FACULDADE DE TECNOLOGIA DA ZONA LESTE – FATEC-ZL

SARA ALVES SALDANHA

Qualidade de software e a importância dos testes nas empresas de desenvolvimento de software

São Paulo

2009

SARA ALVES SALDANHA

Qualidade de software e a importância dos testes nas empresas de desenvolvimento de software

Monografia apresentada à Faculdade de Tecnologia da Zona Leste FATEC - ZL para obtenção do Título de Tecnólogo em Informática para a Gestão de Negócios.

Orientador: Prof. José Abel de Andrade Baptista

São Paulo

2009

Saldanha, Sara Alves.
Estudo Qualidade de Software e as Vantagens de aplicar os testes durante seu desenvolvimento / Sara Alves Saldanha ; orientador José Abel de Andrade Baptista – São Paulo, SP: [s.n], 2009.
56 f.

Monografia (TCC) – Faculdade de Tecnologia da Zona Leste, 2009.

1.Software. 2. Qualidade. 3. Teste de Software. Baptista, J.A. de Andrade. Título: Qualidade de software e a importância dos testes nas empresas de desenvolvimento de software.

Nome: Saldanha, Sara Alves

Titulo: Qualidade de software e a importância do teste integrado

Monografia apresentada à Faculdade de Tecnologia da Zona Leste – Fatec ZL, para obtenção do título de Tecnólogo em Informática para Gestão de Negócios.

Aprovado em:

Banca Examinadora

Prof. _____ Instituição: _____
Julgamento_____ Assinatura: _____

Prof. _____ Instituição: _____
Julgamento_____ Assinatura: _____

Prof. _____ Instituição: _____
Julgamento_____ Assinatura: _____

Prof. _____ Instituição: _____
Julgamento_____ Assinatura: _____

Este trabalho é dedicado à minha mãe, meu alicerce, que sempre esteve ao meu lado, sorrindo comigo nos momentos felizes e chorando nos momentos de dificuldade e por seu incansável apoio durante essa trajetória até a elaboração deste trabalho. Dedico também ao meu pai, no qual sem ele, não teria chego até aqui. Este trabalho é dedicado a vocês, com toda minha gratidão, amor e carinho.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por tudo que me tem proporcionado.

Aos meus pais Teresinha e Saldanha e meus irmãos Mateus e Samuel pela paciência e pelo apoio.

Ao professor Abel Baptista que me orientou e ajudou durante o desenvolvimento deste trabalho.

Aos colegas que de alguma forma contribuíram com o desenvolvimento deste trabalho, em especial, ao Rodrigo, que jamais se negou a oferecer ajuda quando precisei.

A Engenharia deve criar soluções com uma relação custo-benefício adequadas para problemas práticos, pela aplicação de conhecimentos científicos, para construir coisas a serviço da humanidade.

(Junior, José Barreto)

RESUMO

Saldanha, S.A. **Qualidade de software e a importância dos testes nas empresas de desenvolvimento de software.** 2009. 54 f. Monografia (TCC) - Faculdade de Tecnologia da Zona Leste, São Paulo, 2009.

A qualidade é um requisito que deve estar associado a todos os tipos de serviço prestados. Entretanto, quando se trata do desenvolvimento de um software, há muito mais que exigir, pois um único sistema é capaz de integrar todos os departamentos de uma empresa. Quando um cliente deseja comprar um sistema para implementar em sua empresa, ele espera que esse software traga soluções rápidas e eficientes. Muitas vezes, os integrantes de TI não conseguem planejar suas ações, tornando-as imprevisíveis e até caóticas; isto ocorre porque a documentação dos processos, às vezes, não está atualizada ou sequer existe. Os poucos processos estáveis existentes estão sujeitos a esforços individuais, já que não há padrões a serem seguidos ou, se houver, são ignorados; as ferramentas são usadas sem controle gerencial, ou seja, ao acaso; as metodologias são aplicadas informalmente, sem que haja resultados previsíveis. O sucesso, portanto, depende diretamente dos desenvolvedores. Atualmente nas organizações é possível observar a preocupação cada vez maior com a qualidade. Por este motivo ocorre uma constante busca para o conhecimento de metodologias que fazem referência a qualidade e normas que possuem padrões que classificam as empresas mediante auditorias. E com a implementação dessas metodologias e normas é criado um diferencial no mercado, agregando valores importantíssimos à empresa. Na visão de Mowen e Minor (2003, p. 221), “a satisfação do consumidor é a atitude referente a um produto ou serviço após sua compra e uso”. Este trabalho visa demonstrar como desenvolver um software de qualidade, encontrando e prevendo seus erros através do teste integrado, que será analisado mais adiante. Com isso, deseja-se também demonstrar o quão é importante o teste integrado para as empresas de desenvolvimento de softwares. Entretanto, muitas organizações ainda se garantem apenas com o teste unitário, feito pelo próprio desenvolvedor, pois uma área de teste integrado poderia gerar custos que são considerados desnecessários. Os resultados desta pesquisa reforçam a importância da qualidade no desenvolvimento

de um software e como fazê-lo para que o serviço entregue garanta 100% de satisfação do cliente.

Palavras-Chave: Qualidade, Software, Teste.

ABSTRACT

Saldanha, S.A. **Qualidade de software e a importância dos testes nas empresas de desenvolvimento de software.** 2009. 54 f. Monografia (TCC) - Faculdade de Tecnologia da Zona Leste, São Paulo, 2009.

