ISSN: 2175-1897

Indicadores para teste de software em desenvolvimento ágil

RAQUEL BORTOLUCI
Centro Paula Souza – SP – Brasil
raquelborto@gmail.com

MARCELO DUDUCHI
Centro Paula Souza – SP – Brasil
mduduchi@gmail.com

Resumo: Este artigo tem por objetivo analisar os indicadores de teste de software quando são utilizados métodos ágeis para o desenvolvimento. Observa-se que as métricas, que geram indicadores, não são cobertas diretamente pelos métodos ágeis, mas estas são bastante úteis para melhoria de processo de desenvolvimento de software. Com o trabalho verificou-se quais métricas podem ser facilmente utilizadas e quão úteis elas podem ser para o processo de desenvolvimento e para atender as necessidades dos clientes. A pesquisa verificou a importância da automação de testes na geração das métricas e dos seguintes itens: quantidade de defeitos, casos de testes, funções testadas, testes unitários, cobertura de código e o tempo gasto com testes.

Palavras-chave: teste de software, indicadores, métricas, ágil

Abstract: The goal of this paper is to analyze the software testing indicators when agile methods are used for development. The metrics that generate the indicators are not covered directly by agile methods, but they are very useful for software development improvement. In this study was analyzed which metrics can be easily used in the development process and to attend the customers need. The research verified the automation importance in the metrics generation and the importance of the following items: number of defects, test cases, tested functions, unit testing, code coverage and time spent testing.

Keywords: software testing, indicators, metrics, agile

1. Introdução

Implementar um projeto de TI requer vários estudos, tais como: análise de requisitos, custo, prazo e qualidade. Dentre os fatores relevantes na implantação de sistemas de TI, destaca-se a qualidade do software, que implica na satisfação

do cliente quanto à conformidade, confiabilidade, desempenho e outras dimensões que acercam o software. Observa-se que não apenas a rapidez de produção de software que atendam a demanda do mercado e dos clientes é importante, mas também é necessário estar atento quanto a qualidade do software produzido e garantir a confiabilidade destes softwares, destacando-se, assim, as atividades de testes, que tem por objetivo executar um programa utilizando algumas entradas especificas e verificar se o comportamento está de acordo com o esperado (DELAMARO, MALDONATO e JINO, 2007).

Pesquisas recentes mostram que utilização de métodos ágeis tem aumentado de forma considerável entre as empresas que desenvolvem software. Entre as várias razões para esta adoção destaca-se o tempo de resposta mais acelerado ao mercado, gerenciamento de prioridade de mudança, melhor alinhamento entre TI e o negócio, aumento da produtividade e melhoria na qualidade (VERSIONONE, 2013).

Observando a importância das atividades de testes e o crescimento da utilização de métodos ágeis, o artigo tem por objetivo verificar e analisar quais os indicadores têm sido utilizados em testes de software no desenvolvimento com estes métodos e assim oferecer parâmetros aos desenvolvedores e clientes para auxiliar no acompanhamento do desenvolvimento de software em métodos ágeis.

2. Referencial Teórico

Os métodos de desenvolvimento ágeis utilizam-se de pequenas iterações e com entregas ao final de cada interação. Estes métodos são baseados em princípios que valorizam mais os indivíduos e iterações do que processos e ferramentas, o software funcional mais do que a documentação abrangente, a colaboração do cliente mais do que a negociação de contrato e as respostas às mudanças mais do que a orientação a um planejamento (BECK, K. *et al*, 2001). Dos diversos métodos e frameworks ágeis existentes, os mais reconhecidos são: *Extreme Programming, Scrum, Crystal, Dynamic Systems Development, o Feature Driven Development*. Todos estes métodos seguem os preceitos do manifesto ágil, apesar de apresentar abordagens diferentes (RICO, SAYNO e SONE, 2009).

