METODOLOGIA DE DESENVOLVIMENTO E APLICAÇÃO DE TESTES PARA QUALIDADE DE SOFTWARE

Adriano Villa Busch[1]
Ricardo Augusto Lodi[2]
Sueli de Fátima Poppi Borba[3]

BUSCH, A. V.; LODI, R. A.; BORBA, S. F. P. Metodologia de desenvolvimento e aplicação de testes para qualidade de software. **Rev. Ciên. Empresariais da UNIPAR**, Umuarama, v. 8, n. 1 e 2, p. 125-139, jan./dez. 2007.

RESUMO: O estudo demonstra a importância da aplicação de um método correto de desenvolvimento de software, visando diminuir a detecção de bugs, aumentando a confiabilidade do software. Para isso, foi desenvolvida uma metodologia e sua aplicação no processo de planejamento, desenvolvimento e execução de procedimentos nas aplicações para um software de gestão empresarial. O desenvolvimento do presente artigo foi baseado em um estudo de caso que apresenta os processos utilizados em uma empresa de desenvolvimento de software, em busca da qualidade de seu produto.
PALAVRAS-CHAVE: Engenharia de software. Qualidade de software. Testes de software. Metodologia de desenvolvimento.

DEVELOPMENT METHODOGOLY AND APPLICATION OF SOFTWARE QUALITY TESTS

ABSTRACT: This paper demonstrates the importance of applying a correct software development methodology for reducing bugs occurrence and raising software reliability. Therefore, a methodology and its application in planning, developing and running procedures on the applications for a management software were developed. This paper development is based on processes used in a software development company in a pursuit of quality of its product.
KEYWORDS: Software engineering. Software quality. Software tests. Development methodology.

---

[1] Pós-Graduado em Análise e Desenvolvimento de Sistemas Orientados a Objeto com UML - Unipar – Universidade Paranaense - adrianobusch@uol.com.br
[2] Pós-Graduado em Análise e Desenvolvimento de Sistemas Orientados a Objeto com UML - Unipar – Universidade Paranaense - lodi@infomark.com.br
[3] Coordenadora do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas - Professora orientadora - Unipar – Universidade Paranaense - sueli@unipar.br

METODOLOGÍA DE DESARROLLO Y APLICACIÓN DE TESTES PARA CALIDAD DE SOFTWARE

RESUMEN: Este estudio demuestra la importancia de aplicación de un método correcto de desarrollo de software buscando disminuir la detección de Bugs, aumentando la confiabilidad del software. Para eso, fue desarrollada una metodología y su aplicación en el proceso de planificación, desarrollo y ejecución de procedimientos en las aplicaciones para un software de gestión empresarial. El desarrollo de este artículo fue basado en un estudio de caso que presenta los procesos utilizados en una empresa de desarrollo de software, en búsqueda de calidad de su producto.
PALABRAS CLAVE: Ingeniería de software. Calidad de software. Testes de software. Metodología de desarrollo.

1 INTRODUÇÃO

Partindo-se de um princípio de que todos os programadores são bons no que fazem, e nunca cometem erros, poderíamos garantir que todos os *softwares* funcionariam corretamente, e assim teríamos uma situação ideal. Infelizmente, esta não é a realidade. Isso porque os programas possuem um grande número de estados, com fórmulas complexas, atividades e algoritmos. O tamanho do projeto a ser desenvolvido e a quantidade de pessoas envolvidas no processo aumentam ainda mais a complexidade. Assim, a presença de falhas é inevitável. Mas o que significa dizer que um programa falhou? Basicamente significa que o funcionamento do programa não está de acordo com o esperado pelo usuário.

Uma falha é o resultado de um ou mais defeitos em algum aspecto do sistema. Para tentar diminuir estas falhas nos *softwares*, a engenharia de *software* conta com processos de qualidade de *software*, e uma parcela destes processos são os testes.

O teste do *software* é uma das fases do processo de engenharia de *software* que visa atingir um nível de qualidade de um produto e encontrar seus defeitos, para que estes possam ser corrigidos pela equipe de programadores, antes de sua entrega.

Existem atualmente várias definições para o conceito de teste de *software*. De uma forma simples, testar um *software* significa verificar, através de uma execução controlada, se o seu comportamento está de acordo com o especificado. O objetivo principal desta tarefa é encontrar o número máximo de erros e mostrar aos que desenvolvem se os resultados estão, ou não, de acordo com os padrões estabelecidos.

