UM EXEMPLO DE CHECKLIST PARA SCRUMMASTERS

Michael James
(mj4scrum@gmail.com)
14 de Setembro de 2007
(Revisado em 24 de Julho de 2012)
Traduzido em 19 de Outubro de 2015 por Julio Cesar Fausto
(http://www.jcfausto.com/ - hello@jcfausto.com)

Um Facilitador em Tempo Integral?

Um *Scrum Master* ideal pode trabalhar com dois ou três times ao mesmo tempo. Se você está satisfeito em limitar o seu papel à organização de reuniões, garantir as fatias de tempo (*time-boxes*) e responder aos impedimentos que as pessoas explicitamente reportam, você conseguirá desempenhar este papel em um período de tempo parcial. O time provavelmente excederá a linha base, a expectativa pré-*Scrum* da organização, e provavelmente nada catastrófico ocorrerá.

Entretanto, se você é capaz de imaginar um time satisfeito em realizar tarefas que ninguém antes acreditava ser capaz dentro do contexto de uma organização transformada, considere tornar-se um *grande Scrum Master*.

Um grande Scrum Master pode trabalhar com um time por vez.

Recomendamos um *Scrum Master* dedicado para cada time de aproximadamente sete pessoas, especialmente quando se está começando um novo time.

Se você não descobriu todo o trabalho a ser feito, alinhe-se com o *Product Owner*, com as práticas de engenharia do seu time e com a organização fora do seu time. Enquanto não há uma receita padrão para todos, listei coisas típicas que tenho visto os *Scrum Masters* não darem tanta atenção.

Por favor, marque cada caixa com √, Δ, ?, ou N/A, conforme descrito na última página.

Parte I -- Como Meu *Product Owner* Esta Se Saindo?

Scrum Masters apoiam a melhoria da efetividade do *Product Owner* através de suporte na busca de maneiras para manter o *Backlog* do produto e o plano de versão. (Note que somente o *Product Owner* pode priorizar o *Backlog*.)

☐ O *Backlog* do produto está priorizado de acordo com a última visão do/da *Product Owner*?

☐ Todos os requisitos e desejos de todos os *Stakeholders* estão registrados no *Backlog* do produto? Lembre-se: O *Backlog* é emergente.

☐ O *Backlog* do produto possui um tamanho gerenciável? Para manter um número gerenciável de itens, mantenha as coisas de menor granularidade no topo e Épicos genéricos no final do *Backlog*. É contraprodutivo analisar o *Backlog* do produto muito além dos itens do topo. Os requisitos mudarão durante novas conversações entre a equipe de desenvolvimento do produto e os *Stakeholders/*clientes.

☐ Algum requisito (especialmente aqueles mais próximos ao topo do *Backlog* do produto) pode ser melhor expresso sob a forma de uma História de Usuário independente, negociável, valorada, estimável, pequena e testável[1]?

☐ Você instruiu o *Product Owner* sobre o que é débito técnico e como evitá-lo? Uma peça deste quebra-cabeças pode envolver a criação de testes automatizados e refatorações na direção do conceito de "pronto" para cada item do *Backlog*.

☐ O *Backlog* é um *radiador de informação* imediatamente visível para todos os *Stakeholders*?

☐ Se você estiver utilizando uma ferramenta automatizada para a gestão do *Backlog*, todos sabem facilmente como utilizá-la? Ferramentas automatizadas introduzem o perigo de tornarem-se *refrigeradores de informação* caso não haja radiação ativa pelo *Scrum Master*.

☐ Você pode ajudar a radiar a informação a todos através de informações impressas?

☐ Você pode ajudar a radiar a informação através da criação de gráficos grandes visíveis?

☐ Você tem ajudado o seu *Product Owner* a organizar os itens do *Backlog* em grupos de prioridade ou versões?

☐ Todo mundo sabe se o plano da versão ainda está dentro da realidade? Você deve tentar demonstrar gráficos de *Product/Release Burndown*[2] a todos logo após os itens serem dados como "prontos" durante cada reunião de revisão de *Sprint*. Gráficos demonstram a taxa de itens prontos do *Backlog* do produto e os novos itens adicionados permitindo a descoberta antecipada de desvio de escopo/cronograma.