Uma vez que os indicadores de teste serão avaliados, se torna necessário entender as atividades de teste de software. Define-se como teste de software o processo de execução de um programa com a finalidade de encontrar erros, no qual se aumenta a confiabilidade do software encontrando erros e os corrigindo (MYERS; SANDLER; BADGETT. 2011). Uma das formas de testar um software é executar casos de testes, que são um conjunto de entrada de valores, executadas sob uma pré-condição a fim de alcançar um resultado específico. Durante o teste costuma-se encontrar defeitos que são falhas de um componente ou sistema que podem fazer com que estes deixam de executar uma determinada função (GRAHAM, 2008).

De acordo com Ammann e Offutt (2008) cada atividade de desenvolvimento é acompanhada por um nível diferente de teste, isto é, para cada fase de desenvolvimento, existe uma fase de teste. Na literatura, observa-se com maior ênfase o seguintes níveis: o nível de teste de componente ou unitário tem como

ISSN: 2175-1897

principal objetivo garantir que cada unidade individual do software esteja funcionando; o nível de teste de integração que verifica a interface entre os componentes e iterações com diferentes partes de um sistema; o nível de sistema que verifica o comportamento do sistema como um todo e o nível de aceitação serve para criar confiança no sistema. Ele não tem por objetivo encontrar defeitos, mas demostrar que o software está funcionando de acordo com as especificações (ISTQB, 2011). Dentro do contexto de teste software ágil, é importante destacar a utilização de testes automatizados. Eles são bastante utilizados na implementação de métodos ágeis (CRISPIN e GREGORY, 2009).

Agarwal, Garg e Jain (2014) destacam que os métodos ágeis mudaram a forma de desenvolver o software e por isso é necessário que se investigue novas formas de gerar métricas nestes ambientes que possam orientar gerentes e as atividades de garantia de qualidade.

As atividades de testes são importantes para aferição da qualidade de um software, por isso torna-se bastante relevante a utilização de indicadores que proporcionam informações que auxiliam os gerentes de projetos e engenheiros de software ajustar o projeto, processo ou produto para incluir melhorias. Para criar indicadores deve-se criar medidas, indicando quantitativa a extensão, quantidade, capacidade ou tamanho de algum atributo de um processo ou produto; e assim desenvolver métricas, isto é relacionar as medidas individuais (PRESSMAN, 2011).

Como visto anteriormente para se ter indicadores, é necessário gerar métricas. Vários aspectos podem ser medidos em teste de software para gerar estas métricas, tais como verificadas por Nita (2001):

- Cobertura do planejamento de teste;
- a cobertura da execução de testes;
- a produtividade de planejamento de teste;
- o esforço de planejamento/projeto de teste;
- o esforço de execução de teste;
- a natureza dos defeitos;
- o percentual dos defeitos encontrados para cada natureza de defeitos;
- a quantidade de defeitos por linha de código;
- o defeitos encontrados em produção;
- o esforço de teste unitário;
- retrabalho de implementação e o desvio de esforço no projeto.

Estas métricas são bastante utilizadas em métodos tradicionais de desenvolvimento de software, no entanto, nem sempre se enquadram aos métodos ágeis que tem por objetivo diminuir a quantidade de artefatos de documentação não necessários.

Devido à abordagem ágil ser baseada em constante comunicação, algumas dificuldades são encontradas em aplicar métricas, tais como (UNHELKAR, 2013):

- a falta de foco em parâmetros tradicionais que ajudam a estimar o projeto;
- dificuldade em formalizar requisitos,
- incerteza no planejamento das iterações,
- reuniões rápidas que ajudam a verificar o andamento do projeto mas não geram artefatos para métricas e evolução do projeto,
- estimativa feitas dia a dia não a longo prazo,

- testes contínuos que podem resultar em menos fases dedicadas aos testes e assim não gerar métricas de acompanhamento de desenvolvimento do produto,

- interpretações subjetivas visto que o alinhamento de cada indivíduo com a equipe não fácil de mensurar e diferentes estilos de desenvolvimento ágil,

- visto que cada time pode se organizar e desenvolver o software da maneira que mais adequada para cada time, torna-se difícil padronizar o processo e gerar métricas.

