WebBug: uma ferramenta para gestão de erros em projetos de software

Arthur Astar Neves Barbosa Brito Amorim , Fabrício de Sousa Pinto[1]

[1]Faculdade de Tecnologia e Ciências (FTC)
Caixa Postal 450.020-510 – XXX – xx – Brazil

arthurastar@gmail.com, fabbricio7@yahoo.com.br

Abstract. *During the lifecycle of a software many types of failures can be found or can be requested new features. With regard to failures is an important issue that is well documented, so that the fault is corrected as soon as possible and will not happen again. This study aimed to develop WebBug. The WebBug is a multiplatform manager errors that allows the project to monitor and document software bugs, as well as the progress of activities manager, it is noteworthy that the tool offers full compatibility with smartphones and tablets. For the development of the tool was used PHP (Hypertext Preprocessor) and as the default database was selected, MySQL, PostgreSQL and though it supports SQLite. The tool was tested and compared with the validation was used by an academic project and the results were satisfactory.*

Resumo. *Durante o ciclo de vida de um software podem ser encontrados diversos tipos de falhas ou podem ser solicitadas novas funcionalidades. No que tange a questão falhas é importante que esteja bem documentado, de modo que a falha seja corrigida tão breve possível e não volte a acontecer. Este trabalho teve como objetivo desenvolver o WebBug. O WebBug é um gerenciador de erros multiplataforma que possibilita ao gestor do projeto acompanhar e documentar os erros do software, bem como o andamento das atividades, vale ressaltar que a ferramenta apresenta total compatibilidade com smartphones e tablets. Para o desenvolvimento da ferramenta foi utilizado o PHP (Hypertext Preprocessor) e como o banco de dados padrão foi escolhido o MySql, embora suporte o PostgreSQL e o SQLite. A ferramenta foi testada e com relação a validação foi utilizada por num projeto acadêmico e os resultados mostraram-se satisfatórios.*

1. Introdução

Na atualidade e com o súbito desenvolvimento da tecnologia, desenvolver e manter sistemas tornou-se uma árdua tarefa. Diversos fatores acabam gerando uma grande demanda de novas solicitações que por fim também gera expectativas no usuário final e, que nem sempre são atendidas em razão de falhas no gerenciamento de projetos de TI (Tecnologia da Informação). Desta maneira para que se possa continuar a construir softwares de qualidade, é necessário não somente uma equipe comprometida e focada, mas também uma comunicação eficiente entre a equipe de desenvolvimento e o usuário que irá utilizar o software.

Nesse contexto, surge a Gerência de Configuração de Software (GCS), como uma das áreas da Engenharia de Software, sendo esta diretamente responsável pelo controle de mudanças durante todo o ciclo de vida do software. Na atualidade a Gerência de Configuração de Software é uma ferramenta de vital importância para o sucesso de um projeto de desenvolvimento de software (PRESSMAN, 2006).

Durante o ciclo de vida de um software, acabam ocorrendo modificações que são inevitáveis, tais como mudanças nas necessidades do usuário, alteração do ambiente de utilização, correções de erros e defeitos, assim como melhorias nas suas funcionalidades. Todas estas modificações terminam por produzir mudanças nas prioridades do software, na sua documentação e nas regras de negócio (PRESSMAN, 2006). Diante desta constatação e com a atual situação do mercado de software, qualidade passa a ser um elemento crítico. Desta forma as empresas que almejam sobreviver e/ou alcançar expansão necessitam encontrar melhores práticas ou meios para assegurar a qualidade do seu produto final (SOFTEX, 2009).

Este trabalho tem como objetivo apresentar o desenvolvimento do *WebBug*, um gerenciador de defeitos gratuito, que funciona na plataforma web e que auxilia os gestores de projeto na Gerência de Configuração de Software. Durante o levantamento de requisitos para o desenvolvimento, foram realizadas análises de funcionalidade e usabilidade em dois *bug trackers*, o *Bugzilla* e o *MantisBT*. O *Bugzilla* é mantido pela *Mozila Foundation*, já o *MantisBT* é mantido graças a diversos colaboradores ao redor do mundo. É importante salientar, que ambos os softwares analisados são *Open Souce*, ou seja possuem sua distribuição livre.

A ferramenta desenvolvida foi disponibilizada para uso gratuito de acordo com a GPL (*General Public Licence*), para a comunidade acadêmica de forma que contribua de algum modo para ampliar o debate sobre o uso de ferramentas livres que possam ser utilizadas em atividades de gestão de configuração.