O presente artigo propõe uma metodologia de desenvolvimento de *soft-*

ware. Trata-se de um método que parte do presuposto do envolvimento de vários setores na empresa, como a parte de suporte, análise e desenvolvimento, visando aprimorar a organização, podendo melhorar a qualidade final do *software* e satisfazendo cada vez mais o usuário final do sistema. A estrutura do artigo está organizada da seguinte forma: a seção 2 aborda aspectos conceituais sobre qualidade de *software.* A seção 3 apresenta os testes de *software* e técnicas de teste. Na seção 4 é apresentada a metodologia desenvolvida para a realização dos testes propostos. A seção 5 discorre sobre a implantação de metodologias de testes. Na seção 6 é feita a aplicação da nova metodologia. Por fim, na seção 7, são apresentados os resultados e considerações finais.

2 QUALIDADE DE *SOFTWARE*

É impossível negar que a qualidade de *software* é uma meta muito importante. Segundo Pressman (1996) qualidade de *software* pode ser definida como a "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". Para Inthurn (2001), "qualidade de *software* é um conjunto de propriedades a serem satisfeitas, de modo que o *software* atenda às necessidades de seus usuários".

Os fatores que afetam a qualidade de um *software* podem ser categorizados. McCall apud Pressman (1996) propôs uma categorização útil dos fatores que afetam a qualidade de um *software,* conforme ilustra o quadro 1.

Quadro 1: Categorias dos fatores que afetam a qualidade de um software

Categoria	**Função**	**Aplicação**
Corretitude	Ele faz aquilo que se quer?	À medida que um programa satisfaz sua especificação e cumpre os objetivos visados pelo cliente.
Confiabilidade	Ele se comporta com precisão o tempo todo?	À medida que se pode esperar que um programa execute sua função pretendida, com a precisão exigida.
Eficiência	Ele rodará no hardware tão bem quanto possível?	A quantidade de recursos de computação e de código exigida para que um programa execute sua função.

Integridade	Ele é seguro?	À medida que o acesso ao software ou a dados por pessoas não-autorizadas pode ser controlado.
Usabilidade	Ele foi projetado para o usuário?	O esforço para aprender, operar, preparar a entrada e interpretar a saída de um programa.
Manutenibilidade	É possível consertá-lo?	O esforço exigido para localizar e reparar erros num programa.
Flexibilidade	É possível mudá-lo?	O esforço exigido para modificar um programa operacional.
Testabilidade	É possível testá-lo?	O esforço exigido para testar um programa a fim de garantir que ele execute sua função pretendida.
Portabilidade	É possível usá-lo em outra máquina?	O esforço exigido para transferir o programa de um ambiente de sistema de hardware e/ou software para outro.
Reusabilidade	É possível reutilizar parte do software?	À medida que um programa (ou partes de um programa) pode ser reusado em outras aplicações.
Interoperabilidade	É possível compor uma interface com outro sistema?	O esforço exigido para se acoplar um sistema a outro.

Fonte: Adaptado de Pressman (1996).

3 TESTES DE SOFTWARE

Para Inthurn (2001), "teste é uma das áreas da engenharia de software e tem como objetivo aprimorar a produtividade e fornecer evidências da confiabilidade e da qualidade do software, em complemento a outras atividades de garantia de qualidade ao longo do processo de desenvolvimento de software" ou ainda "teste de software é o processo de executar um programa com a intenção de descobrir um erro".

A fase de testes de software é muito desafiadora para o engenheiro de

software, pois é quando o engenheiro cria uma série de casos que têm por objetivo "demolir" o software que ele mesmo construiu. Por este motivo, a atividade de teste é um passo do processo de engenharia de software que, muitas vezes, é visto como destrutivo, ao invés de construtivo. Segundo Ballestero-Alvarez (2000), a fase de testes deve passar por uma etapa em que "ocorre a entrada de dados e arquivos especialmente montados e, propositalmente, devem ser incluídos todos os tipos de erros possíveis, todas as operações inválidas e todas as exceções possíveis. Todas as deficiências e todos os erros deverão ser corrigidos. A simulação deverá ser repetida tantas vezes quantas sejam necessárias, até que funcione corretamente".

