Processo de Teste em uma Fábrica de Software Acadêmica: Um Relato de Experiência

Breno Borges França[1], Walquiria Fernandes Marins[2]

[1-2]Bacharelado em Engenharia de Computação – Centro Universitário de Anápolis (UniEVANGÉLICA) – Anápolis - GO

{[1]brenoborgesfranca, [2]wallmarins}@gmail.com

Resumo. *Com surgimento constante de novas tecnologias, sistemas de software passaram a fazer parte de boa parte das tarefas do cotidiano. Nesse sentido, não atender às expectativas do usuário, pode causar desde problemas financeiros até de integridade. A execução de testes se tornou uma atividade crucial para garantir a qualidade e a eficiência do produto final. O objetivo deste artigo é relatar experiências obtidas na elaboração e na evolução de um processo de teste em projetos de uma fábrica de software acadêmica – intitulada Fábrica de Tecnologias Turing (FTT) - para o desenvolvimento de seu primeiro projeto real de nível internacional. Serão apresentados aspectos sobre a aplicação de: i) testes baseados em metodologias híbridas Scrum e OpenUp; ii) elaboração de um processo para nortear as atividades da equipe de analistas de testes com boas práticas. Por fim, os benefícios advindos desse procedimento são evidenciados.*

Palavras-Chave: *Fábrica de Software; Testes; Scrum; Openup.*

1. Introdução

As atividades exercidas nos âmbitos empresariais, acadêmicos e de lazer são cada vez mais facilitadas pelo uso de *softwares* e, desse modo, tornam-se mais integradas e dependentes das tecnologias. Desde computadores pessoais até *smartphones*, são utilizados e isto tem gerado uma grande demanda para desenvolvimento de *softwares*. Este cenário contribuiu para que empresas de Tecnologia da Informação (TI), determinadas em atender as necessidades de seus clientes, procurassem por metodologias ágeis para seu processo de produção e que, além disso, sejam capazes de fornecer métodos e técnicas que contribuam para o aumento da qualidade do produto final.

Segundo Pressman (1995, p. 724), Qualidade de *Software* é a área que trata da conformidade a "requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido". Nesse sentido, de acordo com as diretrizes do SWEBOK (IEEE, 2004), essa área está presente em grande parte dos processos de desenvolvimento e se relaciona tanto com a qualidade do processo de desenvolvimento do *software* quanto com a qualidade do produto desse processo. Quando se trata de Qualidade de *Software* no processo de Engenharia de *Software* os, testes são uma fase determinante, pois visam assegurar que a qualidade esperada pelo cliente seja realmente seja alcançada.

A maioria das pessoas pensam que o Teste de *Software* tem a finalidade de demonstrar o correto funcionamento de um programa quando, na verdade, ele é utilizado como um processo da Engenharia de *Software* para encontrar defeitos. Pocivi (2011, p 20) afirma que o Teste de *Software* consiste na verificação dinâmica do comportamento de um programa por meio de um conjunto finito de casos de testes, selecionados para verificar o comportamento previamente especificado. Com isso, o objetivo principal desta tarefa é encontrar o número máximo de erros dispondo do mínimo de esforço, ou seja, identificar se os resultados estão ou não de acordo com os padrões estabelecidos.

Os projetos desenvolvidos na FTT eram desenvolvidos e corrigidos a partir da percepção dos desenvolvedores e testes aleatórios no módulo funcional do *software*. Deste modo, percebeu-se a ocorrência de muitas inconsistências no produto final causados pelos mais diversos fatores, dentre eles, podem-se evidenciar o conflito entre a especificação de requisitos e sua implementação e problemas causados pela integração de novas funcionalidades. Diante disso verificou-se a necessidade de formalizar e implantar um processo de Teste de *Software* com o intuito de estabelecer um procedimento padrão e identificar problemas com a maior antecedência possível.

Este artigo apresentará a experiência real da implantação e evolução um processo de testes aplicado em uma fábrica de *software* acadêmica, que utiliza uma metodologia hibrida, *Scrum* e o *OpenUp,* no processo de produção de um sistema de gestão acadêmica e financeira para uma universidade. Ao demonstrar os resultados desta experiência, espera-se contribuir para a criação de novos cenários de Testes de *Software* que sejam motivadores e preparados para grandes projetos, garantindo um produto final de qualidade. Espera-se, com isso, incentivar outras equipes a adotar o processo de testes como uma etapa crucial para a garantia de um produto final de qualidade com boas práticas de execução de atividades e consequente redução de custos e esforços.

