SGB_PGT_PlanoDeGerenciaDeTestes

Versão 1.2

Histórico de Revisões

Nome	Alterações	Data	Versão
Guilherme Gonçalves	Criação do Documento	15/11/2012	0.1
Guilherme Gonçalves	Adicionados os tópicos de ferramentas, normas, e tipos de testes.	16/11/2012	0.5
Guilherme Gonçalves	Adicionado fluxograma de ativiades	16/11/2012	1.0
Raphael Rezende	Adicionado assunto de automação	23/11/2012	1.1
Raphael Rezende	Adicionado fluxograma de automação	27/11/2012	1.2

Sumário

1 Introdução

O presente plano visa estabelecer a Gerência de Testes no Projeto SGB, contemplando ferramentas a serem utilizadas, tipos de testes a serem feitos, normas a serem utilizadas, fluxograma e cronograma de atividades.

2 Ferramentas

Nesta seção estão descritas as ferramentas a serem utilizadas na cobertura dos testes.

2.1 Testlink

TestLink é uma ferramenta case de gerenciamento de teste. Essa ferramenta pertence e é mantida pela comunidade de teste de software, em todo o mundo. TestLink é utilizado para a criação e elaboração de Casos de teste e cenários de teste. Nessa ferramenta também é possível vincular os casos de teste de forma hierárquica e a um mesmo grupo de teste. TestLink não é uma ferramenta que executa caso de teste, essa ferramenta é utilizada para documentar os testes que serão executas por outras ferramentas ou manualmente.

2.1.1 Mantenedores da Ferramenta

- Os líderes de equipe
    - Francisco Mancardi - núcleo de desenvolvimento, de designer e revisor
    - Julian Krien - desenvolvedor, coordenação de projetos
- Os membros da equipe
    - Andreas Morsing - desenvolvimento do núcleo, re-factoring
    - Martin Havlat - gestão, desenvolvimento, lançamentos, documentação, web-master

- Toshiyuki Kawanishi - Japão coordenador da equipe, localização Japão, desenvolvedor
- Masami Ichikawa - testes, automação de testes
- Amit Khullar - desenvolvedor
- Erik Eloff - desenvolvedor
- Andreas Simon - desenvolvedor
- Asiel Brumfield - equipe rastreador administrador, desenvolvedor
- Kevin Levy - desenvolvedor (Reportagem; Requisitos)
- Chad Rosen - Originador do projeto

2.1.2 Configurações

Para instalar a ferramenta TestLink necessita-se que já tenha instalado os seguites softwares:

- Web Service Apache2;
- PHP 5 ou superior;
- MySql 5 ou superior.

2.2 Selenium Web Test

O Selenium Web Test é uma ferramenta de automação de testes funcionais para interfaces Web. Desenvolvida e mantida pela OpenQA como software livre utilizando a licença 2.0 da Apache.

Sua principal vantagem é a utilização do próprio navegador web para realização dos testes. É possível realizar o acesso a sistemas através de navegadores como o Firefox, Internet Explorer ou Safari.

O Selenium é uma ferramenta bem completa, pois comtempla tanto a parte de geração dos casos de testes quanto a de execução. Atualmente é mais considerado como uma plataforma.Divide-se em Selenium IDE, Selenium RC, Selenium Core e o Selenium Grid.

2.1.1 Selenium IDE

O Selenium IDE é um plugim para o Firefox que fornece um ambiente integrado para desenvolvimento de scripts de testes permitindo gravar, editar, executar e depurar os testes. Os scritps podem ser gerados em um padrão próprio do Selenium ou escolher

uma das linguagens de programação disponíveis, como Java, C#, PHP, Perl, Ruby, etc. A partir da captura automática das ações do usuário na tela do sistema é possível montar os procedimentos de testes que serão utilizados na execução dos testes.

2.1.2 Selenium CORE

O Selenium Core é um módulo que deve ser adicionado ao servidor web para a execução dos scripts na linguagem nativa do Selenium.

2.1.3 Selenium RC

O RC (Remote Control) permite a integração com ferramentas de testes unitários (como JUnit) aumentando muito a flexibilidade dos scripts de testes.

2.1.4 Selenium Grid

O Grid é um módulo do Selenium que permite a execução distribuída e em vários ambientes dos scripts de testes.

