Características de uma boa métrica em desenvolvimento de software utilizando Scrum com Kanban

RogerRitter.com.br

rogersmartlife@gmail.com

Resumo. *Métricas de Software é um tema muito discutido em toda comunidade acadêmica e profissional. Sua importância se deve ao desejo de um acompanhamento mais preciso em um projeto de software. Este artigo apresenta algumas métricas de software que podem auxiliar um Scrum Master a atingir este desejo. E por fim, identificar características de uma boa métrica ágil.*

1. Introdução

Os métodos ágeis de desenvolvimento de software promovem um processo empírico, esta forma de se desenvolver exige um ciclo de constante inspeção, adaptação e melhoria. Encontrar maneiras eficazes de se avaliar o processo e uma equipe não é uma tarefa trivial, e muitas vezes leva a uma crença que se cada parte do processo for otimizada, o resultado também será. O que nem sempre é uma verdade.

A preocupação com medidas erradas pode gerar incentivos errados, levando a consequências indesejáveis. GILGUN (1992) diz que as pessoas se comportam de acordo com a forma com que estão sendo medidas: “Diga-me como serei avaliado e eu lhe direi como me comportarei”.

Este artigo apresenta os conceitos relacionados às métricas para auxiliar o Scrum Master de uma equipe ágil, discutindo definições e classificações de diferentes abordagens para a escolha das mais apropriadas métricas, e concluindo com exemplos que podem ser utilizados para identificar algumas características de uma boa métrica ágil.

2. Scrum

Segundo SCHWABER (2013) o Scrum é um *framework* estrutural que está sendo usado para gerenciar o desenvolvimento de produtos complexos desde o início de 1990. Ken Schwaber e Jeff Sutherland que desenvolveram o Scrum, descrevem que o Scrum é um *framework* do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível. Consiste ainda em times associados em papéis, eventos, artefatos e regras no qual cada componente serve a um propósito específico e é essencial para o uso e sucesso do Scrum.

Ainda segundo SCHWABER (2013) o Scrum deixa claro a eficácia relativa das práticas de gerenciamento e desenvolvimento de produtos, de modo que você possa melhorá-las.

2.1 Scrum e Kanban

KNIBERG (2009) descreve o Scrum e o Kanban através de prescrições, sendo a metodologia Scrum definido por:

- Dividir a organização em equipes pequenas, multifuncionais e auto-organizadas.
- Dividir o trabalho em forma de lista de entregáveis pequenos e concretos, classificando-os por prioridade e estimando o esforço relativo de cada item.
- Dividir o tempo e pequenas e curtas iterações de duração fixa, com código potencialmente entregável demonstrado depois de cada iteração.
- Otimizar o plano de entrega e atualizar prioridades em colaboração com o cliente.
- Otimize o processo executando uma retrospectiva depois de cada iteração.

Em resumo, ao invés de um grande grupo gastando muito tempo construindo um grande entregável, em Scrum, se tem uma equipe pequena gastando pouco tempo construindo pequenos entregáveis de forma a integrar regularmente o todo.

Já em Kanban KNIBERG (2009) descreve somente três prescrições, sendo:

- Visualizar o fluxo de trabalho
  - Dividir o trabalho em partes, escrevendo cada item em um cartão e colocando-os na parede.
  - Usar as colunas nomeadas para ilustrar onde cada item está no fluxo de trabalho.
- Limitar o trabalho em progresso (WIP – Work In Progress) limitando explicitamente para quantos itens podem estar em progresso em cada estado do fluxo de trabalho.
- Acompanhar o tempo de execução da tarefa (tempo médio para completar um item), otimizando o processo para tornar o tempo de execução o menor e mais previsível possível.

KNIBERG (2009) define ferramenta como algo com a finalidade de realizar uma tarefa ou atingir um objetivo e um processo sendo a forma de como se realiza determinadas atividades, portanto, Scrum e Kanban se relacionam por serem ambas ferramentas de processo que, em certa medida, auxiliam o processo de maneira mais eficaz, simplificando informações no qual lhe auxiliarão na tomada de decisões.

3. Métrica de Software

SOMMERVILLE (2007) destaca que a medição de software preocupa-se com a derivação de um valor numérico ou o perfil para um atributo de um componente de software, sistema ou processo. Comparando esses valores entre si e com os padrões que se aplicam a toda a organização, pode-se tirar conclusões sobre a qualidade do software ou avaliar a eficácia dos métodos, das ferramentas e dos processos de software.