3.1 Técnicas de teste

Na atualidade, existem muitos métodos e muitas técnicas para se testar um software. Mesmo assim, existem técnicas que sempre foram muito utilizadas e que ainda têm grande valia. Apesar das metodologias e ferramentas de desenvolvimento atuais serem completamente diferentes das antigas, o objetivo principal destas técnicas continua sendo o de encontrar falhas no software. As duas principais técnicas de software são as nomeadas caixa branca e caixa preta (Pádua, 2003).

A técnica da caixa branca "tem por objetivo determinar defeitos na estrutura interna do produto, através do desenho de testes que exercitem suficientemente os possíveis caminhos de execução". Dentro desta categoria de teste, o testador tem acesso ao código fonte da aplicação e pode construir códigos para elaborar casos de teste que cubram todas as possibilidades do programa. Dessa maneira, todas as variações originadas por estruturas de condições são testadas.

Na técnica da caixa preta, o testador não possui acesso ao código fonte do programa. O objetivo é efetuar testes sobre as diversas funcionalidades e verificar se os resultados gerados estão de acordo com o esperado. Estes testes devem levar em consideração todos os eventos que possam ser disparados pelo usuário, como por exemplo, cada clique de mouse a ser realizado em uma interface. Os testes de caixa preta não verificam como ocorre o processamento, mas apenas os resultados produzidos.

3.2 Categorias de teste

Os testes podem ser separados nas seguintes categorias: unidade, componente, integração e sistema.

O escopo alvo dos testes de unidade são pequenos trechos de código. Assim, seu objetivo é o de encontrar falhas de funcionamento dentro de uma

pequena parte do sistema, funcionando independentemente do todo. A atividade de testes de unidade normalmente é considerada uma continuação da etapa de codificação. Depois do código fonte desenvolvido e revisado, iniciam-se os testes de unidade.

Os testes de componente têm por objetivo testar o componente como um todo e não apenas pequenos trechos do código. No entanto, este tipo de teste continua a ser executado sem considerar a integração com outras partes do sistema, ou seja, ele considera apenas o componente a ser testado e não o sistema como um todo.

A partir dos módulos testados por unidade são utilizados os testes de integração, para se construir uma estrutura de programa com o intuito de encontrar falhas provenientes da integração dos componentes do sistema. Normalmente as falhas encontradas são de envio e recebimento de dados, ou seja, alguns módulos retornam ou recebem valores incorretos ou inesperados, fazendo com que outros módulos não reconheçam estes valores, causando uma falha.

Para Pádua (2003), teste de Sistema "é o teste do software em um ambiente semelhante ao ambiente operacional". Ou seja, os testes são executados nos mesmos ambientes, com as mesmas condições e com os mesmos dados que um usuário utilizaria em seu dia-a-dia de manipulação do sistema. O objetivo é avaliar o funcionamento do software em condições reais de trabalho diárias.

4 METODOLOGIA DE DESENVOLVIMENTO E TESTES DE SOFTWARE

Para o desenvolvimento do presente artigo foi utilizado, para análise da proposta, um estudo de caso realizado em uma empresa de desenvolvimento de software. A empresa em questão está passando por um processo de implantação de novas metodologias que visam à qualidade de software. Este processo já passou por uma série de fases, até chegar ao modelo que está sendo utilizado hoje.

A metodologia é composta das seguintes fases: a solicitação de manutenção (SM), que são as solicitações feitas pelo suporte; a análise da SM, que é a fase de estudo de viabilidade do desenvolvimento da SM; a prototipação, ou seja, a elaboração da interface no padrão do sistema; o desenvolvimento, que é a fase em que a SM será desenvolvida e previamente analisada pelo analista; o teste, que é a etapa na qual serão verificados possíveis bug's no sistema; a padronização, que é a etapa de verificação da usabilidade da interface pelo usuário final; a atualização, que é a finalização do processo quando serão atualizados os programas modificados. A figura 1 ilustra como se dá o trâmite do processo de desenvolvimento.