3 Normas

Nesta seção estão descritas as normas a serem utilizadas na cobertura dos testes. Importante ressaltar que as normas serão adaptadas ao contexto da fábrica podendo não ser executadas 100% a risca como estão descritas.

3.1 IEEE 829:2008

O IEEE 829 também conhecido como o Padrão 829 para Documentação de Teste de Software, é um padrão IEEE que especifica a forma de uso de um conjunto de documentos em oito estágios definidos de teste de software, cada estágio potencialmente produzindo seu próprio tipo de documento.

3.2 IEEE 1012:2004

Padrão de Verificação e Validação de software.

Determina se os produtos desenvolvidos dado uma atividade esta nos conformes dos requisitos daquela atividade e se o software satisfaz o planejamento e as necessidades do sistema. A 1012 é especificado para diferetes niveis de integridade.V&V

ajuda na construção da qualidade de um software durante seu ciclo de desenvolvomento, demonstrando se os requisitos do software e do sistema estão corretos, completos, precisos, concistente e testavel. Essa norma inclui a analise, avaliação revisão e inspeção do software e produtos gerados. Essa norma é seguida durante o desenvolvimento do software e não na conclusão.

3.3 IEEE 1044:1995

Padrão de Classificação de Anomalias

A IEEE Std 1044-1993 tem como propósito esquematizar a classificação das anomalias de software. O uso dessa norma no processo de Teste de Software para o projeto SGB, será como guia para elaboração e classificação das ocorrências (erros), defeitos e falhas identificadas na execução das atividades de Teste. Para que possa ser gerenciado o nível da qualidade dos requisitos e dos testes.

3.4 IEEE 1465:2004 (Adaptação da ISO 12119:1994)

Padrão de Requisitos de Qualidade e de Teste.

Requisitos de qualidade para os pacotes de software e instruções de como testar os pacotes de software em prol dos requisitos estabelecidos. Esse requisitos serão aplicados nos pacotes do software e serão oferecidos e entregados, exluido durante o processo de proução de especicação.

4 Tipos e Técnicas de Testes

Nesta seção estão descritas os tipos de teste a serem utilizados na cobertura dos artefatos.

4.1 Teste de Requisito

Para cada requisito funcional deve ser possível definir um ou mais testes a serem realizados no sistema final para ser possível verificar se o sistema cumpre o requisito na integra. Se este teste não for possível ser definido isso vai significar que o requisito necessita de ser clarificado pois muito provavelmente irá criar problemas no desenvolvimento do produto.

Na definição de cada teste deverá ser tomado em conta os seguintes pontos:

1. identificador do requisito

2. requisitos relacionados

3. descrição do teste

4. problemas na realização do teste

5. comentário e recomendações

4.2 Teste Funcional

Teste funcional é aquele onde o objetivo é verificar se uma dada implementação está de acordo com a sua respectiva especificação. Também é conhecido como teste “Caixa Preta”. Um tipo de teste funcional é o chamado teste formal. Nele, tanto as especificações como os procedimentos de geração de casos de teste são formalizados.

A identificação dos casos de teste devem se basear única e exclusivamente nas especificações dos componentes de implementação.

Tecnicas que serão utilizadas serão:

- Análise de Valores Limites.
- Partição por Equivalência.
- Teste Estado-Transição.
- Testes derivados da Especificação.

4.3 Teste de Regressão

O teste de regressão é uma técnica do teste de software que consiste na aplicação de testes à versão mais recente do software, para garantir que não surgiram novos defeitos em componentes já testados.

Utilizado durante o desenvolvimento depois de cada iteração, e em uma nova instância de um componente que estará sendo reutilizavel.

Se, ao juntar o novo componente ou as suas alterações com os componentes restantes do sistema surgirem novos defeitos em componentes inalterados, então considera-se que o sistema regrediu.

4.4 Teste de Aceitação

Tem por função verificar o sistema em relação aos seus requisitos originais, e às necessidades atuais do usuário.

O teste de aceitação é o teste feito antes da fase de implantação do software alvo. de acordo com a 829 será criado 3 estratégias de teste.