Não se pode gerenciar o que não se pode medir, Tom de Marco (1982).
Roger S. Pressman (2011) ressalta que se você não sabe para onde você quer ir, qualquer caminho você pode seguir. Se você não sabe onde você está um mapa não irá lhe ajudar.

A diferenciação de Medida, Métrica e Indicador pode ser melhor compreendida quando se descreve cada uma delas segundo o padrão sugerido pelo IEEE (1983) e IEEE (1990).

Segundo a IEEE (1983) a medida é uma avaliação em relação a um padrão. Um exemplo de medida é 5 cm: centímetro é o padrão e 5 é a medida que indica quantos múltiplos ou frações do padrão estão sendo representados. Em desenvolvimento de software, um exemplo de medida pode ser o número de linhas de código. No entanto, não existe um padrão universal para representar linhas de código, pois estes podem variar de acordo com a linguagem e regras utilizadas para o cálculo das linhas.

A métrica por sua vez é um método para determinar se um sistema, componente ou processo possui certo atributo, segundo a IEEE (1983). Ela é geralmente calculada ou composta por duas ou mais medidas. Um exemplo é o número de defeitos encontrados após a implantação: as medidas que compõem essa métrica são o número de defeitos e a fase (ou data) onde o defeito foi identificado.

O indicador é um dispositivo ou variável que pode ser configurado para um determinado estado com base no resultado de um processo ou ocorrência de uma determinada condição. Por exemplo: um semáforo ou uma flag. Conforme a definição do IEEE (1990) um indicador é algo que chama a atenção para uma situação particular.

Segundo PRESSMAN (2011) a medição é uma ferramenta de gerenciamento que se for utilizada adequadamente, ela proporciona discernimento ao gerente de projeto. E, consequentemente, ela ajuda o gerente de projeto e a equipe de software a tomar decisões que levarão a um projeto bem-sucedido. Destaca ainda que as métricas possuem uma determinada entidade no qual se refere em métricas de processo, de projeto ou de produto/software.

Segundo GUNNING (1962) a métrica de produto/software é uma característica de um sistema de software, documentação de sistema ou processo de desenvolvimento que pode ser objetivamente medido. Exemplos de métrica incluem: o tamanho de um produto em linhas de código; o índice Fog.

3.1 Classificações de Métricas de Software

AUER (2001) descreve o papel do tracker como alguém responsável por prover informações para a equipe sobre o processo do time, utilizando as métricas apropriadas para destacar os pontos de melhoria. No Scrum, este papel pode ser do Scrum Master.

As técnicas de medição de software podem ser classificadas segundo diferentes critérios, o Scrum Master precisa considerar algumas das possíveis classificações a fim de atingir o resultado esperado.

3.1.1 Objetiva e Subjetiva

Segundo SATO (2007), o valor de uma métrica objetiva depende somente do objeto em questão e não do ponto de vista do interpretador. Por exemplo o número de classes de

um determinado código ou o número de *deploy* de um determinado executável, pois estes dados são obtidos por uma ferramenta ou fato que realmente ocorreu sem a necessidade de interpretação.

Diferente, a métrica subjetiva trata do objeto em questão e ao ponto de vista do interpretador como por exemplo em uma escala de 0% a 100% definir a complexidade de um código ou o quanto de qualidade ele tem. Apensar da escala definir um valor numérico, a natureza da métrica ainda é subjetiva, pois é dependente do interpretador.

3.1.2 Quantitativa e Qualitativa

A métrica quantitativa, segundo SATO (2007), define que a pertence a um intervalo de uma certa magnitude e geralmente é representada por um número. No qual permite que medidas quantitativas sejam comparadas entre si, como por exemplo, linhas de código ou número de falhas encontradas. Contudo métricas qualitativas são representadas normalmente por palavras, símbolos ou figuras ao invés de números. Um exemplo de métrica qualitativa é o humor da equipe ou a satisfação do cliente, defende GILGUN (1992).

SEAMAN (1999), define que a maioria dos estudos empíricos em Engenharia de Software usam uma combinação entre os métodos quantitativos e qualitativos. Entre as formas mais comuns de combinar ambas estratégias é a codificação, no qual consiste em extrair os valores quantitativos dos dados qualitativos para permitir o tratamento estatístico ou outra abordagem quantitativa. SATO (2007) ressalta ainda que geralmente uma métrica quantitativa é objetiva e uma métrica qualitativa é subjetiva, mas isto não é sempre verdade.