☐ O *Product Owner* ajustou o plano da versão após a última reunião de revisão da *Sprint*? Uma minoria de *Product Owners, aqueles* que entregam produtos adequadamente testados no prazo, replanejam a versão a cada *Sprint*. Isto provavelmente requererá adiar algum trabalho para versões futuras a medida que itens mais importantes vão sendo descobertos.

Parte II -- Como Meu Time Está Se Saindo?

Embora você seja encorajado a liderar pelo exemplo através da colaboração com os integrantes do time durante seu trabalho, há um risco de que você se envolva mais do que deve em tarefas técnicas. Leve em conta suas principais responsabilidades para com o time.

☐ O seu time está no estado de *flow*? Algumas características deste estado[3]:

- Objetivos claros (expectativas e regras são discerníveis e objetivos são alcançáveis, alinhamento apropriado com o conjunto de habilidades do time).
- Concentração e foco, um nível alto de concentração em um campo limitado de atenção.
- Perda do sentimento de autoconsciência resultado da combinação de ação com atenção.
- *Feedback* imediato e direto (sucessos e falhas no decorrer das atividades são aparentes de forma que comportamentos possam ser ajustados conforme necessário).

---

1 http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/

2 Mike Cohn, Agile Estimation and Planning. (2005).

3 Mihaly Csikszentmihalyi, Flow : The Psychology of Optimal Experience (1990)

- Equilíbrio entre o nível de habilidade e o desafio (a atividade não é muito difícil tampouco muito fácil).
- Senso de controle pessoal sobre a situação ou atividade.
- A atividade é intrinsecamente recompensante de modo que não aparenta requer muito esforço por parte de quem está executando-a.

☐ Os membros do time conhecem uns aos outros, realizam atividades em grupo e celebram os sucessos uns dos outros?

☐ Os membros do time permanecem focados em atingir altos padrões de qualidade e desafiam os demais a crescer?

☐ Há questões/oportunidades que o time não está discutindo por estarem desconfortáveis?[4]

☐ Você tentou formatos e locais diferentes para as reuniões de retrospectiva de *Sprint*?[5]

☐ O time mantém o foco nos objetivos da *Sprint*? Talvez você deva conduzir uma reunião de verificação intermediária (*mid-Sprint*) para refinar os critérios de aceite dos itens do *Backlog* do produto compromissados para a próxima *Sprint*.

☐ O quadro de tarefas da *Sprint* reflete realmente o que o time está fazendo? Cuidado com tarefas escondidas e tarefas maiores do que um dia de trabalho. Tarefas não relacionadas aos compromissos da *Sprint* são impedimentos a estes compromissos.

☐ O time tem de 3 a 9 pessoas com habilidades suficientes para construir um incremento de produto potencialmente entregável?

☐ O quadro de tarefas do time está atualizado?

☐ Os artefatos do time autogerenciável (quadro de tarefas, gráfico de *Burndown* da *Sprint*, impedimentos, listas, etc) estão visíveis e são facilmente manipuláveis pelo time?

☐ Os artefatos acima estão protegidos adequadamente contra intrometidos? Excesso de escrutínio nas atividades diárias por pessoas de fora do time pode impedir a transparência interna do time e a autogestão.

☐ Os membros do time pegam as tarefas de forma voluntária?

☐ A necessidade de pagamento de débito técnico está explícita sob a forma de itens de *Backlog*, fazendo gradualmente do código um local mais agradável para se trabalhar?

☐ Os membros do time estão deixando seus títulos na porta de entrada e se responsabilizando solidariamente com todo o trabalho com o qual o time se comprometeu (testes, documentação para o usuário final, etc)?

---

4 Kerry Patterson, Crucial Conversations: Tools for Talking When Stakes are High (2002). Considere também um facilitador profissional que pode transformar conversar desconfortáveis em confortáveis.

