# Requisitos de Metodologias de Teste de software para processos ágeis

**Igor Gonçalves Leal**
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Belo Horizonte - MG - Brasil
E-mail: igor@ideati.com.br

## RESUMO
Em atualização.

## 1. INTRODUÇÃO

Os processos ágeis de desenvolvimento de software se destacam dos processos de desenvolvimento considerados mais tradicionais devido, principalmente, ao fato de priorizarem implementação de funcionalidades através de código executável ao invés de produção de extensa documentação escrita, e ainda, respostas rápidas às mudanças e colaboração com o cliente ao invés de seguir planos rígidos e negociações contratuais [1, pág. 102 a 106]. Os processos ágeis mais populares são SCRUM, Crystal Clear, Adaptive Software Development, Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM) e Extreme Programming (XP).

Esses processos ganharam maior visibilidade a partir de 1996, quando Kent Beck assumiu o controle de um projeto de folha de pagamento da Chrysler denominado Chrysler C3, onde foram utilizadas práticas do XP, um dos mais difundidos processos ágeis. Tanto o XP quando os demais processos ágeis seguem a metodologia iterativa e incremental de desenvolvimento (Iterative and Incremental Development - IID), largamente difundida no mercado a mais de meio século [2]. Uma particularidade relevante dos processos ágeis é de que as iterações levam em média de três a quatro semanas [5].

As principais questões a serem respondidas para avaliação dos processos ágeis são como possibilitar a aplicação desses conceitos sem uma análise abrangente e extensa sobre o que será desenvolvido, e também, como garantir que funcionalidades desenvolvidas sem o amparo de diversos artefatos documentais realmente atendam a requisitos funcionais e não-funcionais, muitas vezes implícitos.

## 2. A NECESSIDADE DE METODOLOGIAS DE TESTE DE SOFTWARE PARA PROCESSOS ÁGEIS

Uma vez que processos ágeis priorizam código executável ao invés de produção de extensa documentação escrita, um ponto essencial para garantia de que exista fidelidade das implementações em relação aos requisitos está nos testes executados sobre o código produzido.

Os processos ágeis mais populares e suas principais características são [7]:

- SCRUM: Forte presença de liderança no projeto e gerenciamento dos requisitos.
- Crystal Clear: Foco na eficiência e habilidade, gestão de configuração, integração freqüente.
- Adaptive Software Development: Adaptação continua do processo, iterativo, tolerante a mudanças.
- Feature Driven Development (FDD): Desenvolvimento dirigido por atributos, com pequenas iterações, posse individual das classes, desenvolvimento dirigido por atributos ou grupo de atributos, construções regulares.
- Dynamic Systems Development Method (DSDM): Baseado no modelo RAD (Rapid Application Development), muito utilizado no desenvolvimento de aplicações focadas em interface com o usuário. Suas principais características são a prototipagem, testes por todo o ciclo da iteração, alterações reversíveis e aplicação de estudos de viabilidade.
- Extreme Programming (XP): Possuiu um conjunto de práticas para desenvolvimento de software, cujas mais diferenciadas são refatoramento, programação em pares, propriedade coletiva, ambiente único com desenvolvedores e clientes, elaboração de histórias (user stories) e ainda desenvolvimento dirigido por testes (Test Driven Development - TDD).

Dentre os processos descritos, o mais próximo ao escopo desse artigo é o XP, principalmente por sua característica de TDD. Conforme mostrado nas seções seguintes.

## 3. REQUISITOS PARA METODOLOGIAS DE TESTE

Os requisitos para metodologias de teste para processos ágeis não possuem grande diferenciação em relação processos guiados por planos. O que realmente difere é o grau de importância conferido aos testes em cada um dos processos.

Nos processos guiados por planos, existe uma série extensa de artefatos documentais resultantes de uma análise profunda sobre o sistema a ser desenvolvido. Os testes são utilizados nesse caso apenas como mais um artefato produzido pelo processo visando garantir a detecção de erros antes da liberação da versão final.

Nos processos ágeis, principalmente o XP, não existe todo esse aparato documental, apenas as histórias contadas pelos usuários [5]. O papel dos testes nesse caso é de suma importância para o sucesso da metodologia ágil, visto que, após os testes, até mesmo as histórias são descartadas.