The quality is a requirement that must be associated with all types of service provided. However, when it comes to software development, there are many more that require it as a single system is able to integrate all departments of a company. When a customer wants to buy a system to implement in your company, he hopes that this software brings quick and efficient solutions. Many times, TI members can not plan their actions, making them unpredictable and even chaotic, this is because the documentation of processes, sometimes is not updated or even exists. The few existing stable processes are subject to individual efforts, since there are no standards to be followed or, if any, are ignored, the tools are used without management control, randomly, the methodologies are applied informally, without predictable results. The success, therefore, depends directly of the developers. Currently in organizations is possible to observe the increasing concern with the quality. For this reason there is a constant search for knowledge of methodologies that make reference to quality and standards that have standards that classify companies through audits. And with the implementation of these methodologies and standards is created a gap in the market, adding value to the very important business. In view of Mowen and Minor (2003, p. 221), "customer satisfaction is the attitude towards a product or service after your purchase and use." This paper demonstrates how to develop a quality software, finding and providing their mistakes through the integrated test, which will be discussed later. Thus, we want to also demonstrate how important is the integrated test for business software development. However, many organizations still meet only with the unit testing, done by the developer, as an area of integrated test could generate costs that are considered unnecessary. The results of this research reinforce the importance of quality in software development and how to do it for the service delivered to ensure 100% customer satisfaction.

Keywords: Software, Quality, Test

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

SQA Software Quality Assurance (Garantia da Qualidade de Software)

CMMI Capability Maturity Model Integration (Modelo de Integração de Maturidade e Capacitação)

ISO International Organization for Standardization (Organização Internacional de Normalização)

DDA Débito Direto Autorizado

TFC Terminal Financeiro Corporativo

SUMÁRIO

1. INTRODUÇÃO

Qualidade é um termo que faz parte do dia a dia e desempenha papel importante em todos os tipos de organização.

Na Engenharia de Software, qualidade é uma das áreas de conhecimento que tem como objetivo garantir a conformidade do software com as expectativas do cliente, através da definição e normatização de processos de desenvolvimento. Os modelos aplicados na garantia da Qualidade de Software atuam principalmente no processo e no produto.

Atualmente, é possível observar uma necessidade cada vez maior com a Qualidade. Com a aplicação de metodologias e normas, é criado um diferencial no mercado, valorizando muito mais as organizações e seus produtos.

Dessa forma fica clara a necessidade que as empresas apresentam em criar ou aperfeiçoar métodos que garantam a obtenção da qualidade em seus processos e produtos finais. Tornando-a um fator importante e grande motivador dentro da organização.

Devido essa necessidade de aprimoramento e oferta de qualidade, muitas empresas de desenvolvimento têm aderido às técnicas de testes de sistemas.

Durante as décadas 1970, 1980 e 1990, os testes eram realizados pelos próprios desenvolvedores, o que é chamado de teste unitário, entretanto, as informações extraídas do teste unitário não permitiam encontrar todos os possíveis defeitos do sistema em desenvolvimento o que resultava maiores custos às empresas, pois as correções acabavam sendo efetuadas no período de implementação, momento em que a correção costuma ser muito mais cara. Problemas ainda maiores surgiram com o desenvolvimento de sistemas para a internet e as empresas passaram a ser cada vez mais expostas ao público.

Com isso, as organizações passaram a procurar meios de aperfeiçoar suas técnicas de testes, o que necessitou de um processo paralelo ao de desenvolvimento, o que conta com técnicos especializados, que dessem suporte exclusivamente aos projetos de teste. Isso trouxe resultados imediatos sobre a qualidade do produto entregue. Foram criadas diversas técnicas e normas para qualificar os testes efetuados nos sistemas em desenvolvimento ou que sofreram alguma modificação.

Durante este projeto, serão definidos os conceitos de algumas das normas mais conhecidas e requisitadas pelas organizações e também, quais os tipos de testes criados para atender especificamente cada caso, de cada sistema, seguido de um estudo de caso que visa demonstrar e comprovar esta teoria.

2. REFERENCIAL TEÓRICO

Neste capítulo são citadas as definições de software, qualidade, qualidade de software e testes de software. Tem por objetivo esclarecer os conceitos, funcionalidades, importâncias e necessidades desses componentes. Serão abordados assuntos relacionados à qualidade de software e quais métodos e modelos de qualidade foram desenvolvidos para alcançar esse objetivo com sucesso.

2.1 Software

Pode-se definir o software, numa forma clássica, como sendo:

> Um conjunto de instruções que, quando executadas, produzem a função e o desempenho desejados, estruturas de dados que permitam que as informações relativas ao problema a resolver sejam manipuladas adequadamente e a documentação necessária para um melhor entendimento da sua operação e uso. (Pressman, 1995)

Outra definição para o Software:

> O software é a parte programável de um sistema de informática. Ele é um elemento central: realiza estruturas complexas e flexíveis que trazem funções, utilidade e valor ao sistema. Mas outros componentes são indispensáveis: as plataformas de hardware, os recursos de comunicação de informação, os documentos de diversas naturezas, as bases de dados e até os procedimentos manuais que se integram aos automatizados. (Pádua Filho, 2003)

Entretanto, no contexto da Engenharia de Software, o software deve ser visto como um produto a ser "vendido". É importante dar esta ênfase, diferenciando os "programas" que são concebidos num contexto mais restrito, onde o usuário ou "cliente" é o próprio autor. No caso destes programas, a documentação associada é

pequena ou (na maior parte das vezes) inexistente e a preocupação com a existência de erros de execução não é um fator maior, considerando que o principal usuário é o próprio autor do programa, este não terá dificuldades, em princípio, na detecção e correção de um eventual "bug". Além do aspecto da correção, outras boas características não são também objeto de preocupação como a portabilidade, a flexibilidade e a possibilidade de reutilização.