3.1.3 Organizacional e de Acompanhamento

As métricas organizacionais também conhecidas como ‘métricas de avaliação’ são métricas que medem o sistema como um todo, no nível mais alto. Medem o quanto de valor o seu processo consegue entregar no qual sugere uma prática conhecida como Measure Up [Poppendieck (2003)], onde utiliza-se uma medida de avaliação que foge do controle de qualquer sub parte do processo, incentivando a colaboração entre os indivíduos de diferentes equipes e evitando a micro otimização. (Conceito onde somente uma pequena parte (indivíduo) é otimizado/melhorado)

A métrica organizacional ou métrica de avaliação deve ser geralmente escolhidas pelo cliente ou quem quer que espera extrair algum valor do sistema. Como consequência você acaba ficando com um número muito menor de métricas, destaca SATO (2007).

Alguns exemplos de métricas organizacionais:

- Cycle Time: Que mede o tempo médio para que o processo atenda uma nova requisição. O conceito importante é que ela começa a contar com o cliente só termina quando o cliente é atendido.
- Métricas Financeiras: O investimento de projetos de software é justificado com base na melhoria de alguma métrica financeira. Pode ser uma matriz de lucros e perdas ou algo deste sentido, questionando como uma equipe consegue melhorar estes números de outra forma que não seja entregando o software?
- Satisfação do Cliente: Trata-se de uma medida simples que verifica se clientes recomendariam o produto para outro cliente.

As métricas de acompanhamento também conhecida como ‘métricas de diagnóstico’, ‘indicadores’ ou ‘métricas informáticas’. São medidas internas da equipe, segundo SATO (2007), que auxiliam na melhoria do processo através dos ciclos de inspeção e adaptação. Métricas estas que agregam dados não associado a nenhum indivíduo, e também não atreladas ao valor que está sendo entregue. Com um ciclo de vida mais curto, uma vez que o processo evoluiu, elas se tornam desnecessárias e devem ser descartadas. Novas métricas deverão surgir conforme a equipe avança e o ciclo continua em busca da melhoria contínua.

Como exemplo, podemos utilizar o atributo velocidade. Espera-se que após um certo tempo a velocidade de um time s estabilize. Uma vez que a velocidade se torna mais ‘clara’, a equipe geralmente não precisa mais medí-la.

Neste tipo de métrica é comum surgirem diferentes contextos e problemas, podendo-se pensar em diversos outros exemplos. Supondo que a equipe está tentando melhorar a prática de teste, pode utilizar a cobertura de código como métrica de acompanhamento. Se deseja melhorar a integração contínua, mede-se o tempo entre *commits*, ou o número de linhas alteradas em cada *commit*....

3.1.4 Abordagem de Métricas de Software

O processo de medição de software, segundo BARCELLOS (2010) pode ser definido como um conjunto de etapas que deve orientar a realização de medição em uma organização. Um processo de medição eficiente é um fator crítico ao sucesso da medição, pois é ele quem direciona as atividades a serem realizadas para que, com os resultados da análise dos dados coletados, seja possível a identificação de tendências e antecipação aos problemas, afim de prover melhor controle dos custos, redução dos riscos, melhoria da qualidade e, consequentemente, alcance dos objetivos técnicos e de negócio, destaca WANG (2005).

Apesar de propostas existentes possuírem diferenças entre si, percebe-se que, no âmbito do processo de medição as definições propostas consistem basicamente de quatro etapas, descreve BARCELLOS (2010):

- Definição das medidas, definir quais deverão ser as medidas a serem coletadas alinhando-as com o objetivo a ser alcançado.
- Coleta das medidas, coletar as medidas estipuladas nas definições de medidas.
- Análise das medidas coletadas, executar uma análise detalhada com o objetivo de verificar a consistência e a veracidade das medidas coletadas.
- Utilização dos resultados da análise em ações.

3.1.4.1 Técnicas de Medição de Software

Nesta seção são apresentadas algumas abordagens para a escolha de quais métricas utilizar. Um Scrum Master deve utilizar as abordagens apresentadas juntamente com o conhecimento sobre sua equipe e escolher as melhores métricas para a situação que deseja medir.

3.1.4.1.1 Objetivo-Pergunta-Métrica (Goal Question Metric – GQM)

GQM é uma abordagem de cima para baixo (top-down) para estabelecer um sistema de medição direcionado a metas para o desenvolvimento de software, em que a equipe

começa com metas organizacionais, define a medição das metas, levanta questões a abordar os objetivos e identifica as métricas que proporcionem respostas às perguntas.

