Modelagem de Processo

Daniel Yoshizawa
Jacqueline Cardozo
João Pedro Becker Carvalho
Amanda Christoval da Veiga de Aquino

Florianópolis, 2017

1 Departamentos

1.1 Quality Assurance

Setor onde se encontra o Product Owner, P.O., que tem como função analisar as necessidades do mercado e elaborar funcionalidades e cenários que serão repassados a equipe de desenvolvimento como diretivas para o aperfeiçoamento do produto, neste setor também está alocada a equipe de testes que cuidam do teste manual, automatizado, captura e reporte de bugs.

O P.O. pode ser um funcionário da empresa, preferencialmente com experiência na área em que o produto desejado está inserido, porém caso não seja possível pode se contratar alguém de outra área que esteja interessado em assumir este papel, nada impede que até mesmo um desenvolvedor assuma tal papel dependendo da solução, caso se considerem produtos novos e/ou experimentais pode ser melhor utilizar alguém da equipe para validar a ideia com um protótipo antes de alocar mais recursos que podem não valer a pena, isso vai contra algumas definições do SCRUM porém se mostra mais realístico para um cenário como este.

Por se tratar de uma software house muitas vezes o P.O. é o próprio cliente, porém esse não é o cenário ideal uma vez que o tempo de resposta e enquadramento com a realidade da empresa se torna mais complicado, idealmente teríamos um P.O. interno que estaria em contato com este cliente para possíveis alinhamentos.

Dentre as funções do P.O. se encontrar a pesquisa por necessidades do mercado relacionadas ao produto em desenvolvimento, como pesquisas com potenciais clientes, análise de produtos já existentes no mercado e criação de features que possam vir a favorecer o usuário final.

Também cabe ao P.O. o teste das funcionalidades retornadas pelo desenvolvimento para verificar se está tudo de acordo com o que foi pedido e detectar algum bug inicial.

Para o sistema de gerência de atividades, é utilizado o sistema de tickets, sendo este definido na parte X, que será utilizado como principal ferramenta de gerência de tempo e atividades por todas as equipes, assim como definições das milestones, histórico de atividades e como uma das formas de documentação adotadas.

O sistema de tickets é extremamente importante para garantir a maior corretude na troca de informações entre as equipes, o P.O. deve criar tickets para as atividades que deseja que sejam realizadas, assim como eventuais bugs que também devem ser reportados pela equipe de testes, depois deve negociar com o Scrum Master quais tickets vão fazer parte da próxima sprint e uma reunião de pré sprint, também devem ser utilizados para reportar possíveis erros em atividades realizadas, assim se mantém um histórico de mudanças e uma documentação acessíveis por todos como referência para situações parecidas.

1.2 Development Team

As funções da equipe de desenvolvimento são diversas e devem se manter alinhadas com as requisições do P.O., consideração importante considerando que muitas vezes o interesse do desenvolvedor é distinto do que o mercado considera necessário, com pessoas com perfil mais técnico muitas vezes se tem maior foco em soluções para problemas tecnológicos do que voltados para o mercado, por isso é função do Scrum Master e da equipe de desenvolvimento negociar com o P.O. para encontrar um caminho melhor para os dois lados, assim gerando melhorias na infraestrutura do software e até mesmo da empresa enquanto continuam a gera valor para o mercado.

Cabe ao desenvolvedor estimar as horas que levará para determinada atividade, dessa maneira é possível melhorar o planejamento das sprints podendo considerar que atividades similares vão levar o mesmo tempo aproximadamente e também funciona como mecanismo de controle de evolução do desenvolvedor, lembrando que isto nunca deve ser usado como métrica para avaliação de um funcionário, inclusive acesso a essa informação deveria ser negado a qualquer pessoa fora das equipes envolvidas diretamente com dado projeto, removendo assim uma parte da pressão sobre o funcionário para aumentar a produtividade.

O sistema de atribuição de tarefas deve seguir o mecanismo de pull, onde cada desenvolvedor escolhe a atividade que se julga capas e esta com vontade de fazer, jamais utilizar o sistema de push, este pode causar desconforto nas pessoas por se tratar de imposição, claro que existem atividades que ninguém tem interesse em fazer porem são necessárias para isso é necessário conversar com a equipe e chegar a um acordo de quem pode fazer tal atividade, resolvendo no dialogo é na maioria das vezes rápido e não gera problemas para a equipe, por outro lado o sistema de push causa pressão sobre a equipe e gera a falsa impressão de superioridade de membros da equipe, o que não existe, a metodologia adotada aqui é totalmente horizontal em relação aos membros de todas as equipes descritas neste documento.

Todas as técnicas de desenvolvimento podem ser utilizadas, isso fica a cargo da equipe decidir como quer realizar as atividades, utilizar técnicas de eXtreme Programming, XP, como pair programming, code review, TDD, BDD, o que for melhor para a equipe deve ser adotado, não existe obrigatoriedade da criação de testes unitários, black boxes, ou de seguir qualquer técnica de desenvolvimento, este aspecto é totalmente livre, apenas o aberto dialogo e cooperação é incentivado para a melhoria do time como um todo. A justificativa para não definir técnicas obrigatórias é de que cada desenvolvedor deve se sentir confortável com o seu trabalho e ja está sendo adotada uma metodologia um tanto quanto restritiva para o ciclo de desenvolvimento, sabemos dos possíveis problemas que essa abordagem pode gerar, como reinserção de bugs, nivelamento desigual dos membros da equipe, código de baixa qualidade, porem com o tempo e aperfeiçoamento da equipe esses problemas tendem a diminuir e a satisfação do desenvolvedor será maior pois as

escolhas foram dele e este seguiu o caminho que lhe pareceu melhor.

1.3 Test Team

A equipe de testes deve realizar testes manuais, criar e executar scripts de testes automatizados, focando em encontrar falhas nas features implementadas e garantir a integridade do software. Para isso deve seguir roteiros que avaliam bordas, testes de sobrecarga, funcionalidades em cadeias se houver, assim como consistência no design da aplicação.

Quando um erro é detectado pela equipe é necessário analisar a origem do erro e criar um roteiro para poder reproduzir o erro, este deve ser repassado para a equipe de desenvolvimento através de tickets, caso o erro encontrado seja em uma feature conhecida que possui um ticket aberto em validação deve se retornar o ticket com o roteiro para reproduzir o erro, a definição de quando o ticket será retornado a sprint fica a cargo do P.O., Scrum Master e equipe de desenvolvimento, cabe apenas ao tester reportar o erro encontrado e como reproduzi-lo.

Se a detecção de um erro for algum efeito colateral de outras mudanças, ou se o erro encontrado for considerado novo, então é aberto um ticket novo que vai para pré sprint, este deve conter o máximo de informações para facilitar a reprodução do erro pelos envolvidos.

Também deve se rodar os testes automatizados com a maior frequência possível para detectar o quanto antes error que passaram despercebidos pelos testers, assim como garantir a integridade do software de maneira contínua, idealmente um servidor deve rodar os testes automatizados ao menos uma vez por dia, pode se utilizar um batch noturno para isso, caso haja um mecanismo de build contínuo, este só deve criar uma nova versão caso todos os testes estejam passando, aqui deve se considerar o uso de testes unitários caso tenham sido adotados pelo time de desenvolvimento.

Em caso da funcionalidade estar de acordo com o esperado o tester pode fechar o ticket como válido e prosseguir para uma nova tarefa.