Além deste capítulo introdutório, este artigo apresenta outros sete capítulos. O Capítulo 2 aborda a relação entre Metodologias Ágeis e as Atividades de Teste. O Capítulo 3 expõe o contexto do Processo de Teste na Fábrica de Software Acadêmica. O Capítulo 4 apresenta o Método de Pesquisa. O Capítulo 5 evidencia os Resultados. O Capítulo 6 aborda as Considerações Finais, por fim, serão apresentadas as referências.

2. Metodologias Ágeis e as Atividades de Teste

Conforme destaca Pocivi (2011, p 23), o surgimento e difusão dos métodos ágeis foi impulsionado pelo acúmulo de documentação e retrabalho em projetos de pequeno e médio porte, gerados pela falta de dinamismo dos processos existentes. Dentre eles, destaca: *Extreme Programming* (XP), *Scrum*, *Crystal*, *Feature Driven Development* (FDD). Tornou-se necessária uma reavaliação do rigor adotado até então para apresentar novas propostas de abordagem, o que culminou no Manifesto do Desenvolvimento Ágil.

Segundo Sommerville (2007), o sucesso dos métodos ágeis estimulou uma integração com métodos de desenvolvimento tradicionais, resultando em métodos de modelagem ágil e em instâncias do *Rational Unified Process* (RUP).

O *Scrum* foi proposto pela IBM, reflete as boas práticas contidas no RUP e pode ser extensível e otimizado para as necessidades de cada projeto ou ambiente. Se apoia em três pilares fundamentais: i) a **transparência**, que defende que todos os responsáveis pelo resultado devem ter a visão de tudo o que está acontecendo, além de um mesmo entendimento do que está sendo visto; ii) a **inspeção** que define que os artefatos e o progresso em direção ao objetivo devem ser inspecionados frequentemente por todos os usuários do *Scrum*, de maneira a detectar desvios indesejáveis; e iii) a **adaptação** que assume que as coisas mudam e prega a adaptação a mudanças no lugar de tentar evitá-las.

O *OpenUp* (IBM, 2011, *online*) se apresenta como método de desenvolvimento ágil com abordagem iterativo e incremental, baseado no RUP e no *Scrum*, e que valoriza a colaboração entre os integrantes da equipe, bem como busca evitar entregáveis desnecessários. É caracterizado por quatro princípios básicos: i) **colaborar** para alinhar interesses e compartilhar entendimento, balancear as prioridades (necessidades e custos técnicos) para maximizar o valor dos interessados; ii) **enfocar na articulação da arquitetura** para facilitar a colaboração técnica; iii) **reduzir riscos**; e iv) minimizar o sucateamento e o retrabalho e **evoluir continuamente** para demonstrar resultados, e ganhar *feedback* do cliente.

O *OpenUP* possui um conjunto de papéis, tarefas e artefatos envolvidos no desenvolvimento de *software*. O papel define o comportamento e as responsabilidades de indivíduos que trabalham em equipe. A tarefa é uma unidade de trabalho no qual um papel pode ser solicitado a executar. O artefato é um produto formal de trabalho que é produzido, modificado ou usado por uma tarefa; define uma área de responsabilidade; e está sujeito ao controle de versão (IBM, 2011; JUBILEU, 2008; BALDUÍNO, 2011).

2.1. Documentos da Norma IEEE-829

A Norma IEEE-829, também conhecida como padrão para documentação de teste de *software*, relaciona e descreve documentos e respectivos formatos para a realização de oito estágios de tarefas da equipe de teste que incluem planejamento, especificação e o relato de testes. Apesar disso, não recomenda quantos e quais documentos devem ser produzidos. Os documentos são:

- Plano de Teste: contém todo o planejamento para execução dos testes, contendo os itens e funcionalidades a serem testados, as tarefas a serem realizadas, os riscos associados com a atividade de teste até o cronograma das atividades de teste;
- Especificação de Projeto de Teste: especifica o que é apresentado no Plano de Teste que descreve as funcionalidades e características a serem testadas pelo projeto e qualquer outro teste associado;
- Especificação de Caso de Teste: especifica os passos dos casos de teste, contendo dados de entrada, resultados esperados, ações e regras de negócios gerais para a execução do teste;
- Especificação de Procedimento de Teste: expõe os passos para executar um conjunto de casos de teste;
- Diário de Teste: apresenta registros do cronograma dos detalhes relevantes relacionados à execução dos testes;
- Relatório de Incidente de Teste: documenta qualquer evento que ocorra durante a atividade de teste e que requeira análise posterior;
- Relatório/Resumo de Teste: demonstra de maneira resumida os resultados das atividades de teste associados a uma ou mais especificações de projeto de teste e provê avaliações baseadas nesses resultados;
- Relatório de Encaminhamento de Item de Teste: aponta os itens enviados para teste no caso de equipes separadas serem responsáveis pelas tarefas de desenvolvimento e de teste.