GQM define um modelo de medição, em três níveis, segundo PEREIRA (2009), conforme ilustrado na figura 1.

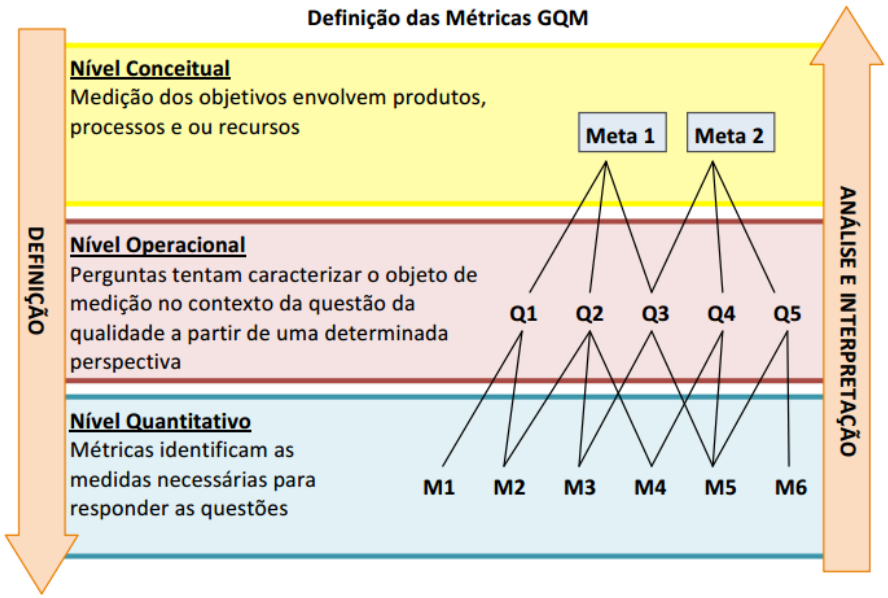

Figura 1. Definição das Métricas GQM.

GQM pode ser aplicado a todo o ciclo de vida de produto, processo e recurso, e está bem alinhada com o ambiente organizacional. A abordagem foi desenvolvida pelo Dr.Victor Basili e colegas durante a década de 1980, em conjunto com o seu trabalho no Laboratório de Engenharia de Software da NASA (SEL). O GQM foi refinado durante a década de 1990 e, agora, serve como uma base para muitas iniciativas de medição. É um meio adequado para alcançar os dados empíricos confiáveis e de conhecimento sobre as práticas de software da organização para se conduzir um processo sistemático de melhoria, conclui PEREIRA (2009).

O principal objetivo da metodologia GQM é fornecer e caracterizar um melhor entendimento dos processos, produtos, recursos e ambiente para estabelecer bases para comparações com trabalhos futuros ou até mesmo com metas (índices) de mercado afirma ANQUETIL (2005).

Embora o foco principal da prática seja definido em métricas direcionadas a metas, também aborda a coleta de dados, análise e interpretação e armazenamento das experiências para uso em futuras iniciativas.
Estas atividades são tão importantes como a definição das métricas porque guia como os dados são realmente utilizados.

Portanto, a abordagem estabelece que primeiramente os objetivos devam ser traçados para que posteriormente sejam definidas suas questões e métricas para avaliação. Sendo assim, pode-se subdividir o processo em três níveis distintos, destaca BASILI (1994):

• Conceitual (Goal): Definição do objetivo (produto, processo ou serviço) para uma variedade de questões, propósito da aplicação do GQM (ex.: melhorar, caracterizar ou controlar), foco de qualidade (características de qualidade), ponto de vista (interessados na aplicação do GQM) e ambiente (contexto para aplicação do resultado).

• Operacional (Question): Definição do conjunto de questões em linguagem natural propostas para caracterizar a forma de avaliação para um objetivo específico definido no nível conceitual.

• Quantitativo (Metric): Definição do conjunto de métricas associadas a cada questão que especificam em termos quantitativos e avaliáveis as informações que se deseja obter durante a avaliação.