Teste de Aceitação formal:

- Executado pelo Usuario final ou por grupo de teste independente
- Funções e recursos devem se conhecidos
- Pode ser automatizado
- Deve ser monitorado e medido

Teste Alfa:

No teste Alfa, os procedimentos para executar o teste não são definidos com tanto rigor como no teste de aceitação formal. As funções e as tarefas de negócios a serem exploradas são identificadas e documentadas, mas não há casos de teste específicos para seguir. O testador individual determina o que fazer. Essa abordagem de teste de aceitação não é tão controlada como o teste formal e é mais subjetiva do que o tipo formal.

Teste Beta:

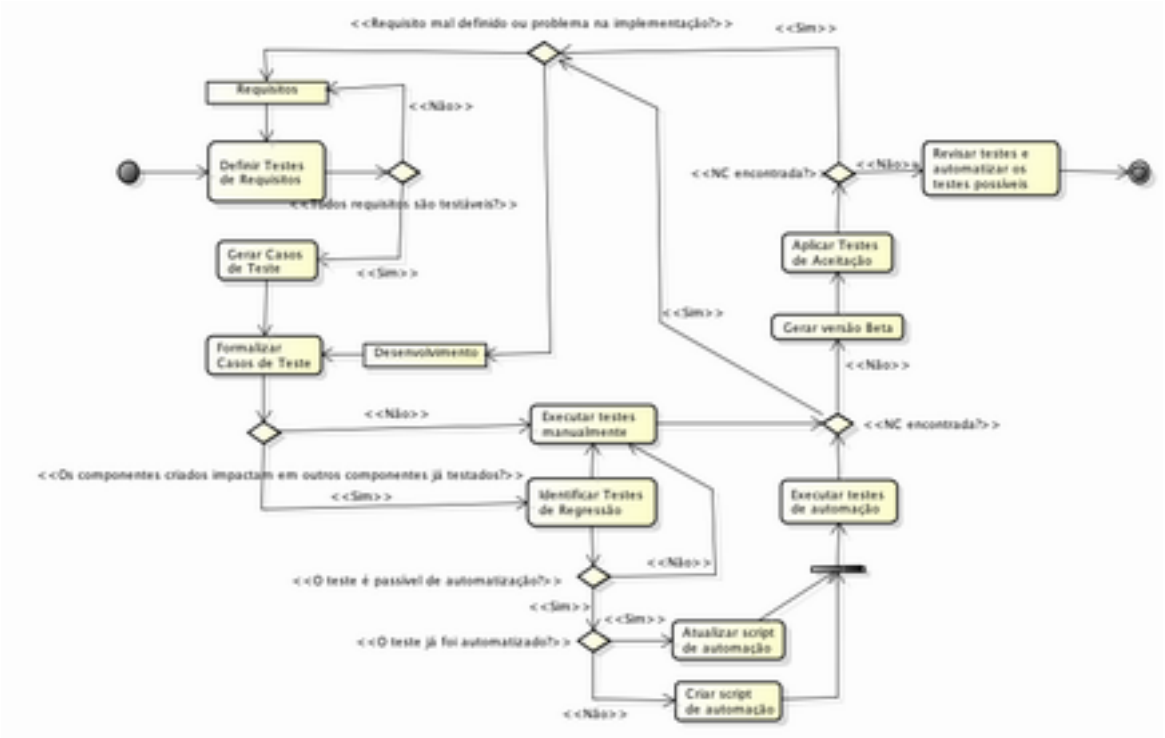
O teste beta é o menos controlado das três estratégias de teste de aceitação. Nesse tipo de teste, a quantidade de detalhes, os dados e a abordagem adotada são de inteira responsabilidade do testador individual. Cada testador é responsável por criar o próprio ambiente, selecionar os dados correspondentes e determinar as funções, os recursos ou as tarefas a serem exploradas. Cada um deles é responsável por identificar os próprios critérios que o levarão a aceitar ou rejeitar o sistema no seu estado atual.

O teste beta é implementado por usuários finais, geralmente com pouco ou nenhum gerenciamento por parte da organização de desenvolvimento (ou outra que não seja do usuário final).