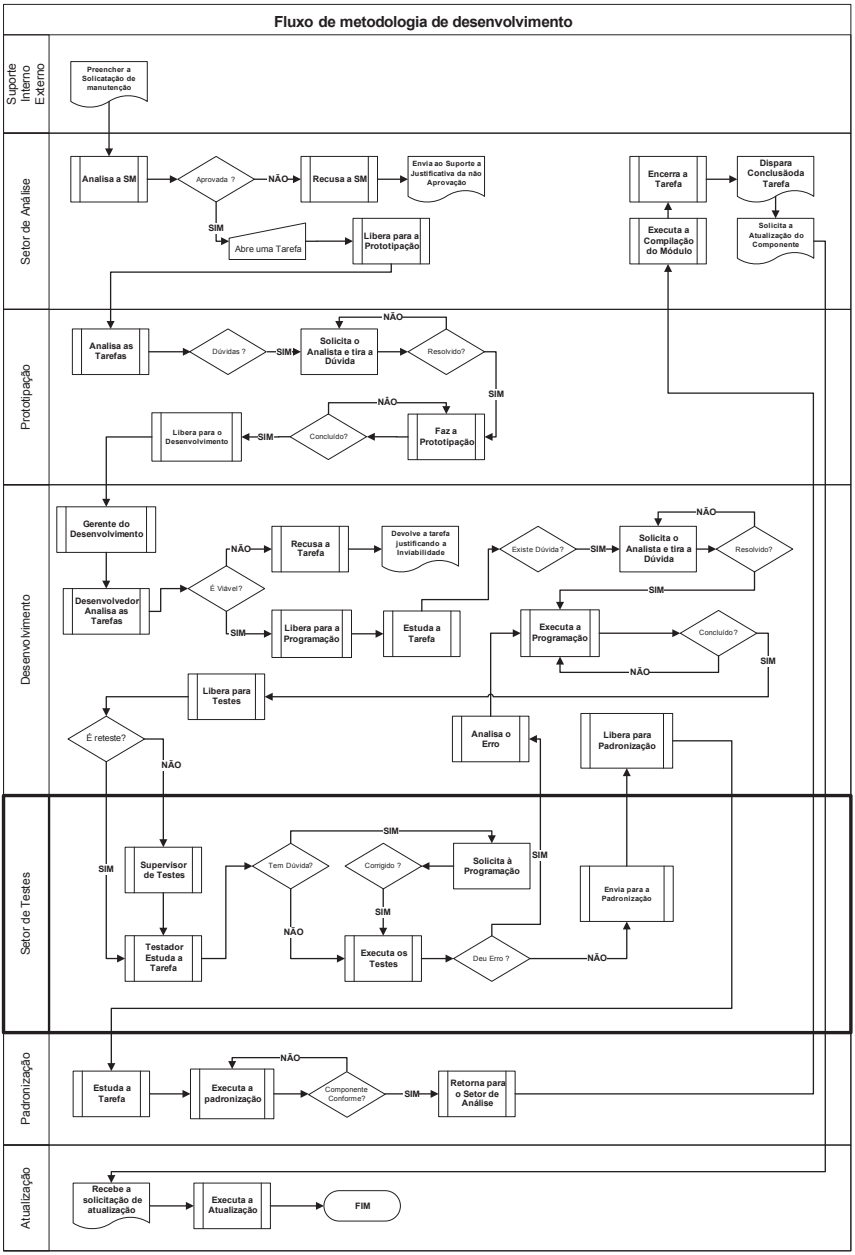

Figura 1: Fluxo de metodologia de desenvolvimento

4.1 Solicitação de Manutenção – SM

O termo SM é definido para identificar todas as solicitações de manutenção, que são feitas para o sistema. Sempre que um bug é encontrado ou algum cliente solicita a implementação ou padronização de algo no sistema, digita-se uma SM.

Uma SM pode possuir três tipagens, (Melhoria, Padronização e Correção/Bug). O quadro 2 apresenta um resumo destas tipagens.

Quadro 2: Tipagem de solicitação de manutenção (SM's)

Tipagem	Objetivo	Exemplo
Melhoria	Melhorar ou agilizar o sistema.	Solicitações de criação de novos relatórios, inserção de novos campos e até mesmo novos processos.
Padronização	Corrigir inconformidades do sistema.	Esta inconformidade pode ser tanto visual quanto funcional. É visual quando esteticamente alguma coisa está fora do padrão, como por exemplo, tamanho dos botões, fontes das letras, cores dos campos, etc. É funcional, quando o comportamento do programa não está de acordo com o especificado. Por exemplo, é padrão do sistema que quando pressionada a tecla TAB em um formulário, o cursor salta e para o próximo campo. No entanto, em uma determinada situação, pressionando-se a tecla TAB, o cursos saltou para o último campo do formulário.
Correção/Bug	Correção de bugs/problemas.	Este tipo de solicitação tem prioridade sobre as outras duas. Quando é detectado que existe algum bug no sistema, imediatamente é digitada uma SM de Correção/Bug e encaminhada para o desenvolvimento, a fim de que o problema possa ser corrigido.