Constatada essa importância para os testes, é possível listar alguns requisitos para a metodologia de testes para processos ágeis, que são detalhados a seguir.

## 3.1 TESTES MANUAIS OU TESTES AUTOMATIZADOS

Testes manuais consistem no desenvolvedor ou até mesmo o próprio usuário final utilizar o sistema a fim de encontrar anomalias no funcionamento do software. É o tipo de teste menos eficiente, pois dificilmente uma pessoa conseguiria testar exaustivamente um sistema de forma que não restassem possibilidades possíveis para falha. Mesmo constatada tal fragilidade em relação à cobertura, testes manuais ainda utilizados pela facilidade de aplicação, já que basta utilizar o sistema como se estivesse em ambiente de produção, simulando algumas entradas e registrando os resultados obtidos [1, pág. 349].

A automação dos testes consiste na utilização de aplicativos específicos capazes de testar exaustivamente um software através de scripts de teste pré-configurados. Não representa 100% de cobertura, uma vez que algumas falhas lógicas só ocorrem através de combinações específicas de entradas de dados. A utilização de ferramentas de teste é fundamental para o sucesso do projeto, no entanto, a tarefa de definição de scripts de teste eficazes necessita de profissional qualificado. Além de qualificação para definição do domínio dos testes, é necessário ainda domínio da ferramenta utilizada e tempo para configuração de cada teste, assim como análise dos resultados obtidos versus resultados esperados. A combinação desses fatores é o fator complicador para aplicação de testes automatizados [1 , pág. 370]

## 3.2 TEST-FIRST OU TEST-LAST

O processo XP adota a técnica de Test-First, onde a cada nova história de usuário gerada, deve haver uma interpretação desta e em seguida a escrita de um teste, cujo escopo atenda unicamente à nova funcionalidade. O próximo passo é a implementação do código que atenda ao teste recém escrito. Enfim, o teste deve ser adicionado à bateria de testes já existente e esta aplicada ao software final. O novo código implementado só é aceito caso passe por toda a bateria de testes [1]. Observa-se que o teste é escrito antes mesmo da implementação do código pelo desenvolvedor.

Uma possível variação para o processo seria a alteração do momento de escrita do teste, passando essa etapa para depois da implementação do código pelo desenvolvedor (Test-Last), sem alteração no restante do processo [1]. A figura 1 apresenta a diferença entre Test-First e Test-Last.

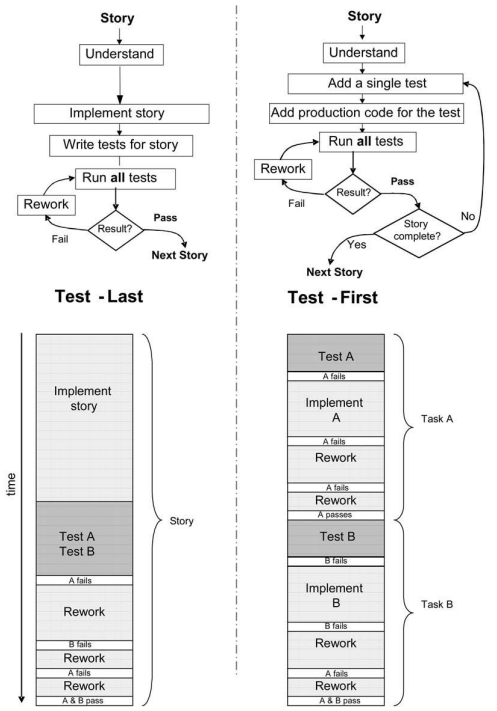


Figura 1: Diferença entre Test-First e Test-Last (extraída de [1])

## 3.3 CRONOGRAMA E RETORNO DO INVESTIMENTO EM RELAÇÃO AOS TESTES

Um fator muito importante a ser avaliado é o tempo consumido pelos testes. Nem sempre é possível destinar o mesmo tempo para testes em todas as funcionalidades do sistema, uma vez que a própria criticidade da funcionalidade pode requerer mais ou menos tempo de testes. O tempo dedicado aos testes deve ser mensurado de forma a não comprometer o cronograma do projeto [3].