4.4 Teste de Unidade

Teste de unidade objetiva testar a menor unidade funcional de um código, no caso de projetos orientados a objetos a menor unidade é o método de uma classe. Para esse projeto foram eleitos os dois principais componentes do sistema: Model e o Base. São eles que modelam as regras do negócio. O critério de aceitação é que possua cobertura de testes no mínimo de 90% sobre o Model e o Base

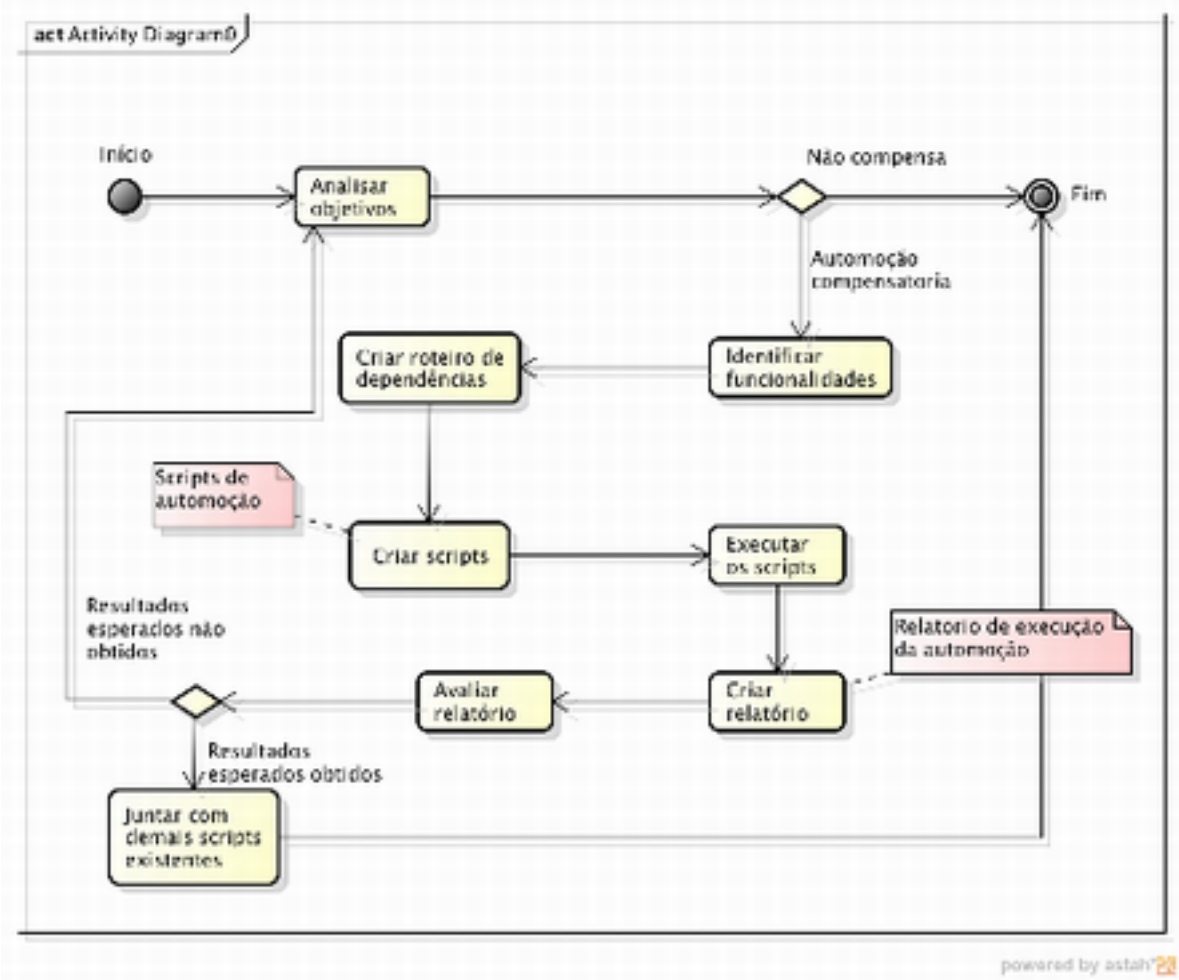
5 Fluxograma de atividades



A equipe de testes irá primeiramente definir os Testes de Requisitos. Deve-se verificar se foi possível gerar testes para todos os requisitos, problemas nos requisitos que impeçam isto, devem ser comunicados e repassados à equipe de qualidade de