4.2 Análise de viabilidade da SM

Assim que o setor de análise recebe uma SM, são verificadas duas coisas: primeiramente, avalia-se a viabilidade de fazer o que está sendo pedido pela SM; em seguida, é observado se o que está sendo solicitado é de uso comum a todos os clientes do sistema ou é uma particularidade do cliente solicitante; caso seja uma particularidade, será estabelecido e negociado um valor, para que a SM possa ser desenvolvida.

Cada SM é individualmente analisada, de acordo com as áreas de atua-

ção de cada analista, ou seja: financeiro, fiscal, faturamento, produção etc. Caso a solicitação esteja de difícil compreensão ou não seja possível desenvolver o que está sendo solicitado, o analista recusará a SM, encaminhando, junto com a recusa, o respectivo motivo. Caso o analista aceite a SM, ele fará um trabalho de detalhamento, no qual é feita uma especificação técnica do que precisa ser feito e como o desenvolvedor deverá executar esta tarefa. A partir daí, o analista gera uma "tarefa" que é encaminhada para o setor de prototipação, conforme se observa na figura 1.

4.3 Prototipação da SM

A etapa de prototipação consiste em construir as telas e formulários de acordo com os padrões estabelecidos pelo sistema. Nem toda tarefa exige o trabalho do setor de prototipação. Este setor só entra em funcionamento se a solicitação exigir um novo processo ou formulário. Vide a etapa de prototipação na figura 1.

4.4 Desenvolvimento da SM

A partir de uma SM, os analistas geram tarefas para que estas sejam encaminhadas para o setor de desenvolvimento. Estas tarefas contêm informações técnicas, tais como nome de tabelas, views, campos, templates etc., necessárias para que os desenvolvedores utilizem na implementação do que foi solicitado na SM.

A equipe de desenvolvimento é dividida por áreas como financeiro, faturamento, produção, fiscal etc. Desta forma, as tarefas chegam para os desenvolvedores segundo suas especialidades e, após a elaboração do que foi solicitado, o desenvolvedor encaminha o programa para o setor de testes.

4.5 Testes da SM

Este setor é responsável pelos testes de funcionalidade, exaustão, integridade, usabilidade, eficiência etc. A equipe de testes controla seus testes em uma tela, conforme ilustra a figura 2.

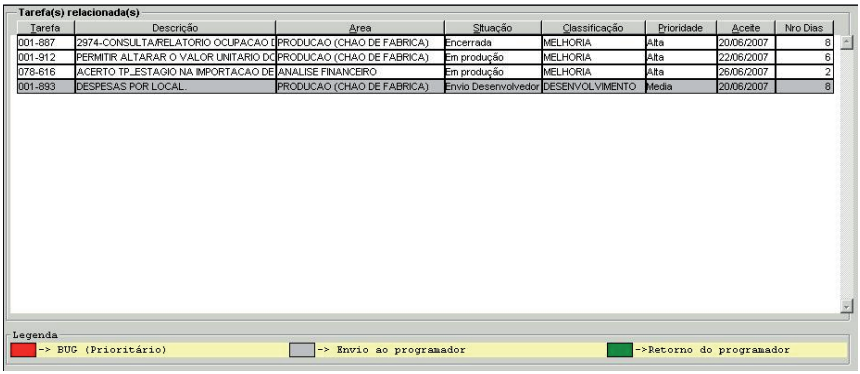

Figura 2: Tela de controle de testes

Os testes podem assumir três tipagens, conforme mostra a figura 2. Vermelho, indica ao testador que se trata de um bug e deve ser testado com prioridade sobre os outros testes; Cinza, indica que o testador encontrou um problema no programa que está sendo testado e encaminhou a tarefa de volta ao desenvolvedor, para que ele possa corrigi-lo; Verde, é quando o desenvolvedor devolve o teste para o testador com o problema solucionado, permitindo que o testador novamente efetue os testes. Este processo de ida e volta entre testador e desenvolvedor se repete, até que o testador não encontre mais nenhum problema. Neste momento, o testador encerra o teste, encaminhando a tarefa de volta para o supervisor de testes. Esta etapa é ilustrada na figura 1.