Este artigo é composto por seis seções. A primeira seção mostra a introdução do trabalho, na segunda seção temos o referencial teórico; a terceira apresenta a metodologia; na quarta seção apresenta o desenvolvimento da ferramenta; a quinta seção os trabalhos relacionados e na sexta as considerações finais.

2. Gestão de Configuração

A Engenharia de Software é uma área que leva em consideração todos os aspectos do desenvolvimento do software, do levantamento de requisitos, até a fase de manutenção deste (SOMMERVILLE, 2003). Esta engenharia possui diversas etapas, onde cada uma abrange ferramentas, métodos e procedimentos para o desenvolvimento do software. Segundo Pressman (2006), o ciclo de vida de um software consiste em todas as ações realizadas durante o desenvolvimento do software.

De forma geral, a gerência de configuração tem por finalidade responder a algumas questões, são elas: Por que mudou? Quando e o que mudou?, Quem fez a mudança? Esta mudança pode ser reproduzida?

Para Swebook (2004) a gerência de configuração, possui algumas subáreas, como a gerência de processo, identificação da configuração do software, controle de

configuração do software, status da contabilidade da configuração de software, auditoria da configuração de software e gerência da liberação e entrega do software. Neste presente trabalho apenas será exposta a subárea de controle de configuração do software.

2.1 Gerenciamento de erros em software

De acordo com Maldonado (2004), defeito poder ser definido pela inaptidão de algum componente em realizar a função pela qual foi originalmente projetado, falha pode ser representada pela situação na qual o software executa uma ação inesperada, é importante ressaltar que não existe falha caso o programa não tenha nenhum defeito. Por fim erro é o resultado de uma falha que podem exibir discrepâncias nas saídas exibidas das quais eram esperadas, ou seja, nem todas as falhas culminarão com erros no programa.

Ainda de acordo com Maldonado (2004) com relação aos defeitos, estes podem ser classificados da seguinte forma: defeito de funcionalidade, defeito de usabilidade, defeito de desempenho, defeito de saída, defeito de limite, defeito de cálculo, defeito de carga, defeito de hardware e defeito de software.

Segundo Caetano (2007), outro fator que deve ser levado em conta no momento do cadastro de um bug é a relação prioridade X severidade. A severidade define o impacto do defeito no funcionamento da aplicação. Já a prioridade define a ordem em que aqueles defeitos devem ser consertados.

2.2 Estudo de Usabilidade

Para Nielsen (2003), a usabilidade é um fator vital para a sobrevivência na web, sendo esta um atributo de qualidade que verifica o quão simples são as interfaces.

Nielsen defende cinco componentes de usabilidade: Aprendizagem: se é simples a realização das tarefas logo no primeiro contato: Eficiência: após o aprendizado, o usuário é rápido para realizar estas mesmas tarefas. Memorização: se os usuários lembram dos passos para realizar estas tarefas após algum tempo. Erros: forma pela qual os usuários lidam com os erros que possam acontecer. Satisfação: se a interface e o design agradam aos usuários

A interface do *WebBug* foi projetada seguindo estes cinco princípios.

3 Metodologia

Para o desenvolvimento das atividades de construção da referida ferramenta, foi inicialmente realizada uma revisão teórica sobre o tema de investigação. Logo em seguida procedeu-se a escolha de uma ferramenta para o desenvolvimento da modelagem do software proposto. Desta forma foi assim escolhido *ArgoUML*. Desenvolvido em *JAVA*, o *ArgoUML* é multiplataforma também sendo distribuído sob a licença BSD (*Berkeley Software Distribution*). No desenvolvimento da aplicação, foi aplicado o *InteliJ Idea*, uma IDE (*Integrated Development Environment*) completa que possui versões comerciais e não comerciais. O SGBD (Sistema Gerenciador de Banco de Dados) escolhido foi o *MySQL* versão 5.0.8, por se tratar de um banco de dados gratuito e que oferece um excelente desempenho sem um alto custo de processamento para o servidor (MySQL, 2012).

A linguagem de desenvolvimento selecionada para construir o software foi o PHP (*Hypertext Preprocessor*) na versão 5.3.15. Por ser uma linguagem com suporte à orientação e objetos, robusta e gratuita, o *PHP* oferece todos os recursos necessários para o desenvolvimento da aplicação. Em conjunto ao *PHP*, foi utilizado o *Smarty*. O Smarty é um template engine escrito em PHP que tem por objetivo separar a apresentação (*html, css, javascript*) da aplicação, proporcionando uma maior organização do código que está sendo escrito. (Smarty, 2012). Ainda em conjunto com o PHP o projeto de desenvolvimento requis o uso do *framework* e distribuidor de pacotes reutilizáveis PEAR (PEAR, 2012), para a construção de todos os formulários do sistema. Por fim os testes de funcionalidade foram realizados em conjunto com a codificação e validação.