Uma vez que os indicadores são importantes para melhoria de processos e produtos, é necessário que também se captem métricas quando os métodos ágeis são utilizados para desenvolvimento. Observa-se que o manifesto ágil não cobre este assunto e muitos podem dizer que as métricas geradas são apenas documentos desnecessários, mas verifica-se que para se estimar um projeto, verificar a quantidade de pessoas necessárias para cada iteração, distribuir atividades e definir custo faz com que seja necessário a utilização de métricas (CHEMUTURI, 2009).

Hartmann e Dymond (2006) destacam que métricas impróprias podem desperdiçar recursos e distorcer a cultura inerente aos métodos ágeis e por isso apontam que as métricas em desenvolvimento ágil devem afirmar e reforçar os princípios de *lean* e ágil, isto é, elas devem estar focadas nos clientes e sem gerar desperdício. Para os autores, deve-se medir o resultado e não apenas as saídas, isto é, os resultados dos valores entregues ao cliente. Deve-se, também, seguir tendências e não apenas números. As métricas precisam responder à uma questão em particular para uma determinada pessoa, isto é, ela precisa servir para um propósito real. É importante ressaltar que estas métricas devem pertencer à um conjunto pequeno de métricas e diagnósticos, pois o excesso de importação pode esconder tendências importantes. Estas métricas devem ser coletadas de maneira fácil e revelar seu contexto e variáveis para que estas sejam significativas ao time de desenvolvimento, promovendo feedback e possibilitando o aprendizado. Os autores observam também que qualidade deve ser boa suficiente para o cliente ou para negócio.

Baseado nos princípios apresentados acima e nas práticas mais utilizadas em métodos ágeis, Vicente (2010) seleciona uma lista de métricas de teste de software em métodos ágeis, dentre os quais destaca-se:

- a cobertura de código a fim de atingir a cobertura de código desejada é verificada;

- o fator de teste que serve para indicar a relação entre o tamanho do código de testes e tamanho do código em produção com o objetivo de verificar a evolução do código de teste e o esforço para criação de testes;

- assim como a quantidade de casos de teste e assertivas para mensurar a produção;

- a porcentagem de assertivas de teste de unidade passando e falhando;

- a funcionalidade testadas e entregues com o objetivo de medir a quantidade de valor de negócio entregue em cada funcionalidade do sistema, sob o ponto de vista do cliente;

- o tempo de execução de e a quantidade de Defeitos Encontrados.

3. Método

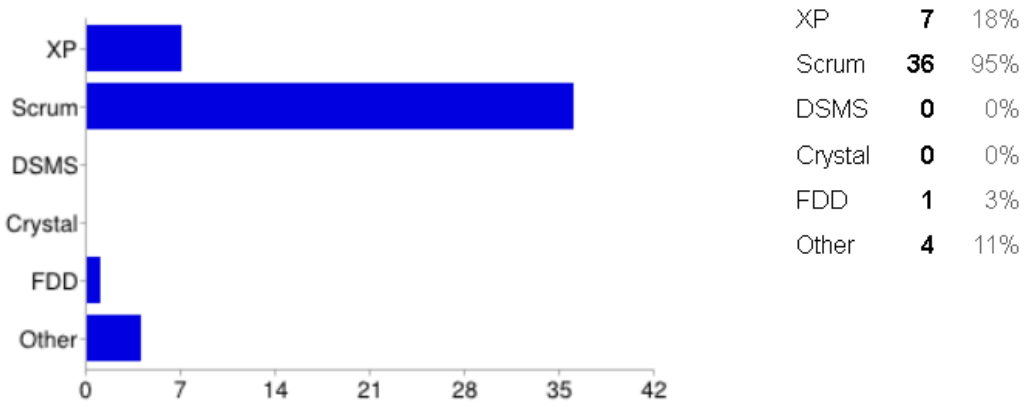
Foi realizada uma pesquisa descritiva com o intuito de verificar quais os principais métodos ágeis utilizados, abordagens de teste e métricas de teste. Uma vez coletados bibliograficamente os assuntos que acercam métricas de teste em métodos ágeis, uma survey foi enviada para comunidades de teste ágil e perfis de redes sociais com o objetivo de alcançar profissionais da área de qualidade de software e assim fazer o levantamento de quais indicadores de teste tem sido mais utilizado em testes de software quando são utilizados métodos ágeis para desenvolvimento.