Outro fator a ser considerado sobre os testes é com relação ao orçamento. As verbas destinadas ao projeto são finitas, mas a possibilidade de cobertura dos testes não tem fim, ou seja, deve ser definido até que ponto a cobertura de testes ainda é viável ao projeto. O orçamento destinado aos testes deve ser coerente com o tamanho do projeto [3].

## 4. CARACTERÍSTICAS COMPLEMENTARES DOS TESTES

Por se tratarem de processos ágeis, que priorizam adaptabilidade ao invés de seguir planos traçados anteriormente, mudanças de requisitos são sempre bem vindas, o que torna o gerenciamento dos requisitos uma tarefa mais complexa e a possibilidade de falhas mais crítica. Vale ressaltar que o objetivo dos testes é de encontrar erros, e não demonstrar que o sistema funciona da forma como deveria [1, pág. 409].

## 5. METODOLOGIAS DE TESTE DE SOFTWARE

Um importante aspecto a ser observado sobre o TDD é a técnica denominada Test-First. Em um experimento com métodos ágeis, dois grupos de alunos de graduação foram organizados de forma que o primeiro grupo foi designado a escrever os testes funcionais antes das implementações (Test-First), enquanto que o segundo grupo foi designado a escrever os testes funcionais depois das implementações (Test-Last).

Ao final do experimento, foi observado que o grupo que optou pela técnica de Test-First escreveu mais testes do que o outro grupo. Comprovou-se que o número de testes é proporcional à produtividade, ou seja, foi possível concluir que a técnica Test-First proporcionou uma maior produtividade do que Test-Last [1].

Uma vez que os testes utilizados como base nos processos ágeis é que validam a implementação dos requisitos, não só com relação á parte técnica mas também no quesito de funcionalidade [4], o ideal é que o próprio usuário escreva o teste que será aplicado à nova funcionalidade implementada. Essa realidade muitas vezes não é possível devido à falta de conhecimento técnico desse usuário com relação às metodologias para escrita de testes automatizados e com boa cobertura [1, pág. 409].

## 6. CONCLUSÕES

Os processos ágeis têm evoluído ao longo dos anos em maturidade e número de adeptos, principalmente pela filosofia de resultados mais rápidos para o usuário final, uma vez que já existem resultados visíveis logo nos primeiros meses do projeto. Essa característica talvez seja a que mais encanta os clientes, interessados em terem seu sistema em uso o mais cedo possível. Para garantir que erros de implementação não ofusquem essa boa impressão no decorrer do projeto, os testes de software são considerados vitais ao processo.

Algumas características necessárias para aplicação dos processos ágeis nem sempre são possíveis, como disponibilidade de unir em um mesmo espaço físico os desenvolvedores e um representante do cliente que tenha poder de decisão e ao mesmo tempo conhecimento profundo dos requisitos. Além disso, a capacitação técnica e experiência dos desenvolvedores é de extrema importância para o sucesso na aplicação dos processos.

## 7. REFERÊNCIAS


[1] Erdogmus, H. (2005) On the Effectiveness of the Test-First Approach to Programming IEEE Transactions on Software Engineering

[2] Craig Larman, Victor R. Basili (2003) Iterative and Incremental Development: A Brief History – IEEE Computer (pág. 47 a 56)

[3] Cem Kaner (2006) Inefficiency and Ineffectiveness of Software Testing: A Key Problem in Software Engineering Software Engineering at the Florida Institute of Technology

[4] George Wilson (2009) The reality of software testing in an Agile Environment Testing Experience - The Magazine for Professional Testers 03/2009 (pág. 94 a 96)

[5] Teles, Vinicius M. (2005) Um Estudo de Caso da adoção das práticas e valores do Extreme Programming - Dissertação de mestrado Universidade Federal do Rio de Janeiro

[6] Paula Filho, Wilson P. (2009) Engenharia de Software: fundamentos, métodos e padrões 3.ed. - LTC (pág. 102 a 106, 349, 370, 409)

[7] Scott W. Ambler (2008) Agility at Scale: Become as Agile as You Can Be IBM Software Group