processo. Então os Casos de Testes serão gerados, e formalizadas no momento que a equipe de testes tiver acesso à implementação. Os casos de testes deverão abranger não só Testes de Requisitos, mas testes relacionados à interface do sistema também. Deve-se verificar então se há impacto em componentes já testados por parte das novas implementações. Caso positivo, deve-se verificar quais componentes deverão passar pelos testes de regressão, e se esses testes são passíveis de automação, os testes que não poderão ser feitos automaticamente, terão de ser feitos manualmente, os que podem ser automatizados, deve-se verificar se já existe ou não script para ele no Selenium, caso positivo, o teste automatizado deve ter seu contexto atualizado para a atual situação, caso não haja script algum para este teste, deve-se criar o script. Criando um script ou atualizando um script, após isso o script deve ser rodado e o teste realizado. Antes de começar a criar os testes automatizados deve-se verificar se o cronograma existe determinado recurso para a criação deste tipo de testes, e mesmo se existir tem-se que analisar a viabilidade da automatização, para isto existe alguns pontos que devemos analisar: primeiramente devemos saber se as rotinas que serão testadas não são voláteis, pois a criação de script e muito onerosa, depois analisamos se o valor agregado daquilo que será testado, vale a pena pelo custo da criação dos scripts de testes automatizados. Com tanto os testes automatizados como manuais realizados, deve-se verificar as não-conformidades, e relatá-las às devidas áreas. Caso não sejam identificadas não conformidades, o software poderá ter uma versão gerada. Assim, com base nessa versão, serão aplicados os Testes de Aceitação que podem ser feitos tanto pelo cliente (Patrocinador ou Dono do Produto), ou por uma segunda equipe de testes, externa à atual equipe. Uma vez que os testes de aceitação são realizados e não conformidades encontradas corrigidas. Deve-se fazer um apanhado de todos os testes realizados na iteração e verificar quais deles podem ser automatizados, os que puderem, devem ter seus scripts gerados.

5.1 Fluxograma automação de testes



Após a liberação de uma versão do sistema devemos levantar a necessidade da automação de alguns testes, se julgarmos a automação para esse cronograma não for compensatoria então nada é feito, porém se for julgada compensatoria, identificamos as funcionalidades as quais deseja-se criar os testes automatizados, após isto devemos criar um roteiro de dependencias entre as funcionalidades, com base neste roteiro iremos por começar a criar os scripts de automação, após finalizados executaremos os scripts, então é criado um relatório da execução da automação, por fim analisamos este relatório e se os resultados obtidos forem diferente dos esperados, voltaremos a analisar os objetivos da automação, porém se os resultados forem iguais aos esperados, então juntamos os scripts gerados com os demais scripts existentes se caso houver, então é finalizada a tarefa.

5.2 Preenchimento do template de casos de teste

A seguir será detalhada a estrutura da criação de casos de teste e como deve ser preenchido cada campo desta estrutura.

O template dos casos de teste e definido da seguinte forma.

Título do Caso de Teste: deve ser inserido o titulo do caso de teste a ser criado.
Objetivo do Teste: deve ser informado os objetivos que devem ser alcançados com a execução do caso de teste, para causa de conferência da conformidade após a execução.
Pré-condições: Deve ser informado todas as condições que devem ser satisfeitas para que o caso de teste seja executado.
nº Ticket: deve ser inserido a URL do ticket no Redmine que abrange este caso de teste. Ex: http://fs.inf.ufg.br:3000/redmine/issues/ + o numero do ticket.
Prioridade do Teste: deve ser informado a urgência do caso de teste, sendo diferenciado por: alto, médio e baixo.
Palavras-chave: estas palavras-chave são informadas para facilitar nas buscas, estas

palavras são definidas primeiramente, e podem ser utilizadas em todos os casos de teste, após criar este catálogo poderá vincular algumas palavras-chave para o caso de teste.

Após ser criado o caso de teste, ele poderá ser vinculado a determinado requisito. Posteriormente ser criado e entrando na opção de editar o caso de teste, a opção de **Atribuir Requisito** é liberada, ao ser selecionada e aberta uma janela com os seguintes campos.

Documento de Especificação de Requisitos: deve ser selecionado a suíte em que os requisitos estão vinculados.
Requisitos atribuídos: caso já haja algum requisito vinculado ao caso de teste, estes serão mostrados, então poderão ser desvinculados.
Requisitos disponíveis: e dada uma lista com os requisitos que estão disponíveis naquela suíte selecionada, então é possível selecionar os requisitos e atribui-los ao caso de teste.