Um produto de software, por outro lado, é sistematicamente destinado ao uso por pessoas outras que os seus programadores. Os eventuais usuários podem, ainda, ter formações e experiências diferentes, o que significa que uma grande preocupação no que diz respeito ao desenvolvimento do produto deve ser a sua interface, reforçada com uma documentação rica em informações para que todos os recursos oferecidos possam ser explorados de forma eficiente. Ainda, os produtos de software devem passar normalmente por uma exaustiva bateria de testes, dado que os usuários não estarão interessados (e nem terão capacidade) de detectar e corrigir os eventuais erros de execução.

2.1.1 Processo de desenvolvimento de Software

Um processo de desenvolvimento de software é um conjunto de atividades, parcialmente ordenadas, com a finalidade de obter um produto de software. É estudado dentro da área de Engenharia de Software, sendo considerado um dos principais mecanismos para se obter software de qualidade e cumprir corretamente os contratos de desenvolvimento, sendo uma das respostas técnicas adequadas para resolver a Crise do software. Os passos a seguir demonstram como é feito o planejamento e o processo de desenvolvimento: (www.inf.ufes.br)

a) Análise Econômica

Visa a estabelecer se o projeto de Software gerará lucro, e se a receita gerada será o suficiente para cobrir os custos. Este processo acompanha todas as demais etapas de desenvolvimento do Software.

b) Análise de requisitos de software

A extração dos requisitos de um desejado produto de software é a primeira tarefa na sua criação. Embora o cliente, provavelmente, acredite saber o que o software deva fazer, esta tarefa requer habilidade e experiência em engenharia de software para reconhecer a incompletude, ambigüidade ou contradição nos requisitos.

c) Especificação

A especificação é a tarefa de descrever precisamente o software que será escrito, preferencialmente de uma forma matematicamente rigorosa. Na prática, somente especificações mais bem sucedidas foram escritas para aplicações bem compreendidas e afinadas que já estavam bem desenvolvidas, embora sistemas de software de missão crítica sejam freqüentemente bem especificados antes do desenvolvimento da aplicação. Especificações são mais importantes para interfaces externas que devem permanecer estáveis.

d) Arquitetura de Software

A arquitetura de um sistema de software remete a uma representação abstrata daquele sistema. Arquitetura é concernente à garantia de que o sistema de software irá ao encontro de requisitos do produto, como também assegurar que futuros requisitos possam ser atendidos. A etapa da arquitetura também direciona as interfaces entre os sistemas de software e outros produtos de software, como também com o hardware básico ou com o sistema operacional.

e) Implementação (ou codificação)

A transformação de um projeto para um código deve ser a parte mais evidente do trabalho da engenharia de software, mas não necessariamente a sua maior porção.

f) Teste

Teste de partes do software, especialmente onde tenha sido codificado por dois ou mais engenheiros trabalhando juntos, é um papel da engenharia de software.

g) Documentação

Uma importante tarefa é a documentação do projeto interno do software para propósitos de futuras manutenções e aprimoramentos. As documentações mais importantes são das interfaces externas.

h) Suporte e Treinamento de Software

Uma grande porcentagem dos projetos de software falham pelo fato de o desenvolvedor não perceber que não importa quanto tempo a equipe de planejamento e desenvolvimento irá gastar na criação do software se ninguém da organização irá usá-lo. As pessoas ocasionalmente resistem à mudança e evitam aventurar-se em áreas pouco familiares. Então, como parte da fase de desenvolvimento, é muito importante o treinamento para os usuários de software mais entusiasmados, alternando o treinamento entre usuários neutros e usuários favoráveis ao software. Usuários irão ter muitas questões e problemas de software os quais conduzirão para a próxima fase.

i) Manutenção

A manutenção e melhoria de software lidam com a descoberta de novos problemas e requisitos. Ela pode tomar mais tempo que o gasto no desenvolvimento inicial do mesmo. Não somente pode ser necessário adicionar códigos que combinem com o projeto original, mas determinar como o software trabalhará em algum ponto depois da manutenção estar completa, pode requerer um significativo esforço por parte de um engenheiro de software. Cerca de ⅔ de todos os engenheiros de software trabalham com a manutenção, mas estas estatísticas podem estar enganadas. Um pequena parte destes trabalha na correção de erros. A

maioria das manutenções é para ampliar os sistemas para novas funcionalidades, as quais, de diversas formas, podem ser consideradas um novo trabalho. Analogamente, cerca de ⅔ de todos os engenheiros civis, arquitetos e construtores trabalham com manutenção de uma forma similar.

2.2 Qualidade

A qualidade de software é uma área de conhecimento da engenharia de software que objetiva garantir a qualidade do software através da definição e normatização de processos de desenvolvimento. Apesar dos modelos aplicados na garantia da qualidade de software atuarem principalmente no processo, o principal objetivo é garantir um produto final que satisfaça às expectativas do cliente, dentro daquilo que foi acordado inicialmente. Existem diversas formas de definir qualidade. Segundo José Barreto Junior (Unemat, www.unemat.br), na tentativa de explicar esse conceito de forma simples e objetiva, obteve-se frases como:

- Qualidade é estar em conformidade com os requisitos dos clientes
- Qualidade é antecipar e satisfazer os desejos dos clientes
- Qualidade é escrever tudo o que se deve fazer e fazer tudo o que foi escrito

Segundo a norma ISO 9000 (versão 2000), a qualidade é o grau em que um conjunto de características inerentes a um produto, processo ou sistema cumpre os requisitos inicialmente estipulados para estes.

Eis algumas definições sobre qualidade:

• “*Qualidade* é uma característica intrínseca e multifacetada de um produto” (BASILI, 1996).