As técnicas de teste de *software* foram criadas por volta dos anos 70, entretanto, as empresas ainda têm uma grande dificuldade com essa atividade. Tal cenário é um reflexo da falta de profissionais especializados e a grande dificuldade em implantar um processo de teste utilizando as técnicas existentes nas normas já criadas.

3. O Processo de Testes na Fábrica de Software Acadêmica

O papel da equipe de testes e o conjunto de atividades relacionadas foram definidos de forma que atendam à norma IEEE-829 e a metodologia hibrida S*crum* e *OpenUP*. Os testes realizados na FTT compreendem: Teste de Documentação, Teste Funcional, Teste de Compatibilidade e Teste de Regressão, sendo que os três últimos são apoiados por ferramentas de automação.

3.1. Contexto da FTT

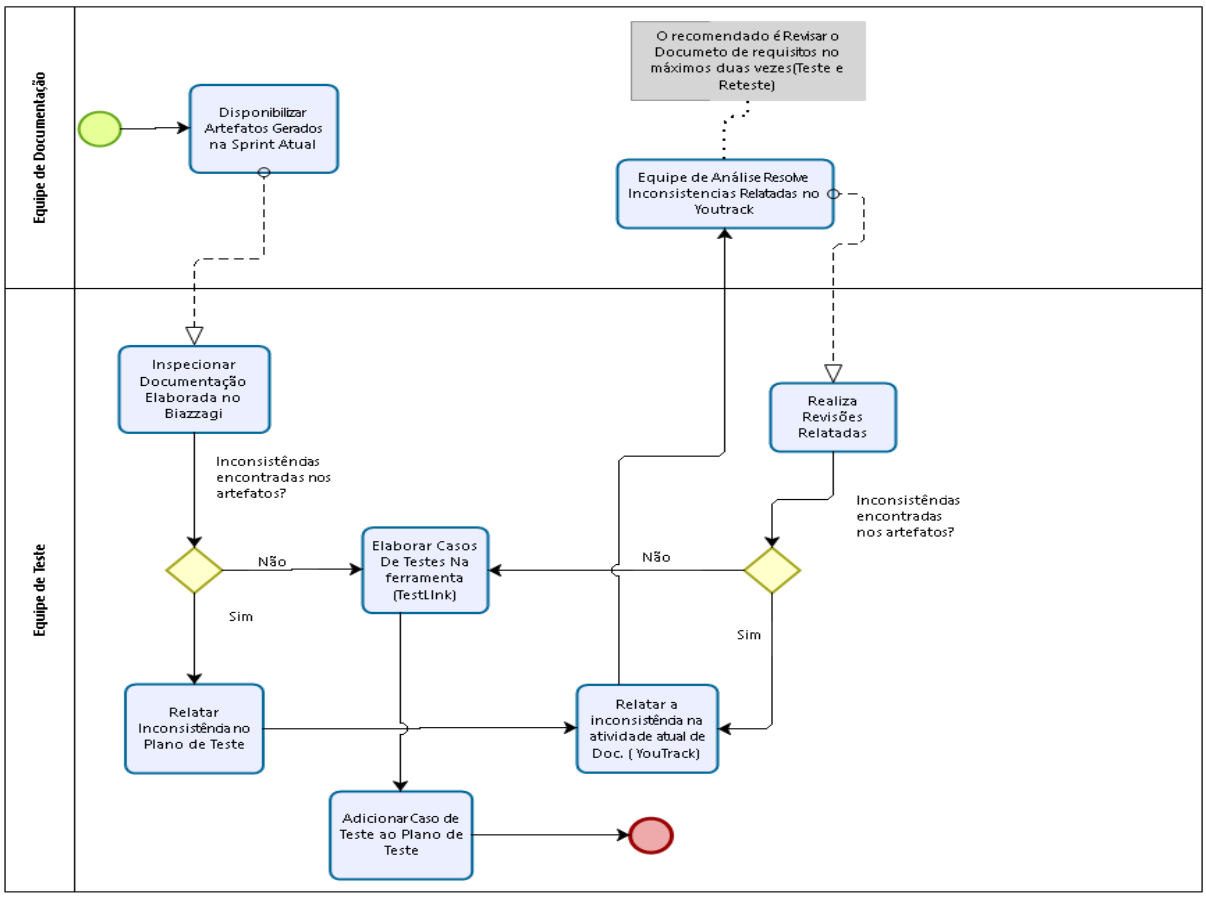
Uma Fábrica de *Software* é um processo estruturado, controlado e melhorado de forma contínua que considera abordagens de engenharia industrial da indústria tradicional. Seu objetivo é gerar produtos de software de forma segmentada, produtiva, econômica e dinâmica.