Um questionário foi aplicado com desenvolvedores e testadores que utilizam métodos ágeis a fim de verificar quais métricas tem sido mais utilizada em ambientes e desenvolvimento ágil e se as equipes de desenvolvimento utilizam os indicadores para melhoria de processo e alinhadas às necessidades dos clientes. A pesquisa contou com 38 respondentes tanto de comunidades de desenvolvimento de software ágil internacional, quanto de comunidade nacional. Participaram da pesquisa pessoas ligadas direta e indiretamente a rede de contatos profissional do pesquisador.

4. Resultados e Discussão

A primeira parte da pesquisa preocupou-se em verificar como o software está sendo desenvolvido pelas pessoas que responderam à pesquisa. Uma vez que os métodos ágeis podem ser utilizados de maneira combinada, o formulário permitiu que as pessoas selecionassem mais de uma opção. Entre os resultados obtidos verificou-se que 95% dos entrevistados utilizam Scrum, 18% utilizam XP (nota-se que todos os participantes que utilizam XP também utilizam scrum), 3% utilizam FDD e 11% selecionaram que utilizam, também, outros métodos. Conforme pode ser verificado na figura 1.

Figura 1 – Métodos utilizados no para desenvolvimento



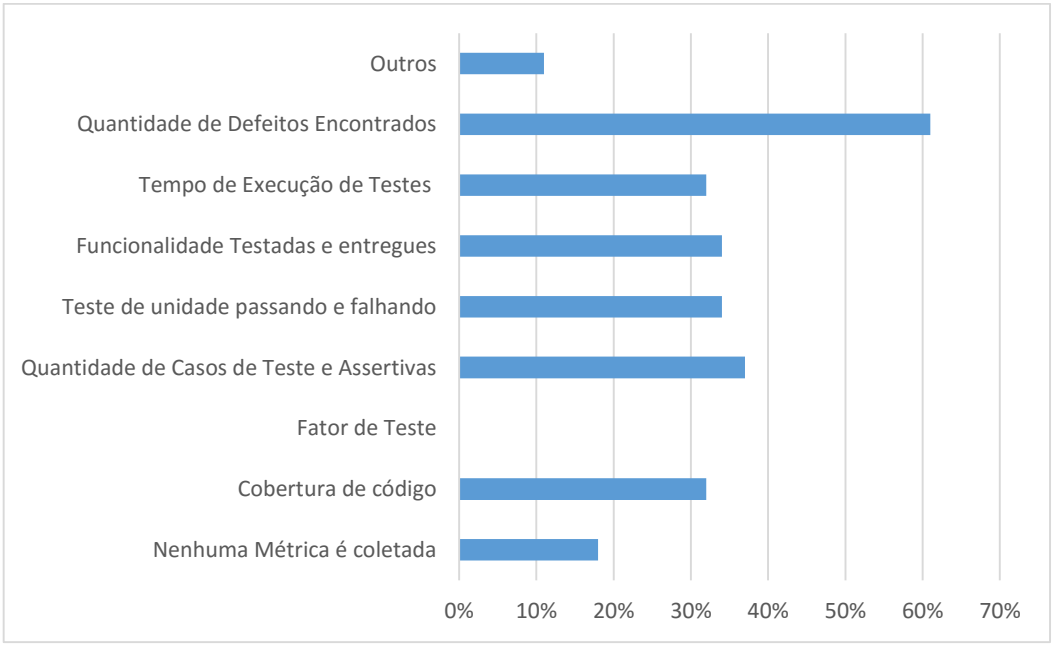
Fonte: Os Autores

Entre os métodos não selecionados na pesquisa bibliográfica os respondentes selecionaram Kanban e TDD. Observa-se que o Kanban é um sistema do *Lean Manufacturing* utilizado para auxiliar na criação do fluxo utilizando-se da visibilidade, no qual os operadores utilizam sinais visuais de fluxo de produção para determinar se uma ação deve ser efetuada. Ele é utilizado no scrum em forma de um quadro de tarefas, funcionando de maneira parecida e dando ao time de desenvolvimento visibilidade do fluxo de trabalho (SABBAG, 2013). Já o TDD (*Test Driven Development*) é um método de desenvolvimento de software no qual utilizando-se testes automáticos, os desenvolvedores escrevem um código do software, executam este teste e quando o teste falha, fazem modificações no software e rodam o teste novamente, até que este passe (BECK, 2013).