4.6 Padronização

Semelhante aos testadores, a padronização executa testes com o intuito de identificar inconformidades visuais ou funcionais nos programas, conforme explicação no tópico 4.1. A tela para controle dos testes de padronização é igual à tela dos testadores mostrada no tópico anterior. A ilustração desta etapa é apresentada na figura 1.

4.7 Atualização

Após ter passado por todos estes processos, a tarefa volta para o analista. Este, por sua vez, fará um estudo para verificar se a tarefa está de acordo com o que foi solicitado. Em caso positivo, o analista encaminha esta tarefa para o gerente dos analistas, para que seja efetuada a atualização desta SM em todos os servidores e, assim, tornar-se disponível para todos os clientes. Caso a tarefa não

tenha ficado em conformidade com o que foi solicitado, o analista encaminhará de volta ao desenvolvedor responsável e o ciclo se inicia novamente, conforme representado na figura 1.

5. A EVOLUÇÃO DA IMPLANTAÇÃO DE METODOLOGIAS

A metodologia de desenvolvimento utilizada atualmente pela empresa em questão sofreu uma série de mudanças até chegar ao modelo atual, que foi descrito na seção 4. Esta evolução pode ser dividida em seis fases:

- Na primeira fase, o processo de desenvolvimento contava apenas com os setores de desenvolvimento e suporte. Assim, os desenvolvedores, além de programarem, tinham que dar suporte a clientes. Essa equipe de suporte fazia o atendimento a clientes, tanto internamente, quanto em visitas.
- A segunda fase veio para melhorar o formato do setor de suporte. Para isso ocorreu uma divisão da equipe em duas partes, suporte interno (Supin) e suporte externo (Supex). A equipe de Supin passou a oferecer atendimento interno aos clientes, por telefone ou ferramentas de comunicação instantânea. A equipe de Supex ficou responsável pelas visitas. Com isso, os desenvolvedores podiam se dedicar exclusivamente à programação.
- Foi na terceira fase que surgiu o setor de padronização, com o intuito de alcançar um padrão para o desenvolvimento do software.
- A quarta fase buscou melhorar o formato do setor de desenvolvimento. Para isso, semelhante ao que foi feito com o setor de suporte, o setor de desenvolvimento também sofreu uma divisão. Foram criados os setores de desenvolvimento e análise. O setor de análise ficou responsável por discutir, analisar e estudar a viabilidade de cada projeto e detalhar tarefas para que a equipe de desenvolvimento fique concentrada apenas em programar.
- Uma das mais importantes fases para este trabalho foi a quinta, pois foi nesta fase que nasceu o setor de testes. Esta equipe em conjunto com as equipes de desenvolvimento e análise, busca a qualidade de software, executando testes e corrigindo os problemas junto a equipe de desenvolvimento.
- Não menos importante que as outras, a sexta e última fase trouxe a criação do setor de prototipação. Este setor veio para auxiliar os setores de desenvolvimento e análise e é responsável pela criação das interfaces.

A figura 3 ilustra as fases do processo de implantação de metodologias de desenvolvimento.