5 Derby/Larson Agile Retrospectives: Making Good Teams Great (2006).

Parte III -- Como Estão As Práticas de Engenharia do Time?

☐ O sistema em desenvolvimento tem um botão “aperte para testar” que permite a qualquer um (mesmo de times diferentes) a convenientemente detectar quando uma falha de regressão foi introduzida (quebra de funcionalidade)? Tipicamente isto é atingido através da utilização de modelos *xUnit* de testes (JUnit, NUnit, etc).

☐ Existe equilíbrio entre os testes automatizados fim a fim (testes funcionais) e testes unitários?

☐ O time está escrevendo tanto testes de sistema quanto testes unitários na mesma linguagem em que o sistema está sendo desenvolvido? A colaboração não é incentivada por *Scripts* proprietários ou ferramentas de (“captura e repetição”) que somente uma parte do time sabe como manter.

☐ O time descobriu a importância da área cinzenta entre os testes de sistema e testes unitários[6]?

☐ Um servidor de integração contínua[7] dispara um alerta quando alguém causa uma falha de regressão? Há como reduzir este ciclo de informação de horas para minutos? (“Construções diárias são para fracos.” – Kent Beck).

☐ Todos os testes são integrados e rodam no servidor de integração continua?

☐ Os integrantes do time descobriram a alegria em se fazer projeto continuo e constantes refatorações[8] como uma alternativa ao *“Big upfront design”*? A refatoração tem uma definição estrita : Mudar a estrutura interna sem mudar o comportamento externo. Refatoração deve ocorrer muitas vezes por hora, toda vez que houver um código duplicado, lógica condicional complexa (visível em excessos de identação ou métodos longos), identificadores mal nomeados, excesso de acoplamento entre objetos, etc. Refatoração com confiança somente é possível com cobertura por testes automatizados. Negligenciar refatorações fará com que seja mais difícil modificar o produto no futuro, especialmente porque será difícil encontrar bons desenvolvedores dispostos a trabalhar em código ruim.

☐ A definição de pronto para cada item do *Backlog* inclui cobertura total por testes automatizados e refatorações? Aprender Desenvolvimento Guiado por Testes (TDD) aumenta a probabilidade de atingir esta condição.

☐ Os integrantes do time estão programando em pares na maior parte do tempo? Programação em par pode aumentar drasticamente a manutenção do código e reduzir as taxas de erro. Ele desafia os limites das pessoas e algumas vezes parece levar mais tempo (quando mensurado em linhas de código em vez de funcionalidades entregáveis). Lidere pelo exemplo começando alguns dias da semana pareando com alguns integrantes do time. Alguns deles preferirão trabalhar desta forma.

Parte IV -- Como A Empresa Está Indo?

☐ A comunicação entre times está ocorrendo em quantidade e forma apropriadas? "*Scrum of Scrums*" é a única maneira de atingir isso, e não necessariamente a melhor.

---

6 http://blog.collab.net/agile/2007/03/07/junit-is-not-just-for-unit-testing-anymore/

7 http://www.martinfowler.com/articles/continuousIntegration.html

8 Martin Fowler, Refactoring: Improving the Design of Existing Code (1999).

- ☐ Os times são independentes o suficiente para produzir funcionalidades, mesmo que tenham que extrapolar algumas fronteiras de arquitetura[9]?
- ☐ Os *Scrum Masters* estão encontrando-se uns com os outros e resolvendo a lista de impedimentos organizacionais?
- ☐ Os impedimentos organizacionais, quando apropriado, são colados na parede do escritório do gerente? O custo pode ser mensurado em valor monetário, perda de tempo de mercado, perda de qualidade ou custo de oportunidade? (Cuidado, aprenda com os erros do Ken Schwaber: "Um *Scrum Master* morto é um *Scrum Master* que não serve para nada" [10])
- ☐ A sua organização possui caminhos de carreira compatíveis com os objetivos coletivos dos seus times? Responda "Não" se há incentivos de carreira[11] para fazer programação ou trabalho de projeto as custas dos testes, testes automatizados ou documentação para o usuário.
- ☐ A sua organização é reconhecida pela mídia ou por órgãos de referência como um grande local para se trabalhar ou um líder em sua indústria?
- ☐ Você trabalha para criar uma *organização que aprende*?