Posteriormente foi verificado quais os níveis de testes têm sido utilizados nos testes de software. Nesta pergunta também foi permitido a resposta de várias alternativas. Observou-se uma seleção bastante homogenia, no qual 79% utilizam testes unitário, 82% utilizam testes de integração, 74% utilizam testes de sistema e 79% utilizam testes de aceitação.

Visto o cenário de desenvolvimento e teste, chegou-se ao objetivo principal da pesquisa que foi selecionar quais métricas mais utilizadas nos testes de software. Observou-se que a métrica mais coletada foi a quantidade de defeitos abertos durante o desenvolvimento com 61%, este indicador foi o mais expressivo diante dos outros coletados. Logo após o número de casos de testes e assertivas foi selecionado com 37%, e teste unitários e funções testadas foram igualmente selecionadas com 34%. A cobertura de código e o tempo de teste foram selecionados com 32%, verificou-se, também que nenhum dos respondentes utilizam o fator de teste para as métricas e que 11% utilizam outras métricas, tais como dados de performance, cobertura de requisitos e taxa de testes que passam. Apenas 18% não coletam nenhum tipo de métrica (figura 2).

Figura 2 – Indicadores de Teste



Fonte: Os Autores

Ao se analisar um defeito é possível verificar várias informações, tais como o nível do defeito, a origem, a severidade, facilidade de remoção, complexidade, confiabilidade, manutenção, cronograma, orçamento, portabilidade e conformidade com os requisitos. A satisfação do usuário em relação ao software pode ser medida de várias formas, em relação aos defeitos encontrados em um software, é possível dizer que quanto menos defeitos encontrandos, mais satisfeitos os usuários estarem (JONES, 2008). Além de considerar a satisfação do cliente e também o risco de software falhar em operações que requerem que o software funcione com a precisão, deve-se analisar o custo de um defeito. Verifica-se que quanto antes um defeito é descoberto, mais barato ele é para ser consertado (GRAHAM, 2008). Considerando o risco de um defeito, custo e satisfação de cliente quando não encontra defeitos em sua aplicação, é bastante razoável que este seja um dos indicadores mais utilizados.

Os casos de teste devem ser planejados de acordo com diferentes objetivos de teste. Eles são importantes para verificar se os requisitos do software estão sendo alcançados, além de encontrar defeitos quando executados. Por estes motivos os casos de testes são importantes como indicadores para que se verifique se o objetivo do teste está sendo alcançado, mesmo em métodos ágeis que priorizam mais o software funcionando do que documentação, este tipo de controle tem sido utilizado e pode ser indicador bastante útil para verificar se o software tem atingido a qualidade esperada.

O principal objetivo de medir as funções testadas é verificar a quantidade de valor de negócio entregue em cada funcionalidade do sistema com objetivo de verificar se as necessidades dos clientes foram alcançadas. Em geral, pode-se executar testes de aceitação, fazer demonstrações para o cliente no final de uma iteração e ter sempre uma comunicação com os clientes para atingir este objetivo.

Os testes unitários são bastante importantes para o desenvolvimento de software ágil, pois além de verificar se o componente que acabou de ser desenvolvido está funcionando, eles podem ser automatizados ou até mesmo se utilizar de técnicas como TDD que constroem um teste antes do desenvolvimento do código e assim executar um conjunto de testes quando se desenvolve um novo componente permitindo que o software seja sempre verificado de forma rápida.

Observa-se também que uma das formas de verificar a amplitude do que foi testado é analisar qual a cobertura de código-fonte utilizando-se critérios estruturais e assim atingir a cobertura necessária para a validação do software. Está técnica pode ser bastante útil e utilizada de forma conjunta com os testes unitários.