3.1.4.1.2 Lean

A metodologia Lean, segundo BASSI (2012) é distribuída nos princípios de eliminar o desperdício, amplificar o aprendizado, adiar comprometimentos e manter a flexibilidade, entregar rápido, tornar a equipe responsável, construir integridade e visualizar o todo. O princípio de eliminar o desperdício foca no sentido de que o desperdício em si pode acontecer em vários sentidos, entre eles: dinheiro, recursos, tempo, esforço e espaço. Cada etapa e atividade realizada no processo devem contribuir para que seja possível construir um produto final com menos custo, mais rapidez e com qualidade.

Ao procurar adaptar-se aos conceitos que funcionaram para a cadeia de suprimentos e no desenvolvimento de produtos, POPPENDIECK (2003), destaca uma abordagem onde se devem escolher as métricas mais apropriadas. A abordagem Lean distingue as métricas organizacionais das métricas de acompanhamento onde um de seus princípios para o desenvolvimento de software é buscado como ‘Otimizar o todo’, medindo um nível acima.

Segundo SATO (2007), a tendência natural para avaliação de desempenho é a decomposição onde o senso comum informa que se as partes de um sistema forem otimistas, quanto a métrica, o sistema também o será. No entanto a micro otimização tende a degradar o resultado final, pois não é possível se medir tudo.

SATO (2007) ainda destaca que em Lean e em algumas outras métricas ágeis de medição, quando a avaliação de um indivíduo é realizada por uma métrica que leva em conta apenas fatores dentro do seu campo de influência, ele tende a trabalhar para otimizar essa métrica, deixando de pensar no sistema como um todo. A prática sugerida pela abordagem Lean é medir sempre em um nível acima.

A abordagem Lean busca ocultar o desempenho individual, pois quando a contagem de defeitos leva em conta o indivíduo que causou o erro, esta passa a ser uma métrica de avaliação, gerando os incentivos equivocados. DEMING (1986), diz que a baixa qualidade nunca é responsabilidade de um indivíduo, mas sim de um processo de gerenciamento que permite que a ausência de defeitos seja impossível.

Além de métricas organizacionais, a abordagem também sugere as métricas de acompanhamento para ajudar a equipe, no qual é importante que seja totalmente dissociada de qualquer avaliação de desempenho. Seu propósito deve ser informacional.

4. Característica de uma Boa Métrica Ágil

Com base em diversas fontes, HARTMAN (2006) propõe uma compilação de algumas das heurísticas que um Scrum Master deve considerar quando estiver escolhendo uma métrica para sua equipe.

- Reforçar princípios ágeis: colaboração com o cliente e entrega de valor são princípios estruturais para os métodos ágeis;
- Medir resultados e não saídas: ao valorizar a Simplicidade, os melhores resultados podem ser aqueles que minimizam a quantidade de trabalho realizado. Medir os resultados obtidos é mais importante que medir as saídas das atividades do processo;
- Seguir tendências e não números: os valores representados por uma métrica são menos importantes que a tendência demonstrada. Aos medir a velocidade da equipe, por exemplo, é melhor se preocupar com sua estabilização do que com o valor absoluto que ela representa;
- Responder uma pergunta específica para uma pessoa real: toda métrica deve expor informação para um ponto de vista específico. Se outra pessoa tem outra pergunta é melhor usar outra métrica;
- Pertencer a um conjunto pequeno de métricas e diagnósticos: é impossível medir tudo. Muita informação pode esconder o que realmente importa. Minimize o número de métricas e meça o que é mais importante;
- Ser facilmente coletada: para métricas de acompanhamento objetivas e quantitativas, o ideal é ter uma coleta automatizada;
- Revelar, ao invés de esconder, seu contexto e suas variáveis: uma métrica deve deixar claro os fatores que a influenciam para evitar manipulações e facilitar a melhoria do processo;
- Incentivar a Comunicação: uma métrica isolada de seu contexto perde o sentido. Um bom sinal é quando as pessoas comentam o que aprenderam com uma métrica;
- Fornecer feedback frequente e regular: para amplificar o aprendizado e acelerar o processo de Melhoria, as métricas devem ser frequentemente atualizadas e disponibilizadas na área de Trabalho Informativa;
- Encorajar um alto nível de Qualidade: o nível de qualidade deve ser definido pelo cliente e não pela equipe. Os métodos ágeis exigem sempre um alto nível de Qualidade do software desenvolvido.

5. Conclusão

As métricas de software, tanto ágil como de desenvolvimento tradicional é de suma importância para se identificar o andamento e a saúde de qualquer projeto, o que também leva a estimativa de software, onde em ambas áreas o conhecimento empírico sobre uma equipe faz a diferença.