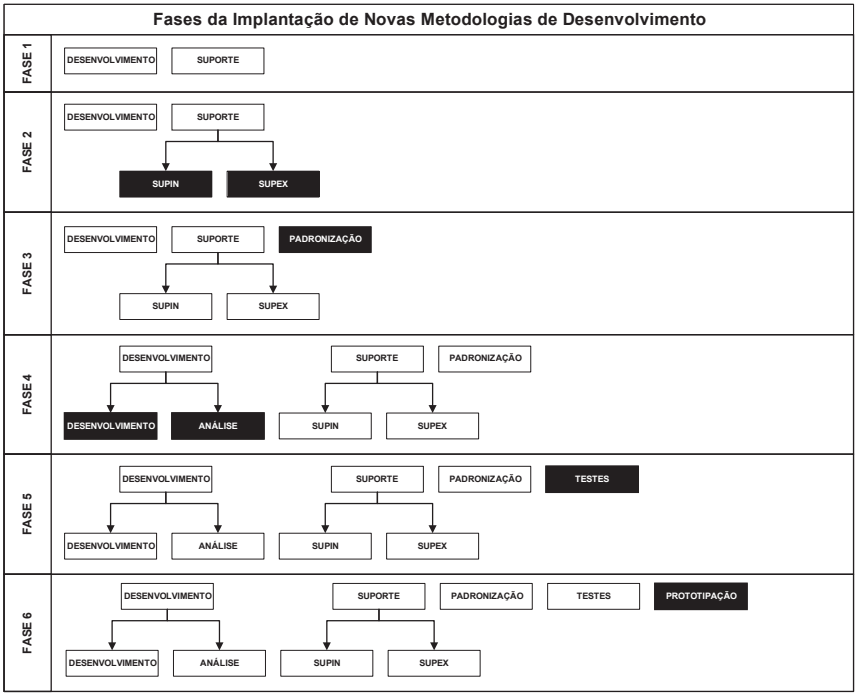

Figura 3: Fases da implantação de novas metodologias de desenvolvimento

6 APLICAÇÃO DA NOVA METODOLOGIA DE DESENVOLVIMENTO

Conforme afirmado na seção 4, para o presente artigo foi utilizado um estudo de caso de uma empresa de desenvolvimento de software. Sendo assim, para exemplificar a aplicação desta nova metodologia de desenvolvimento e demonstrar seus benefícios, o trabalho utilizou o método de amostragem. A seguir a descrição de como foi esta aplicação.

A figura 1 apresentada na seção 4 ilustra sete fases/setores que compõem a nova metodologia que está sendo apresentada; destas sete fases/setores, quatro (suporte, análise, desenvolvimento, teste) possuem uma política de divisão por área de especialidade, ou seja, estes setores são subdivididos de forma que cada integrante de cada setor seja especialista em apenas uma destas áreas de atuação (financeiro, faturamento, fiscal, produção etc).

Diante deste cenário, foi coletada uma amostra das SMs digitadas para a área de faturamento no período de 01/01/2008 a 31/01/2008, com o total de 38 SM´s e o resultado pode ser visto no quadro 3.

Quadro 3: SM´s abertas no período de 01/01/2008 a 31/01/2008 – área de faturamento

SM	Tipagem de SM			Grau de Dificuldade	Data de Inclusão	Data de Atualização	Situação em 31/01/2008	Retorno p/ correção após 31/01/2008
	Melhoria	Padroni-zação	Correção Bug					
6122	X			Altíssimo	3/1/2008	30/1/2008	Atualizado	
6123	X			Alto	3/1/2008	30/1/2008	Atualizado	
6142	X			Altíssimo	3/1/2008	30/1/2008	Atualizado	
6156	X			Altíssimo	7/1/2008	30/1/2008	Atualizado	
6164	X			Alto	8/1/2008	30/1/2008	Atualizado	
6165	X			Alto	8/1/2008	30/1/2008	Atualizado	
6225	X			Altíssimo	9/1/2008	30/1/2008	Atualizado	
6261	X			Altíssimo	11/1/2008	30/1/2008	Atualizado	
6280	X			Médio	11/1/2008	30/1/2008	Atualizado	
6308	X			Alto	14/1/2008	30/1/2008	Atualizado	
6310	X			Baixo	15/1/2008	30/1/2008	Atualizado	
6314	X			Altíssimo	16/1/2008	30/1/2008	Atualizado	15/2/2008
6328					16/1/2008		Recusada	
6329	X			Alto	16/1/2008	30/1/2008	Atualizado	
6352	X			Baixo	16/1/2008	30/1/2008	Atualizado	
6355		X		Altíssimo	16/1/2008	17/1/2008	Atualizado	
6370	X			Baixo	17/1/2008	30/1/2008	Atualizado	
6388	X			Alto	17/1/2008	30/1/2008	Atualizado	
6391	X			Baixo	17/1/2008	30/1/2008	Atualizado	
6395			X	Altíssimo	18/1/2008	18/1/2008	Atualizado	
6397			X	Médio	18/1/2008	18/1/2008	Atualizado	
6443		X		Médio	18/1/2008	18/1/2008	Atualizado	
6353	X			Altíssimo	18/1/2008	30/1/2008	Atualizado	
6472	X			Médio	18/1/2008	30/1/2008	Atualizado	
6473	X			Altíssimo	22/1/2008	30/1/2008	Atualizado	
6474					22/1/2008		Recusada	
6483	X			Altíssimo	22/1/2008	30/1/2008	Atualizado	
6513	X			Altíssimo	22/1/2008	30/1/2008	Atualizado	
6514	X			Altíssimo	23/1/2008	30/1/2008	Atualizado	25/2/2008
6530	X			Médio	23/1/2008	30/1/2008	Atualizado	