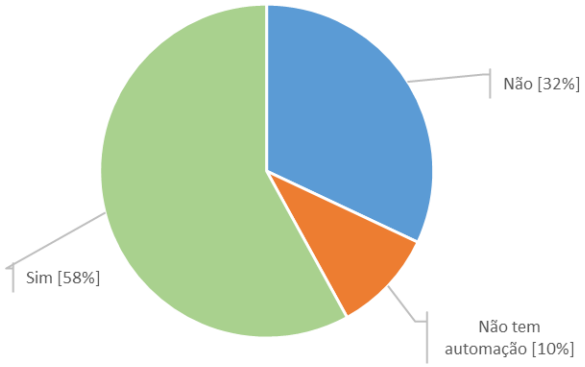
Os métodos ágeis devem ter apenas as disciplinas necessárias para se alcançar as necessidades dos clientes. Fazer um plano de projeto e desenvolver um software com alta qualidade para que se atenda às necessidades de mercado rapidamente com soluções inovadoras (RICO, *et al.*, 2009) Uma vez que os métodos ágeis devem ser eficientes o tempo de teste deve ser verificado a cada iteração para que se possa verificar quanto se tem gasto com esta atividade e assim refletir sobre os processos e melhorias que possam ser feitas no projeto.

O Fator de teste tem por objetivo indicar a relação entre o tamanho do código de testes e tamanho do código em produção. Este indicador não é utilizado por nenhum respondente da pesquisa. Verificou-se, também, que apenas 18% dos

entrevistados não utilizaram nenhum tipo de métrica durante o teste software, enfatizando a ideia inicial de que as métricas são úteis no desenvolvimento ágil.

Após verificar pesquisa bibliográfica que a automação é bastante utilizada nos métodos ágeis e que esta pode gerar dados para métrica, verificou-se com os respondentes se eles tinham a maior parte de seus testes automatizados e 55% deles responderam que grande parte dos testes eram automáticos Já 45% disseram a grande parte não era automatizada. Observa-se que a automação tem sido utilizada como atividade principal dos testes e por isso analisou-se se os dados gerados na automação eram utilizados para criar algum tipo de métrica. 58% dos entrevistados disseram que sim, que elas geravam métricas. 32%, no entanto, responderam que nenhuma métrica era gerada com os dados dos testes e 10% não responderam sobre as métricas gerada em automação, pois não possuíam nenhum tipo de automação (figura 3).

Figura 3 – Análise de métricas de acordo com a geração de métricas úteis no contexto da automação.



Fonte: Os autores

Os testes automatizados ajudam não apenas na melhoria do processo de verificação e validação do software, como também para geração de métricas. Observou-se que a maior parte dos entrevistados as utilizam como indicadores no processo de teste, mostrando, assim, a importância da automação em métodos ágeis.

Uma vez que o objetivo dos métodos ágeis é gerar apenas documentos e dados úteis para criação de software, perguntou-se aos respondentes se os dados gerados nas métricas eram utilizados na aprendizagem e melhoria do processo de desenvolvimento e verificou que 55% dos entrevistados utilizam esta métrica para melhoria do processo, enquanto 45% não as utilizam. Este resultado mostra que a mais de 55% estão alinhados com os valores ágeis gerando métricas úteis e trazendo melhorias ao próprio processo. Observa-se que não houve diferença significativa quando comparado os diferentes métodos utilizados para desenvolvimento.

Perguntou-se, também, se estas métricas estavam alinhadas com as necessidades do cliente e verificou que metade dos respondentes tinham as métricas alinhadas, enquanto a outra metade não tinha. Este dado mostrou que 50% dos desenvolvedores se preocupam em gerar valor na métrica para atender

as necessidades dos clientes. Observa-se que colaboração com os clientes é um dos um dos valores ágeis e por isso gerar métricas neste sentido traz conexão com os princípios ágeis a fim de gerar informações que tragam valor ao cliente.

