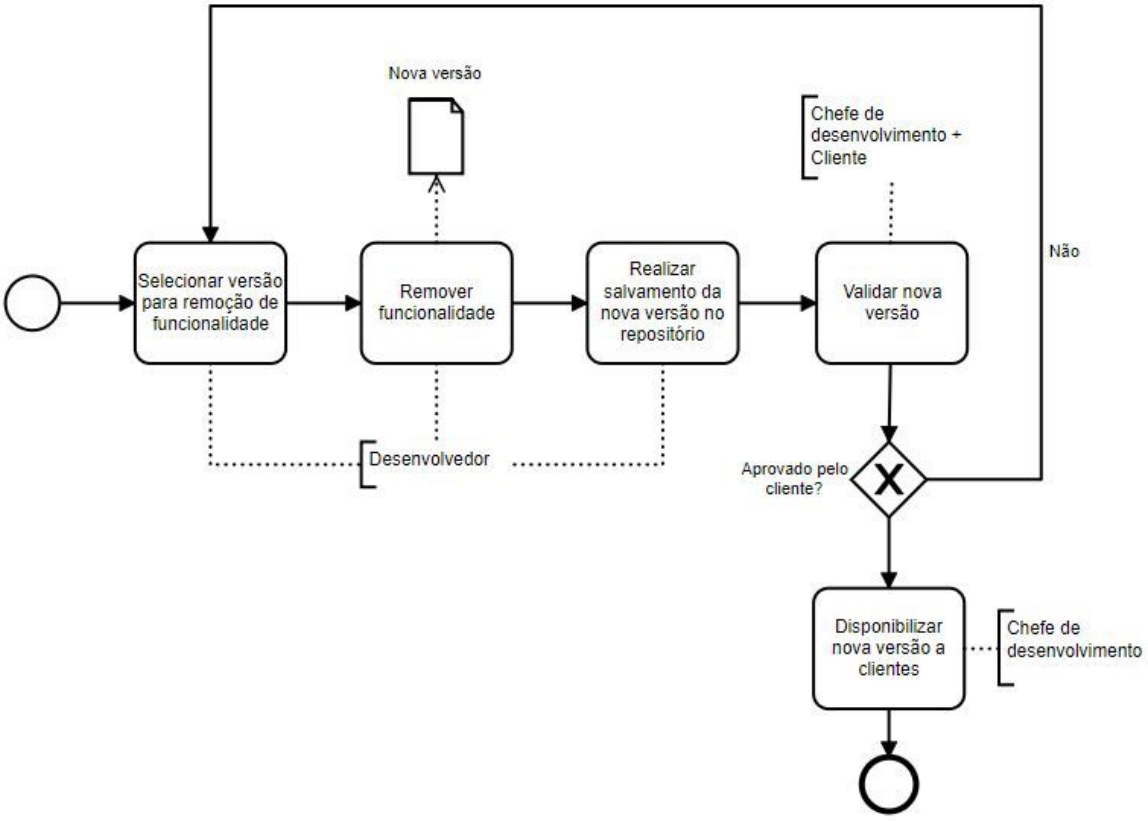


Diagrama BPNM 3 - Fluxo de remoção de funcionalidades

Modificações:

1. Seleciona versão do produto.
2. Modifica o produto de acordo com a especificação do cliente.
3. Salva uma versão no repositório em que estão armazenadas as demais versões.
4. Valida a versão com o cliente, para saber se está atendendo suas necessidades e especificações.
5. Caso seja aprovada, publica essa versão para o uso do cliente.
6. Caso não seja aprovada, deve se voltar ao passo 1.