4 Desenvolvimento da ferramenta de gestão de bug: *WebBug*

Dentre as principais características do *WebBug*, pode-se destacar o pleno gerenciamento de defeitos de software, cadastro de clientes, projetos, esforço despendido na resolução de um erro, *wiki*, possibilidade de criação de campos personalidades para coleta de informações, além de diversos relatórios. Outra característica que deve ser ressaltada, é sua compatibilidade com dispositivos móveis. Como foi explanado anteriormente, o *WebBug* foi desenvolvido para ser utilizado na plataforma web. A ferramenta está disponível no endereço: *http://fabriciosousa.com/webbug* e o código fonte está disponível no repositório: *https://sourceforge.net/projects/webbug/*

Desta forma, o *WebBug* permite controlar e prover manutenção de *bugs* em projetos de software, rastrear o bug desde o momento de sua criação passando pela fase de reportar o erro até correção.

De forma a facilitar a implementação, desenvolvimento e manutenção do sistema, este foi divido em módulos específicos. A Figura 01 demonstra os módulos que foram desenvolvidos:

Figura 01: Módulos do Sistema

O fluxo principal de trabalho do *WebBug*, como pode ser visto na Figura 02, é basicamente definido por três utilizadores. De início o usuário reporta um novo bug, em seguida o gerente de projetos reconhece o defeito, define as prioridades e severidades e agenda a correção com algum dos desenvolvedores, por fim o desenvolvedor corrige o defeito e informa ao gerente que este último foi realizado com sucesso.

Caso o gerente note algum problema, ele pode reenviar o bug para o mesmo ou para um novo desenvolvedor.

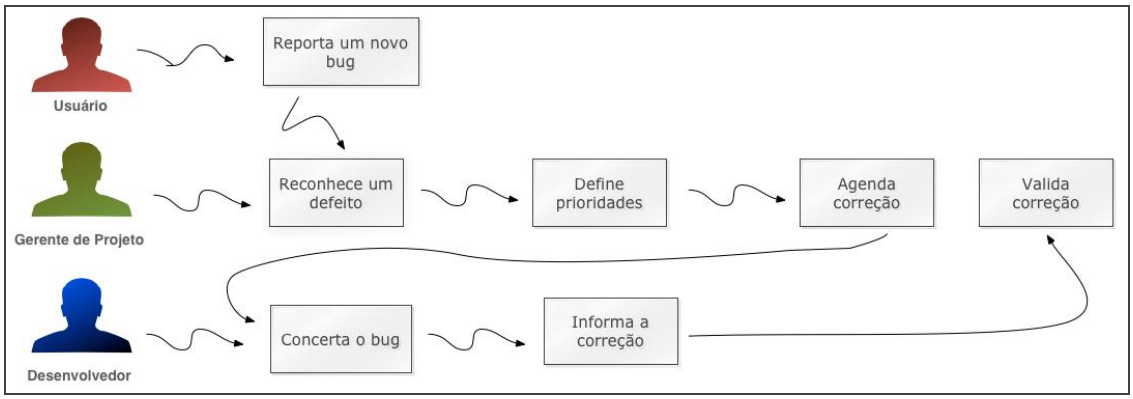

Figura 02: Fluxo de trabalho do WebBug

Após o *login*, o usuário é redirecionado para a tela de seleção de projetos. Nesta tela o mesmo deve selecionar o projeto no qual deseja utilizar o sistema. Caso o usuário não selecione o projeto, este estará impedido de utilizar o software, já que o nível de acesso é definido pelo projeto no qual ele selecionou.

Para melhor exemplificar, um usuário pode ser gerente de projeto no projeto X e ser um programador ou um usuário simples no projeto Y. A Figura 04 exibe a tela de seleção de projetos.

Na Figura 03 temos as "minhas atividades", é uma das telas mais utilizadas do sistema. Nela são exibidas todas atividades que estão pendentes para o usuário. No caso do gerente de projeto, assim que um novo bug é reportado, o sistema cria uma atividade para que ele possa verificar e dar continuidade ao processo. No momento em que o desenvolvedor conclui uma atividade, outra atividade é gerada para que o gerente do projeto possa validá-la. Desta forma o gerente de projetos tem controle sobre todos o processo de detecção, validação e correção de bugs. Para o desenvolvedor, são exibidas as atividades pendentes que lhe foram designadas, ordenadas por prioridade e severidade levando em consideração a previsão de término da mesma.