• “A relevância de cada faceta pode variar com o contexto e ao longo do tempo, pois as pessoas podem mudar seus posicionamentos e atualizar seus referenciais, com relação a um objeto ou a uma questão. Portanto, a *qualidade* não é absoluta, mas depende da perspectiva do avaliador. Como tal, qualquer medida de

qualidade deve ser subjetiva, sumarizando as impressões de alguma classe particular de indivíduos, que interajam com o produto.” (GENTLEMAN, 1997)

- “*Qualidade* é um conceito complexo, porque possui significados diversos para diferentes pessoas, em um contexto altamente dependente. Portanto, não é trivial haver medidas simples de *qualidade* aceitáveis para todos. Para estimar ou melhorar a *qualidade* de *software* numa organização, deve-se definir as características de *qualidade*, que se está interessado e, então, decidir como serão medidas” (KITCHENHAM, 1996).

- "*Qualidade* é ausência de deficiências, ou seja, quanto menos defeitos, melhor a *qualidade*" (JURAN, 1992).

- "*Qualidade* é a correção dos problemas e de suas causas ao longo de toda a série de fatores relacionados com marketing, projetos, engenharia, produção e manutenção, que exercem influência sobre a satisfação do usuário" (FEIGENBAUM, 1994).

- "*Qualidade* é a conformidade do produto às suas especificações." (CROSBY, 1986). As necessidades devem ser especificadas, e a *qualidade* é possível quando essas especificações são obedecidas sem ocorrência de defeito.

- "*Qualidade* é tudo aquilo que melhora o produto do ponto de vista do cliente" (DEMING, 1990). Deming associa *qualidade* à impressão do cliente, portanto não é estática. A dificuldade em definir *qualidade* está na renovação das necessidades futuras do usuário em características mensuráveis, de forma que o produto possa ser projetado e modificado para dar satisfação por um preço que o usuário possa pagar.

- "*Qualidade* é desenvolver, projetar, produzir e comercializar um produto de *qualidade* que é mais econômico, mais útil e sempre satisfatório para o consumidor" (ISHIKAWA, 1993).

A qualidade faz parte do dia a dia e desempenha papel importante em todos os tipos de organização. Durante muito tempo, os testes de softwares eram efetuados pelos próprios desenvolvedores, o que são conhecidos como testes

unitários. As informações extraídas destes testes, não permitiam que todos os defeitos fossem detectados. Muitas vezes os próprios usuários eram também envolvidos para aprovar o resultado desses testes ou mesmo participar da criação dos dados de teste. Isso gera uma incidência muito grande de defeitos, que apareciam quando os sistemas já estavam em produção e sua correção costuma ser muito mais cara. (Bastos et. al., 2007)

Isso levou as organizações a procurar novos caminhos para melhorar a qualidade do software desenvolvido. Um desses caminhos foi o aprimoramento da atividade de teste, através de um processo paralelo, contando com técnicos especializados, que pudesse suportar os projetos de teste, que, por sua vez, estavam vinculados aos projetos de desenvolvimento. Isso tem trazido resultados imediatos sobre a qualidade do produto entregue.

Cada vez mais, empresas têm adotado esse processo, que significa, num curto prazo, também a redução de custos com manutenção dos softwares.

Segundo Kruchten (2007 apud Bastos et. al., p. 11), existem pelo menos três dimensões que precisam ser consideradas antes de se iniciar qualquer ciclo de testes: Confiança, Funcionalidade e Performance.

2.2.1 Certificação de Qualidade

Um aspecto interessante da qualidade é que não basta que ela exista. Ela deve ser reconhecida pelo cliente. Por causa disso, é necessário que exista algum tipo de certificação oficial, emitida com base em um padrão. A Certificação em uma norma ou padrão é a emissão de um documento oficial indicando a conformidade com esta determinada norma ou padrão. É claro que, antes da emissão do certificado, é preciso realizar todo um processo de avaliação e julgamento de acordo com uma determinada norma. Embora uma empresa possa se auto-avaliar ou ser avaliada por seus próprios clientes, o termo Certificação costuma ser aplicado apenas quando efetuado por uma empresa independente e idônea, normalmente especializada neste tipo de trabalho. No Brasil, o INMETRO é o órgão do governo

responsável pelo credenciamento destas instituições que realizam a certificação de sistemas de qualidade. (Marini, Marcelo Jean, 18, 2002)

2.3 Qualidade de Software

Os softwares devem atender a certos atributos de qualidade e, para isso, é preciso identificar seus objetivos ou requisitos de qualidade, que complementam os requisitos funcionais, de desempenho, de custos e de cronograma, normalmente especificados para o desenvolvimento do software.

Na década de 60 e 70, a noção de qualidade de software estava ligada a parte interna do software, ou seja, era avaliada em termos de linhas de código de programa e tempo de execução. Com o advento dos microcomputadores, a disseminação das redes de micros pelas empresas e a participação maior do usuário no desenvolvimento do software, a noção de qualidade incorporou aspectos de satisfação do usuário; qualidade do processo de desenvolvimento do software e usabilidade e manutenibilidade do software. A etapa observada nos anos 80 envolve o desmantelamento dos tradicionais centros de processamento de dados centralizados e uma maior qualificação dos usuários de software. Nos anos 90, a discussão de qualidade de software assume um caráter mais formal, com a apresentação das Normas ISO 9000-3 (ABNT, 1993) e 9126 (ABNT, 1994).

Esse período também é marcado pela terceirização dos serviços de desenvolvimento de software. Neste período, na literatura, são encontradas várias abordagens na questão da qualidade: a utilização das métricas para medir custo, escopo e complexidade dos sistemas (DeMarco, 1991), a avaliação dos fatores que afetam a qualidade e a relevância da fase de testes (Pressman, 1995).

2.3.1 Software Quality Assurance (SQA)

Segundo Adriano Serrano (imasters.uol.com.br/artigo),

> [...] SQA é o processo que garante a qualidade do desenvolvimento de softwares focalizando todas as etapas e artefatos produzidos pelo desenvolvimento. Assegurando que todos eles estejam em conformidade com as necessidades predefinidas.