Conclusão

Se você consegui marcar a maioria destes itens como realizados e você ainda tem tempo de sobra durante o dia, eu gostaria de conversar com você.

Não há uma fórmula pronta para criação de engenhosidade humana. Este artigo lista alguns pontos que podem ou não ajudá-lo dentro do seu contexto.

Assim que você entender o que precisa fazer para fazer a diferença, você poderá ficar com medo de fazer. Este é um sinal de que você está no caminho certo.

9 http://FeatureTeamPrimer.org/

10 Ken Schwaber, Agile Project Management with Scrum (2004).

11 Alfie Kohn, Punished By Rewards: The Trouble With Gold Stars, Incentive Plans, A's, Praise, and other Bribes (1999).

Formulário de impedimento organizacional

Problema aparente:

Causa raiz (utilize 5 vezes "por quê?"):

Impactos de negócio:

Impactos emocionais:

solicitação clara e realizável:

- -

Formulário de impedimento organizacional

Problema aparente:

Causa raiz (utilize 5 vezes "por quê?"):

Impactos de negócio:

Impactos emocionais:

solicitação clara e realizável:

FORMULÁRIO DE IMPEDIMENTO ORGANIZACIONAL

PROBLEMA APARENTE:

CAUSA RAIZ (UTILIZE 5 VEZES “POR QUÊ?”):

IMPACTOS DE NEGÓCIO:

IMPACTOS EMOCIONAIS:

SOLICITAÇÃO CLARA E REALIZÁVEL:

- -

Formulário de impedimento organizacional

PROBLEMA APARENTE:

CAUSA RAIZ (UTILIZE 5 VEZES “POR QUÊ?”):

IMPACTOS DE NEGÓCIO:

IMPACTOS EMOCIONAIS:

SOLICITAÇÃO CLARA E REALIZÁVEL:

FORMULÁRIO DE IMPEDIMENTO ORGANIZACIONAL

PROBLEMA APARENTE:

CAUSA RAIZ (UTILIZE 5 VEZES "POR QUÊ?") :

IMPACTOS DE NEGÓCIO:

IMPACTOS EMOCIONAIS:

SOLICITAÇÃO CLARA E REALIZÁVEL:

- -

Formulário de impedimento organizacional

PROBLEMA APARENTE:

CAUSA RAIZ (UTILIZE 5 VEZES "POR QUÊ?"):

IMPACTOS DE NEGÓCIO:

IMPACTOS EMOCIONAIS:

SOLICITAÇÃO CLARA E REALIZÁVEL:

INSTRUÇÕES

Se você recebeu esta lista de checagem como uma tarefa de treinamento e o seu empregador atual (ou o mais recente) estava tentando algo como *Scrum*, por favor aplique isto em relação ao que você viu lá. Marque cada item com uma das seguintes opções:

- √ (para “Fazendo bem”)
- Δ (para “Pode ser melhorado e eu já sei por onde começar”)
- ? (para “Pode ser melhorado, mas como?”)
- N/A (para “Não se aplica” ou “Não trará benefícios”)

Ou se o seu empregador atual (ou o mais recente) não tentou ainda algo como *Scrum,* marque cada item com uma das seguintes opções:

- √ (para “Fazendo bem” ou “Será fácil fazer bem”)
- Δ (para “Será um desafio e eu sei como começar”)
- ? (para “Será um desafio e eu não sei por onde começar”)
- N/A (para “Não se aplica” ou “Não trará benefícios”)

Quanto todos os itens estiverem marcados, descreva de 2 a 6 impedimentos organizacionais nos formulários de impedimento organizacional em anexo, sejam eles não derivados desta lista de checagem. Escolha impedimentos que você tem pelo menos 1% de esperança de vê-los mudando.