Figura 03: Tela de "minhas atividades"

Na Figura 04, temos a tela de reportagem de erros,é uma das mais importantes se não a mais importante tela do sistema. Nesta tela o usuário reporta um novo erro ao

sistema. De forma clara, o usuário pode visualizar os campos para a descrição do bug, informar os passos realizados para que o erro ocorresse e adicionar outras informações que ele entenda que são convenientes para a boa descrição do erro. Ainda nesta tela, o usuário pode enviar anexos de forma que comprovem o erro. O gerente do projeto, também pode criar outros campos que ele entenda que são relevantes para o projeto.

Na Figura 04 temos a tela de listagem de bugs, exibe todos os bugs do projeto ordenados por ordem cronológica, e sua cor de fundo representa a prioridade do mesmo.

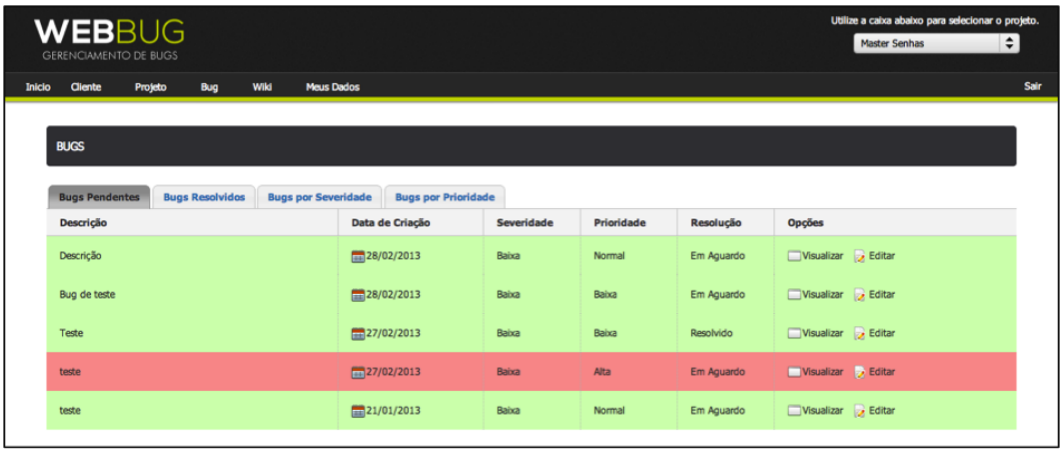

Figura 04: Listagem de bugs

De forma a facilitar a utilização do sistema por parte do gerente de projeto, as configurações do projeto ficaram organizadas em um único local, de modo que ele tem fácil acesso para fazer alterações necessárias no projeto. A Figura 05 representa o menu de configurações.

Figura 05: Menu de configurações

No momento em que o gerente de projetos verifica que a requisição em questão é realmente um bug, ele deve criar uma atividade e associá-la a um desenvolvedor. Desta forma, o bug irá aparecer nas “minhas atividades” do desenvolvedor selecionado e ele poderá dar início à correção do mesmo.

Seguindo os princípios de IHM, o *WebBug* evita que o usuário cometa erros ao preencher formulários e utilizar o sistema. Quando uma ação é executada, o sistema retorna uma mensagem clara para que o usuário entenda o que aconteceu A Figura 06 exemplifica as mensagens de erro e sucesso do sistema.

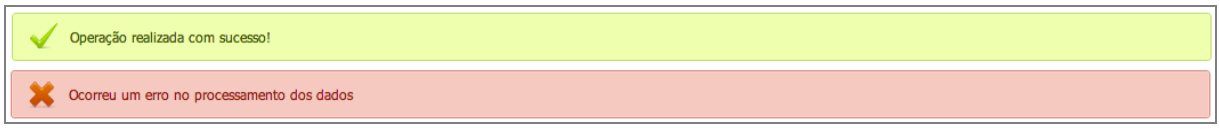

Figura 06: Mensagens de retorno do sistema

Antes que os formulários são enviados e preenchidos, são aplicadas regras de validação. A Figura 06 demonstra um formulário sendo validado. O *WebBug* possui diversos relatórios a fim de auxiliar no processo de gestão. Dentre os relatórios pode-se citar: Bugs por projeto, por usuário, número de bugs criados em um determinado período, tempo gasto em na resolução de uma determinada atividade. Para exemplificar os relatórios, a Figura 07 exibe o relatório de bugs em um determinado período.