De acordo com Pressman as atividades da Gestão de qualidade abrangem:

> [...](1) um processo de garantia de qualidade de software (Software Quality Assurance – SQA); (2) tarefas especificas de garantia de qualidade e controle de engenharia (incluindo revisões técnicas e estratégia de teste multicamada); (3) prática de engenharia de software efetiva (métodos e ferramentas); (4) controle de todos os produtos de trabalho de software e das modificações feitas neles; (5) um procedimento para garantir a satisfação de normas de desenvolvimento de software (quando aplicável) e (6) mecanismos de medição e relatório (Pressman, 2006, p. 577).

2.3.2 Modelo de Qualidade de Software

O objetivo dos modelos de qualidade é direcionar a melhoria contínua em todos os parâmetros da qualidade por um sistema de medição orientado a metas. Com isso, foram desenvolvidas normas específicas para garantir o desenvolvimento de softwares com padronização, normatização e regras, como é possível observar no quadro abaixo:

Norma	Comentário
ISO 9126	Características da qualidade de produtos de software
NBR 13596	Versão brasileira da ISO 9126
ISO 14598	Guias para a avaliação de produtos de software, baseados na utilização prática da norma ISO 9126
ISO 12119	Características de qualidade de pacotes de software (software de prateleira, vendido com um produto embalado)
IEEE P1061	Standard for Software Quality Metrics Methodology (produto de software)
ISO 12207	Software Life Cycle Process. Norma para a qualidade do processo de desenvolvimento de software
NBR ISO 9001	Sistemas de qualidade - Modelo para garantia de qualidade em Projeto, Desenvolvimento, Instalação e Assistência Técnica (processo)
NBR ISO 9000-3	Gestão de qualidade e garantia de qualidade. Aplicação da norma ISSO 9000 para o processo de desenvolvimento de software

NBR ISO 10011	Auditoria de Sistemas de Qualidade (processo)
CMM	Capability Maturity Model. Modelo da SEI (Instituto de Engenharia de Software do Departamento de Defesa dos EEUU) para avaliação da qualidade do processo de desenvolvimento de software. Não é uma norma ISO, mas é muito bem aceita no mercado.
SPICE ISO 15504	Projeto da ISO/IEC para avaliação de processo de desenvolvimento de software. Ainda não é uma norma oficial ISO, mas o processo está em andamento.

Tabela 1 – Algumas normas para Qualidade de Software

Fonte: www.unemat.br

2.3.3 International Organization for Standardization (ISO)

A Organização Internacional para Padronização ou Organização Internacional de Normalização, popularmente conhecida como ISO é uma entidade que atualmente congrega os grêmios de padronização/normalização de 170 países.

Fundada em 23 de Fevereiro de 1947, em Genebra, na Suíça, a ISO aprova normas internacionais em todos os campos técnicos, exceto na eletricidade e eletrônica, cuja responsabilidade é da International Electrotechnical Commission (IEC), fundada em 1906. (www.iso.org)

A ISO 9000 trata da família "gestão da qualidade". Isto significa que a organização tem a cumprir:

- requisitos do cliente, de qualidade;
- requisitos regulamentares aplicáveis;
- aumentar a satisfação do cliente;
- conseguir a melhoria contínua do seu desempenho na busca destes objetivos.

A ISO tem como objetivo promover o desenvolvimento de normas, testes e certificação, com o intuito de encorajar o comércio de bens e serviços. No Brasil as normatizações ISO ficam por conta da ABNT (Associação Brasileira de Normas Técnicas).

2.3.3.1 ABNT NBR ISO 9001

A ABNT NBR ISO 9001 é a versão brasileira da norma internacional ISO 9001 que estabelece requisitos para o Sistema de Gestão da Qualidade (SGQ) de uma

organização, não significando, necessariamente, conformidade de produto às suas respectivas especificações. O objetivo da ABNT NBR ISO 9001 é prover a confiança de que o seu fornecedor poderá fornecer, de forma consistente e repetitiva, bens e serviços de acordo com o que foi especificado pelo cliente. A ABNT NBR ISO 9001 não especifica requisitos para bens ou serviços os quais estão sendo solicitados. Isto cabe ao cliente definir, tornando claras as suas próprias necessidades e expectativas para o produto. Sua especificação pode se dar através da referência a uma norma ou regulamento, ou mesmo a um catálogo, bem como a anexação de um projeto, folha de dados, etc.

Existem métodos que garantem que a empresa pelo qual se contrata um serviço atende às normas ABNT NBR ISO 9001. Segundo o site Visão Real (http://www.visaoreal.com.br/iso_9001.htm), esses métodos consistem em:

- Avaliação de segunda parte – a organização prestadora é avaliada diretamente pelo cliente com o objetivo justamente de verificar se, os serviços prestados atendem às normas ABNT NBR ISO 9001 e aos requisitos do seu cliente;
- Avaliação de terceira parte – a própria organização fornecedora contrata uma terceira parte imparcial, ou seja, uma entidade certificadora, preferencialmente credenciado pelo Inmetro, para avaliar a conformidade de seu Sistema de Gestão de Qualidade de acordo com a norma ABNT NBR ISO 9001.