A FTT - Fábrica de Tecnologias *Turing*, fundada em 2006, é vinculada ao curso de Engenharia da Computação do Centro Universitário de Anápolis (UniEVANGÉLICA). Seu foco é promover a prática dos conceitos trabalhados no curso para prover sua consolidação através de projetos reais. Em outras palavras, consiste em um ambiente que possibilita a prática do desenvolvimento de tecnologias com a simulação do funcionamento de uma fábrica de Software da maneira mais realística possível.

Neste ambiente o aluno terá a oportunidade de vivenciar todas as etapas do processo de desenvolvimento dessas tecnologias: desde a análise de negócio até a realização da fase de testes, implantação e manutenção dos sistemas desenvolvidos.

Os projetos desenvolvidos têm entre um e seis anos de duração e envolvem entre oito e quatorze alunos estagiários, bolsistas e voluntários. O grupo de alunos é organizado em quatro equipes: i) gerência de projetos, onde os responsáveis tem o papel de *Scrum Master*; ii) equipe de documentação liderada pelo *ProductOwner* (PO), responsáveis pela elaboração dos documentos de *software;* iii) equipe de desenvolvimento, responsável por codificar o que foi elaborado pela equipe de documentação; e iv) equipe de teste, responsável por encontrar o máximo de erros dispondo do mínimo de esforço.

Figura 1. Processos de Atividades da Equipe de Teste e a interação com a equipes de documentação da FTT.



Fonte: Os Autores (2017).

Além de foco no mercado de trabalho, a FTT, promove atualização tecnológica contínua dos acadêmicos através dos núcleos de capacitação e pesquisa. Atende a projetos internos da instituição e projetos externos de empresas locais, desde que estejam de acordo com os objetivos e competências a serem desenvolvidos no ambiente mencionado.