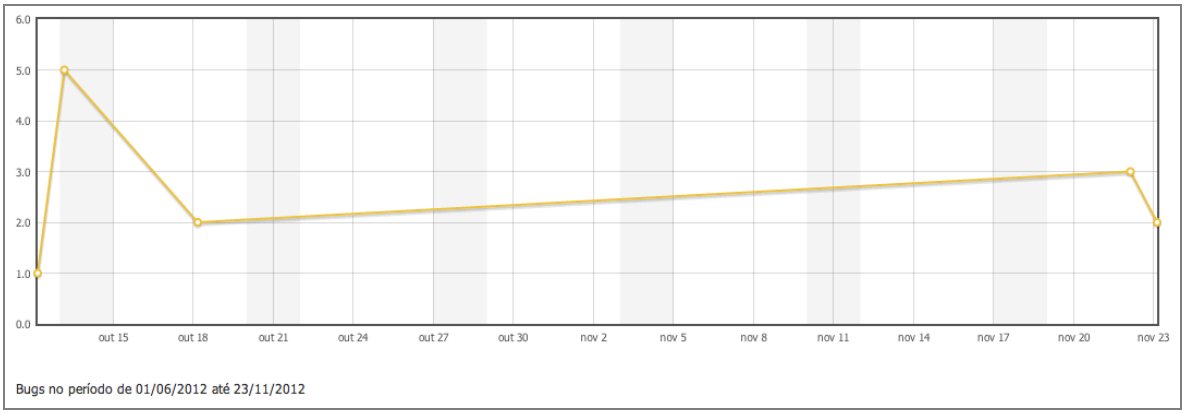

Figura 07: Relatório de bugs por período

4.1 Validação da Ferramenta

A ferramenta foi utilizada por uma turma da disciplina Engenharia de Software, da Faculdade de Tecnologia e Ciências (FTC), do campus de Vitória da Conquista, composta por dezesseis alunos divididos em quatro turmas, para a gestão de bugs. Após a execução do projeto, foi submetido um questionário para avaliação pelos alunos e obtemos o seguinte resultado: 87% ficou muito satisfeito, 10% satisfeito e 3% regular. Com relação a recomendação da ferramenta 95% recomendaria a ferramenta para ser utilizada e apenas 5% não recomendaria. Com relação aos princípios de IHM 92% acharam que o *Webbug* possui boa usabilidade e 8% não.

5 Trabalhos relacionados

Para o desenvolvimento do *WebBug*, foram selecionadas duas ferramentas já existentes, onde foi realizada uma análise de funcionamento e de interface. As ferramentas escolhidas foram o *Bugzilla* e o *MantisBT*. Estas ferramentas foram escolhidas por serem amplamente conhecidas na área de estudo.

O *Bugzilla* é uma ferramenta de apoio na gestão de erros de software, é principalmente utilizada como reportador de erros em software. Inicialmente o *Bugzilla* foi escrito em TCL (*Tool Command Language*). Hoje a ferramenta foi migrada para a linguagem *PERL*. Seu código está sob licença MPL (*Mozila Public Licence*) em conjunto com a GPL. A ferramenta suporta os bancos de dados *MySQL, PostgreSQL* e *Oracle* (ARAÚJO;DAIBERT;TOLEDO, 2010) .

Segundo Caetano (2007) ao contrário do Bugzilla o MantisBT é uma ferramenta escrita em PHP cujo código fonte também é *Open Source* e seu principal objetivo é prover suporte à gestão de defeitos de gerenciamento de erros baseada na web. Oferece suporte aos bancos de dados *MySql, MsSQL* e *PostgreSQL*. O *MantisBT* pode ser instalado em *Windows, Linux, MAC OS* dentre outros, já que funciona com servidor *web Apache*. É distribuído sob os termos da GNU (*General Public Licence*). Possui também uma distribuição denominada *MantisTouch*, que torna ferramenta amigável para dispositivos móveis.

Após a análise, foi desenvolvido um quadro comparativo com as principais características/funcionalidades de cada uma das ferramentas com as quais foram analisadas e na oportunidade também foram comparada as funcionalidades do *WebBug*, como pode ser observado na Tabela 3. Com relação a IHM (Interface Homem Máquina) os princípios analisados foram: diálogo simples e natural; falar a língua do usuário; consistência, *feedback* (retorno), boas mensagens de erros, prevenção de erros, ajuda e documentação, saídas claramente marcadas e uso de atalhos.