A figura abaixo demonstra, de forma simples, o processo de conformidade à ABNT NBR ISO 9001:

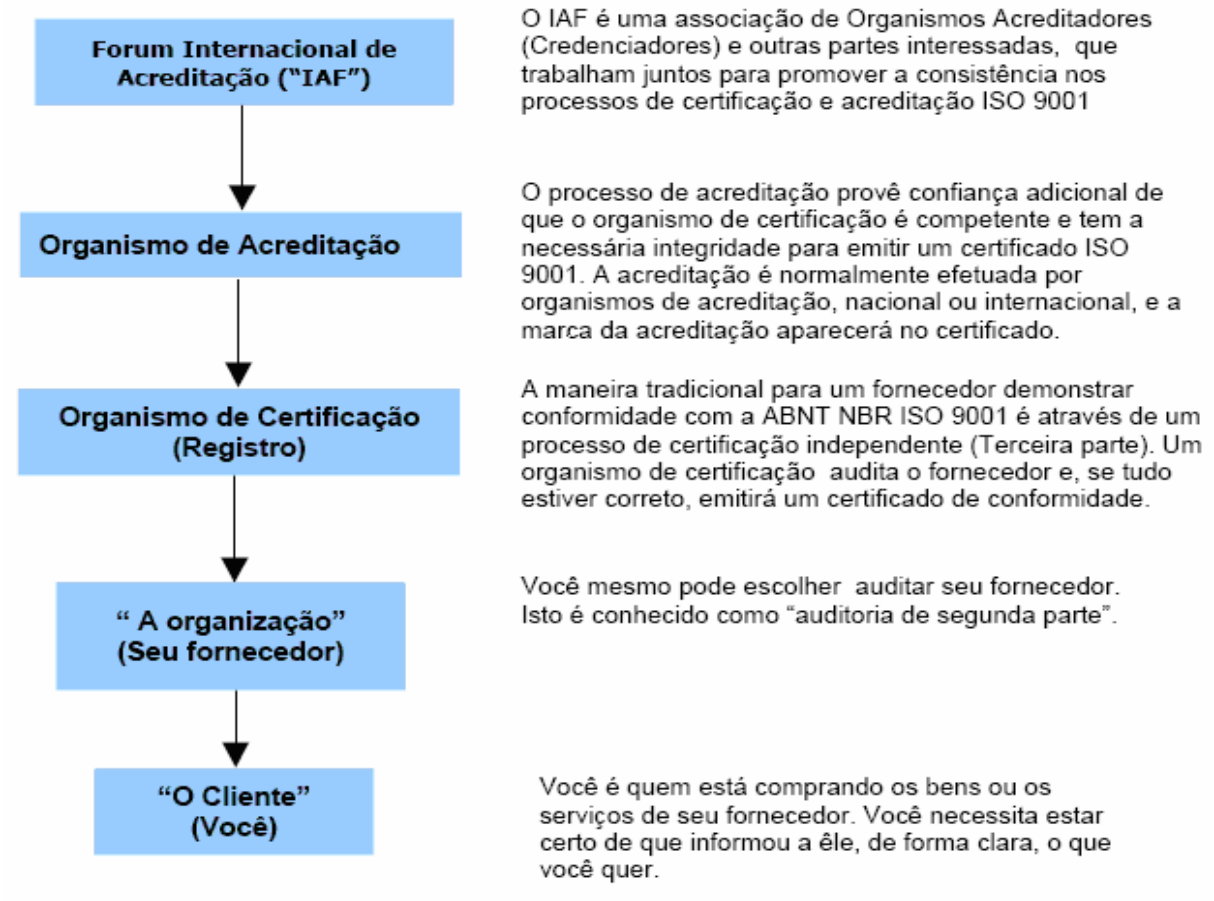


Figura 1 – Algumas formas de demonstrar conformidade à ABNT NBR 9001

Fonte: www.visaoreal.com.br/iso_9001.htm

2.3.3.2 ISO 12207

Processos de ciclo de vida de software. Esta norma estabelece uma estrutura comum para os processos de ciclo de vida de software, cobrindo desde a concepção até a retirada do software do mercado. A norma também provê um processo que pode ser utilizado para definir, controlar e melhorar os processos de ciclo de vida de software. As atividades que podem ser executadas durante o ciclo de vida de software estão agrupadas em cinco processos fundamentais, oito processos de apoio e quatro processos organizacionais:

- Processos fundamentais: aquisição, fornecimento, desenvolvimento, operação e manutenção.

- Processos de apoio: documentação, gerência de configuração, garantia da *qualidade*, verificação, validação, revisão conjunta, auditoria e resolução de problema.
- Processos organizacionais: gerência, infra-estrutura, melhoria e treinamento.

Processos Fundamentais	Início e execução do desenvolvimento, operação ou manutenção do software durante o seu ciclo de vida.
Aquisição	Atividades de quem um software. Inclui: definição da necessidade de adquirir um software (produto ou serviço), pedido de proposta, seleção de fornecedor, gerência da aquisição e aceitação do software.
Fornecimento	Atividades do fornecedor de software. Inclui preparar uma proposta, assinatura de contrato, determinação recursos necessários, planos de projeto e entrega do software.
Desenvolvimento	Atividades do desenvolvedor de software. Inclui: análise de requisitos, projeto, codificação, integração, testes, instalação e aceitação do software.
Operação	Atividades do operador do software. Inclui: operação do software e suporte operacional aos usuários.
Manutenção	Atividades de quem faz a manutenção do software.
Processos de Apoio	Auxiliam um outro processo.
Documentação	Registro de informações produzidas por um processo ou atividade. Inclui planejamento, projeto, desenvolvimento, produção, edição, distribuição e manutenção dos documentos necessários a gerentes, engenheiros e usuários do software.
Gerência de Configuração	Identificação e controle dos itens do software. Inclui: controle de armazenamento, liberações, manipulação, distribuição e modificação de cada um dos itens que compõem o software.
Garantia da Qualidade	Garante que os processos e produtos de software estejam em conformidade com os requisitos e os planos estabelecidos.
Verificação	Determina se os produtos de software de uma atividade atendem completamente aos requisitos ou condições impostas a eles.
Validação	Determina se os requisitos e o produto final (sistema ou software) atendem ao uso específico proposto.
Revisão Conjunta	Define as atividades para avaliar a situação e produtos de uma atividade de um projeto, se apropriado.
Auditoria	Determina adequação aos requisitos, planos e contrato, quando apropriado.
Resolução de Problemas	Análisar e resolução dos problemas de qualquer natureza ou fonte, descobertos durante a execução do desenvolvimento, operação, manutenção ou outros processos. .
Processos Organizacionais	Implementam uma estrutura constituída de processos de ciclo de vida e pessoal associados, melhorando continuamente a estrutura e os processos.
Gerência	Gerenciamento de processos.
Infra-estrutura	Fornecimento de recursos para outros processos. Inclui: hardware, software, ferramentas, técnicas, padrões de desenvolvimento, operação ou manutenção.
Melhoria	Atividades para estabeler, avaliar, medir, controlar e melhorar um processo de ciclo de vida de software.
Treinamento	Atividades para prover e manter pessoal treinado.