Não houve diferença significativa na utilização de métodos quando se comparando os níveis de testes utilizados, assim como o método utilizado para desenvolvimento. Destacando, assim, que a forma de produzir o software dentro dos métodos ágeis verificados, não impacta diretamente na geração de indicadores.

5. Considerações Finais

Verificou-se com este artigo que os métodos ágeis enfatizam a colaboração com o cliente e que apenas a documentação necessária deve ser gerada. As métricas não são um assunto discutido diretamente pelos métodos ágeis, mas elas são de grande utilidade para avalição e planejamento do processo de desenvolvimento de software ágil.

Com o intuito de gerar métricas necessárias aos clientes e para melhoria do processo de teste, esta pesquisa analisou as principais métricas utilizadas no teste de software e como estas podem gerar indicadores que sejam uteis tanto para o processo quanto para o cliente sem que exista um esforço de geração de dados que não sejam utilizados.

Destaca-se que a automação de testes é uma atividade que facilita as atividades de teste e que se bem utilizadas podem gerar dados úteis para a geração de métricas, por isso ela é bastante utilizada em métodos ágeis. Observou-se que a automação pode trazer vantagem não apenas nas tarefas de teste software, mas também como fonte de indicadores para o teste de softwrae.

Entre os indicadores mais utilizados, verificou-se a importância da quantidade de defeitos, casos de testes, funções testadas, testes unitários, cobertura de código e o tempo gasto com testes. Por último, destaca-se que a preocupação das métricas é trazer indicadores que tragam valor ao cliente, por isso a necessidade da utilização de métricas que possam gerar indicadores neste sentido.

Referências

.

AGARWAL, A; GARG, N, K; JAIN, A. Quality Assurance for Product Development using Agile. 2014 *International Conference on Reliability, Optimization and Information Technology -ICROIT* 2014, India, 2014.

BECK, K. *Test-driven development: by example*. Boston: Pearson Education, 2003.

BECK, K. *et al*. *Manifesto for Agile Software Development***,** 2001**.** Disponível em: <http://Agilemanifesto.org>. Acesso em: 06/06/2014.

CRISPIN, L; GREGORY J. *Agile Testing: A Practical Guide for Testers and Agile Teams*. 1. ed. Boston: Addison Wesley, 2009.

GRAHAM, D. et al. *Foundations of Software Testing: ISTQB Certification*. 2. ed. London. Cengage Learning. 2008

HARTMANN, D; DYMOND, R. Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value. *Proceedings of Agile 2006 Conference*. Minnesota, USA, Washington, DC, USA: IEEE Computer Society, 2006.

ISTQB. *International Software Testing Qualifications Board. Foundation Level Syllabus*. 2011. Disponível em: <http://www.istqb.org/downloads/viewcategory/16.html> Aceso em: 12/10/2014

JONES, C. *Applied Software Measurement: Global Analysis of Productivity and Quality*. McGraw-Hill/Osborne: 2008

MYERS ,G, J. SANDLER,C. BADGETT, T. *The Art of Software Testing*, 3. ed. New Jersey: Wiley Publishing, 2011.

DELAMARO, M, E; MALDONATO, J, C; JINO, M*. Introdução ao Teste de Software*. Rio de Janeiro: Elsevier, 2007.

NITA, E, F. Melhoria da Qualidade de Produto e de Processo de Software a partir da Análise de Indicadores de Teste**.** *I Simpósio Brasileiro de Qualidade de Software*. 2001.

PRESSMAN, R.S. *Engenharia de Software: Uma abordagem Profissional*. 7. ed. Porto Alegre: AMGH, 2011.

RICO, D., F.; SAYANI H, H.; SONE, S. *The Business Value of Agile Software Methods: Maximizing ROI with Just-in-Time Processes and Documentation.* 1 ed. Fort Lauderdale J. Ross Publishing. 2009.

UNHELKAR, Bhuvan. *The Art of Agile Practice: A Composite Approach for Projects and Organizations*. Auerbach Publications:2013.

VICENTE, A. A. *Definição e gerenciamento de teste no contexto de métodos ágeis*. Dissertação de Mestrado. Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Paulo, SP. Orientador: Márcio Eduardo Delamaro. 2010.