6531	X			Médio	23/1/2008	30/1/2008	Atualizado	
6544			X	Médio	23/1/2008	30/1/2008	Atualizado	
6551	X			Altíssimo	24/1/2008	30/1/2008	Atualizado	
6559	X			Baixo	24/1/2008	30/1/2008	Atualizado	
6566			X	Altíssimo	28/1/2008	29/1/2008	Atualizado	
6576	X			Altíssimo	28/1/2008	27/2/2008	Ag. Atualização	
6605	X			Médio	29/1/2008	27/2/2008	Ag. Atualização	
6608	X			Altíssimo	31/1/2008	27/2/2008	Ag. Atualização	

Através de uma análise da amostragem do quadro 3, percebe-se que no mês de janeiro de 2008 foram abertas trinta e oito SM's para a área de faturamento, sendo que 84% destas SM's foram para o tipo "Melhoria", 5% para "Padronização" e 11% para correção de bugs.

Da mesma forma, é observado que 45% das SM's tinham um grau de dificuldade "altíssimo", 16% "alto", 21% "médio", 13% "baixo" e, apenas duas SM's, 5% tiveram seu pedido recusado, ou seja, percebe-se um grau de dificuldade de desenvolvimento muito elevado e um grau de recusa muito baixo.

Foram efetuadas várias análises dos dados apresentados no quadro 3, mas um dos fatores mais importantes a serem observados é que das trinta e oito SM's abertas no mês de janeiro de 2008, apenas duas precisaram retornar para receberem algum tipo de manutenção. Esse fato que comprova um nível de qualidade muito alto das SM's, que passam por todas as fases da nova metodologia de desenvolvimento.

7 CONSIDERAÇÕES FINAIS

Vários fatores levaram a empresa em questão a adotar novas metodologias que a ajudassem no desenvolvimento de seu produto. Contudo, neste artigo, é analisada a contribuição da implantação de cada fase na evolução do processo de desenvolvimento em busca da qualidade de software.

A descrição da aplicação da nova metodologia de desenvolvimento trouxe resultados importantes. Além de exemplificar como ocorre o processo de controle das SM's, também permitiu observar as vantagens de se aplicar esta nova metodologia no desenvolvimento de software.

Deve-se ressaltar que seres humanos são passíveis a erros e que isso pode influenciar consideravelmente a qualidade de um software. No entanto,

com a aplicação correta de todas as fases da metodologia de desenvolvimento apresentada neste trabalho, observa-se que é muito pequena a probabilidade de este *software* vir a apresentar problemas depois de ter sido liberado para utilização.

Sendo assim, evidencia-se que esta nova metodologia de desenvolvimento auxilia e muito na busca pela qualidade de software da empresa em questão. No entanto, se fazem necessários novos estudos e pesquisas mais aprofundadas, seja para a criação de novas fases ou para a adaptação destas fases à realidade de uma outra empresa de desenvolvimento.

REFERÊNCIAS

BALLESTERO-ALVAREZ, M. E. **Manual de organização, sistemas e métodos**: abordagem teórica e prática da engenharia da informação. 2. ed. São Paulo: Atlas, 2000.

FILHO, W. de P. P. **Engenharia de software**: fundamentos, métodos e padrões. Rio de Janeiro: Ltc, 2003.

GUSTAFSON, D. A. **Engenharia de software**: teoria e problemas de engenharia de software. Porto Alegre: Bookman, 2003.

INTHURN, C. **Qualidade & teste de software**. Florianópolis: Visual Books, 2001.

PETERS, J. F.; PEDRYCZ, W. **Engenharia de software**: teoria e prática. Rio de Janeiro: Campus, 2001.

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

TESTE de software. Disponível em: <http://pt.wikipedia.org>. Acesso em: 15 jun. 2007.