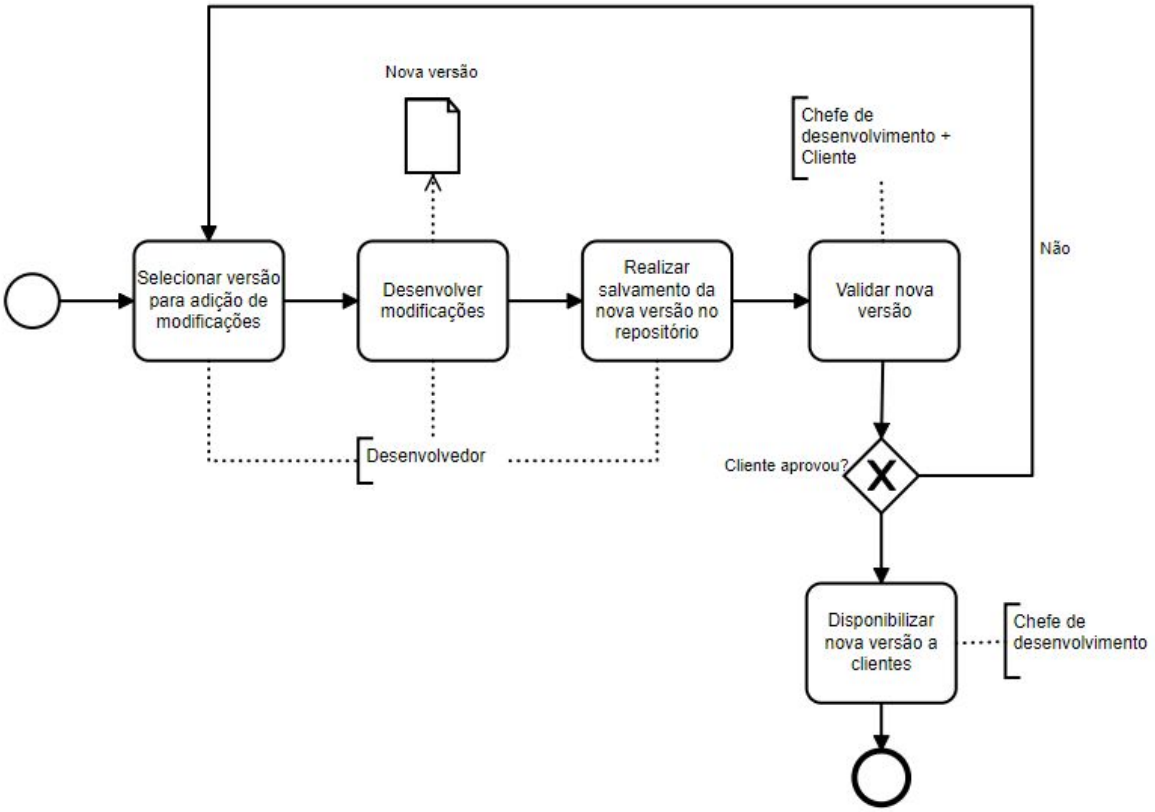


Diagrama BPNM 2 - Fluxo de modificações

Como o software está disponível no site da empresa para cada versão gerada é preciso é preciso ter a descrição das correções implementadas e também novas funcionalidades que foram adicionadas ou removidas, para que novos clientes possam escolher a primeira vista uma versão que melhor se adeque ao seu problema, podendo a partir dessa customizar de acordo com suas necessidades.

6. Riscos

Os riscos do projeto estão no mau gerenciamento do mesmo e consequentemente na não documentação das modificações realizadas e criação de novas versões. Caso isso aconteça pode se acarretar perdas significativas na produtividade e dinheiro para empresa.

7. Manutenção

A proposta da empresa é realizar as manutenções solicitadas pelo cliente de forma que o mantenha funcionando adequadamente.

8. Referências

DANTAS, Cristiane. **Gerência de Configurações de Software.** Disponível em: <https://www.devmedia.com.br/gerencia-de-configuracao-de-software/9145> Acesso em 16/06/2018.

VIEIRA, Denisson. **Scrum - Aprenda Scrum em 9 minutos**. Disponível em: <https://www.youtube.com/watch?v=XfvQWnRgxG0> Acesso em 16/06/2018.

Scrum. Disponível em: <https://www.desenvolvimentoagil.com.br/scrum/> Acesso em 16/06/2018.