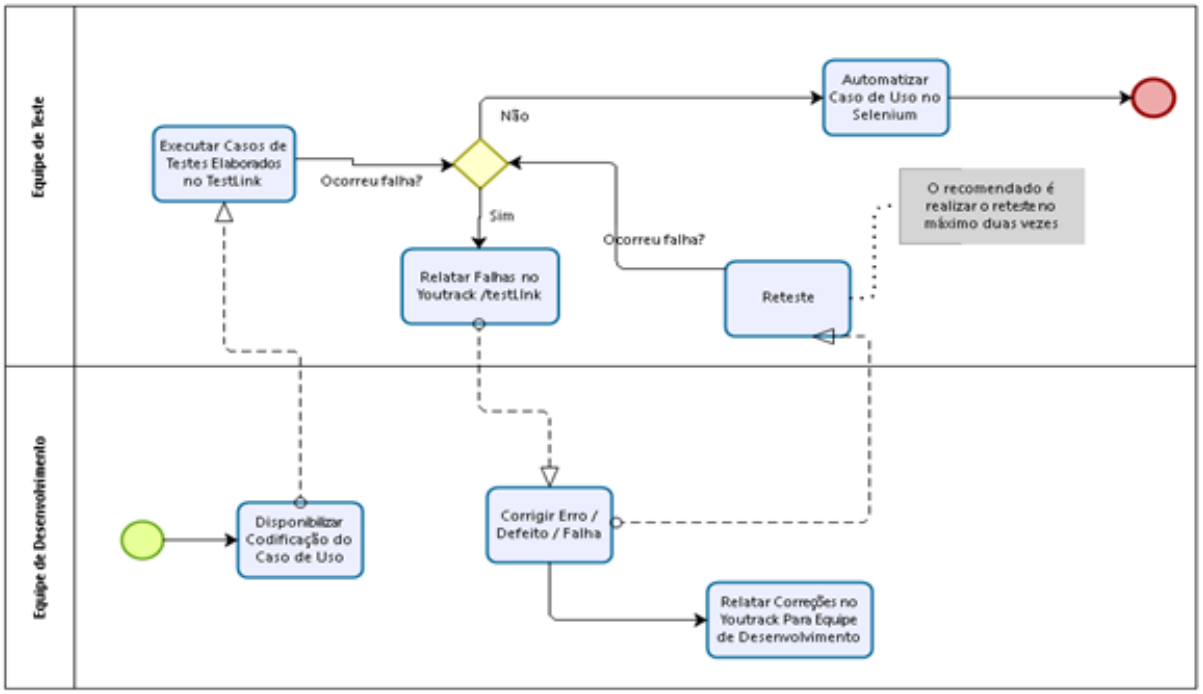
3.2. O Processo de Testes da FTT

Visando estabelecer um processo para equipe de teste o primeiro passo consistiu em analisar e compreender a metodologia em que a equipe de teste de *software* estava inserida e, em seguida, verificar quais as atividades seriam necessárias para atender aos princípios ágeis e à norma IEEE-829. O processo estabelecido para equipe de teste subdivide-se em duas etapas principais: i) inspeção; ii) testes funcionais; cujos passos podem ser observados no processo detalhado na sequência (Figura 1).

No que se refere à etapa de inspeção, de acordo com o processo definido, no decorrer de uma *Sprint*(ciclo de tempo) a equipe de teste deverá aguardar a documentação dos requisitos previsto na *Sprint Backlog* (lista de tarefas que o *Scrum Team* se compromete a realizar em uma *sprint*), disponibilizado pelo PO do projeto no repositório local da fábrica (através do sistema *GitBlit).* Após a disponibilização dos primeiros artefatos gerados, seja ele Fluxo de Processos, Diagrama de Casos de Uso ou Documento de Especificação de Casos de Uso, a equipe de testes deverá partir para a inspeção do documento seguindo a especificação do projeto de teste e o plano de teste elaborado pela equipe de testes.

Caso seja encontrada alguma inconsistência durante a inspeção dos requisitos, o analista de testes deverá relatar a inconsistência no seu plano de teste e ao analista de sistemas responsável pela documentação através da ferramenta pertinente à gerência de projetos (neste caso, foi adotado o *YouTrack*).

Figura 2. Processos de Atividades da Equipe de Teste e a interação com a equipes de desenvolvimento da FTT.



Fonte: Os Autores (2017).

Após a correção de todas as inconsistências, o analista de testes deverá iniciar a elaboração de casos de testes através da ferramenta utilizada para essa finalidade (atualmente o *TestLink).*Para

elaborar um caso de teste o analista de testes deverá utilizar os documentos que contém: Especificação de Requisitos, Especificação de Mensagem, Diagrama de Caso de Uso, Especificação de Regra de Negócio. Ao concluir os casos de teste, o analista deverá adicionar todos os casos de testes ao plano de teste para que os mesmos sejam executados nos testes funcionais detalhados na Figura 2.

Para iniciar os testes funcionais o analista de testes deverá acessar o sistema pertinente e executar os testes adicionados ao plano de teste (registrados na ferramenta *TestLink).* Caso seja encontrada alguma inconsistência, o analista deverá relatar a(s) inconsistência(s) através do sistema de gerenciamento utilizado (Y*ouTrack)* e atribuir ao responsável por corrigir erros, defeitos ou falhas. Após a fase de revisão, o requisito deverá estar na fase de "pronto" onde o analista poderá elaborar sua automatização através da ferramenta pertinente (*Selenium WebDriver).* Como resultado, serão gerados os *scripts* de todos os casos de testes elaborados que serão reutilizados em *sprints* subsequentes para realizar testes de integração, *stress*, regressão entre outros que se fizerem necessários ao longo do projeto.

Caso surjam atividades não programadas, o líder da equipe de testes deverá distribuí-las entre os membros da equipe de forma a garantir a continuidade, bom despenho e evitar o atraso de das atividades planejadas.

4. Método

Este estudo consiste em um relato de experiência vivenciado por alunos do curso de graduação em Engenharia de Computação, que atuam como estagiários da FTT, sediada no Centro Universitário de Anápolis em Goiás, acerca da formalização e aplicação de um processo de teste de *software*.