No Scrum, o Scrum Master, neste artigo algumas vezes denominado de tracker, é o líder da equipe. E para auxiliar seu gerenciamento pode fazer uso de ferramentas, no qual as métricas de software é uma destas, que, com objetivos e acompanhamentos claros, podem trazer grandes benefícios a uma determinada equipe.

Por fim, o artigo apresentou o objetivo das métricas de software nos métodos ágeis de desenvolvimento e características para identificar uma boa métrica ágil utilizando Scrum e Kanban, segundo HARTMAN (2006).

6. Referências Bibliográficas

ANQUETIL, N.; OLIVEIRA K. M.; SOUZA K. D. Uso do GQM para avaliar implantação de processo de manutenção de software. IV Simpósio Brasileiro de Qualidade de Software, 2005.

AUER, K.; MILLER, R. Extreme Programming Applied: Playing to Win. Addison-Wesley Professional, 2001.

BARCELLOS, P. M. Medição de Software: Um importante pilar da melhoria de processos de software. R. Engenharia de Software Magazine, 2010. Disponível em: https://www.assembla.com/spaces/procsw/documents/bSawKO44Cr37tYeJe5cbLA/download/bSawKO44Cr37tYeJe5cbLA. Acesso em 02 mai. 2014.

BASILI, V.; ROMBACH, H. Goal Question Metric Paradim. Encyclopedia of Software Engineering. New York: John Riley and Sons, 1994.

BASSI, F. D. L., (2012) "Experiências com desenvolvimento Ágil", São Paulo, http://www.teses.usp.br/Dissertacao_Metodos_Ageis_Dairton_Bassi.pdf, Acesso em: 05 jun. 2014.

DEMING, W. E.: Out of The Crisis. Massachusetts Institute of Technology, 1986.

GILGUN, F. J. Definitions, methodologies, and methods in qualitative family research. Qualitative Methods in Family Research. 1992.

GUNNING R. Techniques of Clear Writing. Nova York: McGraw-Hill. 1962.

HARTMANN, D.; DYMOND, R. Appropriate agile measurements: Using metrics and diagnostics to deliver business value. In Agile 2006 Conference, pages 126–131, 2006.

HAZAN, C., (2008) "Análise de Pontos de Função: Uma aplicação nas estimativas de tamanho de Projetos de Software", Engenharia de Software, Rio de Janeiro, p.25-30.

IEEE standard glossary of software engineering terminology, 1983.

IEEE standard glossary of software engineering terminology, 1990.

KNIBERG, H.; SKARIN, M., (2009) "Kanban and Scrum – Making the most of both" Estados Unidos da América. ISBN: 978-0-557-13832-6. http://www.infoq.com/br/minibooks/kanban-scrum-minibook. Acesso em 05 jun. 2014.

PEREIRA, P.; TORREÃO, P.; MARÇAL, A. S. Entendendo Scrum para Gerenciar Projetos de Forma Ágil. Disponível em: http://www.siq.com.br/DOCS/EntendendoScrumparaGerenciarProjetosdeFormaAgil.pdf. Acesso em: 02 set. 2013.

POPPENDIECK, M.; LEAN T.: Software Development: An Agile Toolkit for Software Development Managers. Addison-Wesley Professional, 2003.

PRESSMAN, Roger S.; Arakaki, Reginaldo; Arakaki, Julio; Andrade, Renato Manzan de (Rev.). Engenharia de software: uma abordagem profissional. 7. ed. São Paulo: McGraw Hill, 2011.

SATO, T. D. Uso eficaz de métricas em métodos ágeis de desenvolvimento de software. Instituto de Matemática e Estatística – USP, São Paulo. 2007.

SCHWABER, K. SUTHERLAND, J. (2013) "The Scrum Guide of Scrum.org", https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf, Acesso em: 05 de jun. 2014.

SEAMAN, B. C. Qualitative methods in empirical studies of software engineering. IEEE Transactions on Software Engineering. 1999.

SOMMERVILLE, Ian; Hirama, Kechi (Rev.). Engenharia de software. 8. ed. São Paulo: Pearson Addison Wesley, 2007.

TOM DEMARCO, Controlling Software Projects, Yourdon Press, 1982.

WANG, Q., LI, M., 2005, Measuring and Improving Software Process in China, In Proceedings of International Symposium on Empirical Software Engineering - ISESE 2005, Hoosa Head,Australia, p. 183-192.