Tabela 3: Comparação de funcionalidades entre o Bugzilla, MantisBT e WebBug.

Características / Funcionalidades	***WebBug***	***MantisBT***	***Bugzilla***
Múltiplos projetos por instância	Sim	Sim	Sim
Baseado na Web	Sim	Sim	Sim
Segue princípios da IHM	Sim	Não	Não
Suporte a múltiplos bancos de dados	Sim	Sim	Sim
Relatórios	Sim	Sim	Sim
Wiki própria	Sim	Não	Não
Suporte para dispositivos móveis	Sim	Sim	Não
Campos personalizados para o projeto	Sim	Sim	Sim
Notificações por e-mail	Sim	Sim	Sim
Controle de horas gastas	Sim	Não	Não

Integração com o Portal de Engenharia de Software[1]	Sim	Não	Não
Suporte nativo ao português do Brasil	Sim	Não	Não

Fonte: Autoria Própria

6 Considerações Finais

Durante a pesquisa, pôde-se observar que existem diversos gerenciadores de erros em softwares no mercado, gratuitos e pagos. Estes quase que nunca seguem princípios da IHM o que termina dificultando o seu uso para o usuário final.

Aproveitando-se da tendência de se estar cada vez mais conectado, a estrutura do software foi desenvolvida para oferecer total compatibilidade com os navegadores mais modernos além dos principais *smarthphones* e *tablets* disponíveis no mercado.

Por se tratar de um projeto acadêmico, todo o código fonte além de todos os artefatos produzidos durante o desenvolvimento deste projeto foi disponibilizado de forma *Open Source*. O código fonte foi enviado ao repositório do Portal de Engenharia de Software, desta forma pesquisadores e usuários podem realizar o *download* da versão atual da ferramenta, realizar efetuar melhorias e disponibilizá-las para toda a comunidade. Também foi disponibilizada uma versão totalmente funcional da ferramenta para que os interessados utilizem a ferramenta antes de efetuar o download.

A gestão de erros é uma área extensa e para o propósito definido pode-se concluir que este trabalho alcançou as metas estabelecidas. Com o intuito de dar continuidade a este trabalho, em algum momento no futuro pode-se implementar novas funcionalidades , melhorar as existentes além de desenvolver pesquisas que ampliem o foco do tema.

Referências

BUGZILLA, Team. Bugzilla Development Roadmap. Disponível em <http://www.bugzilla.org/status/roadmap.html> Acesso em 25 de novembro de 2013.

CAETANO, Cristiano. Gestão de Defeitos - Ferramentas Open Source e melhores práticas na gestão de defeitos.Engenharia de Software, 2007, ed.:1

MALDONADO, J. C. et al. Introdução ao Teste de Software. Notas didáticas do ICMC, Instituto de Ciências Matemáticas e de Computação - ICMC/USP, São Carlos, Abril 2004

MYSQL. Why MySQL? Disponível em: <http://www.mysql.com/why-mysql/>

Acesso em 08 de novembro de 2013.

NETCRAFT. Web Server Survey. Disponível em: <http://news.netcraft.com/archives/2010/05/14/may_2010_web_server_survey.html>

[1] A ferramenta utiliza o mesmo banco de dados do *100 Risco* (Gestão de Risco) e GMétricas. (Gestão de Métricas). Compartilhando desta forma todas as informações.

Acesso em 16 de março de 2013

NIELSEN, Jackob. Usability 101: Introduction to Usability. Alertbox: Current Issues in web Usability. Disponível em < http://www.useit.com/alertbox/20030825.html> Acesso em 1 de dezembro de 2013

PEAR. PHP extension and Application Repository. Disponível em: <http://pear.php.net> Acesso em 14 de novembro de 2013.

PRESSMAN, R. S. Software Engineering: A Pracctiitioner's Approach. In: PRESSMAN R. S. Engenharia de Sofware (R. D. Penteado, Trad.) 6 ed, São Paulo: Mcgraw-Hill, 2006.

SMARTY. All about smarty. Disponível em: <http://www.smarty.net/about_smarty> Acesso em 11 de novembro de 2013.

TOLEDO, J. V; DUMONT, M. S, ARAÚJO,M. A. P. “Gerenciamento de Defeitos em Projetos de Software.” Engenharia de Software, 2010, ed:.2