Através da aplicação dos métodos empírico e indutivo, a FTT foi utilizada como ambiente de observação e experimentação durante o período compreendido entre junho de 2016 a maio de 2017 e proveu informações para subsidiar a produção deste trabalho através de dados extraídos das ferramentas de gerenciamento utilizadas.

5. Resultados

Com a inserção do processo de Teste de *Software* notou-se uma melhora considerável na qualidade dos artefatos gerados, na implementação das funcionalidades durante o processo de desenvolvimento do *software* e, principalmente, na manutenção e atualização das funcionalidades após sua entrega. Para obter estes resultados foram aplicados o processo de Verificação e Validação de *Software* com foco na qualidade do produto desenvolvido e entregável.

A verificação tem o intuito de assegurar consistência, completude e corretude do produto em cada fase e entre fases consecutivas do clico de vida do *software* enquanto a validação visa assegurar que o produto final corresponda aos requisitos do usuário.

O Teste de *Software* está diretamente ligado à qualidade do *software*, deste modo, com a verificação da consistência do desenvolvimento e validação dos resultados obtidos junto ao cliente os projetos desenvolvidos na FTT após a implantação deste procedimento possuem uma qualidade superior e uma redução dos recursos humanos e de tempo empregados.

6. Considerações finais

Este trabalho se propôs a relatar a experiência no aprendizado de alunos do curso de graduação de Engenheira de Computação do Centro Universitário de Anápolis, através de uma fábrica de *software* acadêmica.

Teste de *Software* é uma atividade de grande importância do processo de desenvolvimento de *software* que, para garantir a qualidade dos produtos finais, deve ser acompanhado, avaliado e evoluído. Os modelos de maturidade de teste visam avaliar a qualidade das atividades envolvidas no processo de teste. Este artigo relata a experiência da aplicação de um processo em uma equipe de teste em um projeto real com a metodologia *Scrum* e *OpenUp.*

Ao demonstrar os resultados desta experiência, evidenciou contribuições que podem inspirara criação de novos cenários de Testes de *Software* bem estruturados e preparados para lidar de forma satisfatória com grandes projetos, bem como identificar potenciais melhorias.

As perspectivas futuras direcionam para a agregação de novos níveis e técnicas de testes ao processo atual, além de explorar as possibilidades de integração das ferramentas utilizadas e da intersecção entre suas informações para apresentar estatísticas reais e embasar as decisões gerenciais e de projeto.

O modelo de fábrica tende a se expandir no mercado como uma solução viável para boa parte dos problemas de custo, prazo e conhecimento técnico envolvidos em projetos de *software*, o que se solidifica através da crescente demanda por terceirização dos serviços de desenvolvimento de sistemas bem como constantes evoluções das áreas de conhecimento da engenharia de *software*.

Referências

BALDUINO, R. *Introduction to OpenUP (Open Unified Process)*. Disponível em <http://www.eclipse.org/epf/general/OpenUP.pdf>. Acesso em: 10 de maio de 2017.

BECK, K. et al. *Manifesto for Agile Software Development*. 2001. Diponível em: <http://www.agilemanifesto.org>. Acesso em: 10 de maio de 2017.

CRESPO, A. N., MARTINEZ, M. R., JINO, M., ARGOLO, M. T.; Application of the IEEE 829 Standard as a Basis for Structuring the Testing Process. *The Journal of Software Testing Professionals*, Vol. 3.

IBM. *OpenUp: Processo de Desenvolvimento de Software com Abordagem Ágil*. Disponível em: <http://epf.eclipse.org/wikis/openuppt/>. Acesso em: 10 de maio de 2017.

IEEE Computer Society. *SWEBOKGuide to the Software Engineering Body of Knowlegment*, 2004. Disponível em http://www.swebok.org/htmlformat.html. Acesso em: 10 de maio de 2017.

JUBILEU, A. P. *Modelo de Gestão do Processo de Venda e Desenvolvimento de Software On-Demand para MPE's*. Tese (Doutorado) – Universidade de São Paulo, São Carlos: 2008.

POCIVI, V. C. B. *Um estudo para melhoria do processo de ensino e aprendizagem de engenharia de software em cursos de graduação*. Recife, 2011. (Mestrado em Engenharia de Software) - Centro de Estudos e Sistemas Avançados do Recife – C.E.S.A.R, 2011.

PRESSMAN, R.S. *Engenharia de software*. São Paulo: Makron Books. 1995.

SOMMERVILLE, I. *Engenharia de Software*. 8 ed. São Paulo: Pearson Addison-Wesley, 2007.