Tabela 2 – Processos da norma ISO 12207

Fonte: www2.unemat.br/rhycardo/download/qualidade_em_software.pdf

2.3.3.3 ISO 15504

A ISO/IEC 15504, também conhecida como SPICE (Software Process Improvement and Capability Determination), é a norma que define o processo de desenvolvimento de software. Na realidade, trata-se de uma evolução da ISO/IEC 12207, porém possui níveis de capacidade para cada processo assim como o CMMI. Em outubro de 2003, a Norma ISO/IEC 15504 (SPICE) para a avaliação de processos de software foi oficialmente publicada pela ISO. Tal norma define um modelo bi-dimensional que tem por objetivo a realização de avaliações de processos de software com o foco na melhoria dos processos (gerando um perfil dos processos, identificando os pontos fracos e fortes, que serão utilizados para a elaboração de um plano de melhorias) e a determinação da capacidade dos processos viabilizando a avaliação de um fornecedor em potencial. Esta norma está sendo desenvolvida desde 1993 pela ISO em conjunto com a comunidade internacional através do projeto SPICE com base nos modelos já existentes como ISO 9000 e CMM. Segundo a norma, uma avaliação de processo de software é uma investigação e análise disciplinada de processos selecionados de uma unidade organizacional em relação a um modelo de avaliação de processo. A ISO/IEC 15504 define um modelo de referência de processo que identifica e descreve um conjunto de processos considerados universais e fundamentais para a boa prática da engenharia de software, e define seis níveis de capacidade, seqüenciais e cumulativos que podem ser utilizados como uma métrica para avaliar como uma organização está realizando um determinado processo e também podem ser utilizados como um guia para a melhoria. A ISO/IEC 15504 define também um guia para a orientação da melhoria de processo, tendo como referência um modelo de processo e como uma das etapas a realização de uma avaliação de processo. Este guia sugere 8 etapas seqüenciais, que inicia com a identificação de estímulos para a melhoria e o exame das necessidades da organização. Em seguida existem ciclos de melhoria, nos quais um conjunto de melhoria são identificadas, uma avaliação das práticas correntes em relação à melhoria é realizada, um planejamento da melhoria é feito, seguido pela implementação, confirmação,manutenção e acompanhamento da melhoria.

2.3.4 Capability Maturity Model Integration (CMMI)

O CMMI foi lançado em 2001 pelo Instituto de Engenharia de Software da universidade de Carnegie-Mellon (SEI/CMU) e se define como uma integração e evolução do certificado CMM. Derivado principalmente dos modelos CMM – for Software (SW) e for Systems Engineering (SE), o grande diferencial para o modelo antigo é que no CMMI também é abordada a integração de engenharia de sistemas. Sua adoção é cada vez maior nas empresas que necessitam aumentar a produtividade e eliminar os defeitos no fim do desenvolvimento do software.

O CMMI (Capability Maturity Model Integration) é um modelo de referência que fornece orientação para o desenvolvimento de processos de softwares e tem como objetivos eliminar suas inconsistências; aumentar sua clareza e entendimento; fornecer uma terminologia comum e um estilo consistente; estabelecer regras de construção uniformes e assegurar consistência com a ISO/IEC 15504. Como seus antecessores, o CMMI não define como o processo deve ser implementado, mas prescreve suas características estruturais e semânticas em termos de objetivos e de grau de qualidade com que o trabalho deve ser realizado. O CMMI constitui tanto um modelo de capacidade como um modelo de maturidade. O modelo dentro de uma empresa pode ser alcançado em etapas consecutivas, representando a idéia de maturidade ou também de maneira contínua, onde são mensuradas a capacidade em práticas individuais. Quando uma organização atinge um nível de maturidade, considera-se que seus processos alcançaram uma determinada capacidade, ou seja, tem mecanismos que garantem a repetição sucessiva de bons resultados futuros relacionados principalmente à qualidade, custos e prazos.

O CMMI é dividido em cinco estágios:

1. Executado
2. Gerenciado
3. Definido
4. Quantitativamente Gerenciado
5. Otimização

ESTÁGIO	COMPETÊNCIAS
1- EXECUTADO	Estágio inicial – completa falta de planejamento e controle dos processos. Os funcionários estão focados basicamente em atividades corretivas que surgem a todo momento.
2-GERENCIADO	São estabelecidos processos básicos de gerenciamento de projeto para planejar e acompanhar custos, prazos e funcionalidades. Compromissos são firmados e gerenciados. A disciplina de processo permite repetir sucessos de projetos anteriores em aplicações similares. Tipicamente, possui gerenciamento de projetos estabelecido; alguns procedimentos técnicos escritos; acompanhamento de qualidade; gerência de configuração inicial; atividades básicas de medição e análise. O sucesso depende basicamente do gerenciamento do projeto.
3-DEFINIDO	Atividades de gerenciamento básico e as de Engenharia de Software são documentadas, padronizadas e integradas em processos-padrão. Todos os projetos de desenvolvimento ou manutenção de softwares utilizam uma versão de um desses processos adaptada às características especificas de cada projeto. Possui processos gerencias e técnicos bem definidos, possibilidade de avaliação do processo; ferramentas e metodologias padronizadas; medições iniciais de desempenho; inspeções e auditorias rotineiras; testes padronizados; gerência de configuração; evolução controlada dos processos técnicos e gerenciais.
4-QUANTITATIVAMENTE GERENCIADO	Métricas detalhadas do processo de software e da qualidade do produto são coletadas. Tanto o processo como o produto de software são quantitativamente compreendidos, avaliados e controlados. Relatórios estatísticos são gerados. Tipicamente, encontra-se estabelecido e em uso rotineiro um programa de medições, a qualidade é planejada por um grupo dedicado, sendo rotineiramente avaliada e aprimorada
5-OTIMIZAÇÃO	A melhoria contínua do processo é estabelecida por meio de sua avaliação quantitativa e da implantação planejada e controlada de tecnologias e idéias inovadoras. Projetos-piloto são realizados para a absorção e internalização de novas tecnologias. Tipicamente, um alto nível de qualidade e de satisfação dos clientes é alcançado rotineiramente, com grande foco na melhoria contínua.

Tabela 3 – Quadro de Competências por Estágio CMMI

Fonte: http://kplus.cosmo.com.br/materia.asp?co=30&rv=Vivencia

3. PROCESSO DE TESTE DE SOFTWARE

"O teste consiste em executar o programa com a intenção de encontrar erros (bugs)" (Myers, 1979)

Segundo Beizer, 1990 (www.macoratti.net)

> Há um mito segundo o qual, se fôssemos realmente bons para programar, não haveria 'bugs' a ser procurados. Se pudéssemos realmente nos concentrar, se todos usassem programação estruturada, projeto 'top-down', tabelas de decisão, se tivéssemos as balas de prata certas, então não haveria 'bugs'.

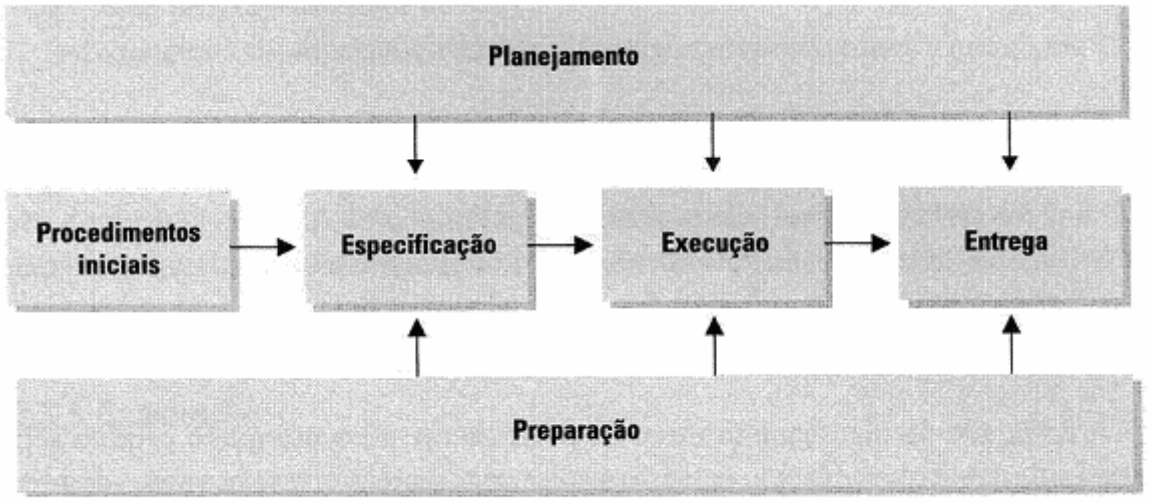
Para DeMarco, Tom, 1991 (www.macoratti.net)

> É de conhecimento geral entre os analistas de software que nunca se elimina o último bug de um programa. Os bugs são aceitos como uma triste realidade. Esperamos eliminá-los todos, um por um, mas nunca conseguiremos nos livrar deles.

Um importante aporte feito por Morell (1987), é que o teste, por ser uma forma de verificação de conformidade e qualidade do produto, não pode ser feito sem a sua comparação com especificações detalhadas. Segundo o autor, esta seria a única forma de avaliar corretamente a qualidade do software.

3.1 Ciclo de vida do processo de teste

O ciclo de vida do processo de teste é composto por diversas etapas ou fases, sendo quatro delas sequenciais ou em cascata e duas paralelas (Rios; Moreira, 2003, p. 44). A figura 2 exemplifica esse ciclo, conhecido como Modelo 3P X 3E.



Fonte: RIOS, E. & MOREIRA, T. *Teste de software*. Rio de Janeiro, Alta Books, 2003.

Figura 2 – Modelo 3P X 3E do ciclo de vida do processo de teste

3.1.1 Procedimentos inicias

Nesta etapa deverá ser aprofundado um estudo dos requisitos do negócio que dará origem ao sistema de informação a ser desenvolvido, de modo a garantir que o mesmo esteja completo e sem nenhuma ambigüidade. Essa abordagem é restrita ao projeto de teste.

Deve ser elaborado um plano com todas as atividades que serão executadas – se possível, com as necessidades de recurso de pessoal e de ambiente.

3.1.2 Planejamento

Consiste em elaborar a Estratégia de Teste e o Plano de teste a ser utilizados de modo a minimizar os principais riscos do negócio e fornecer os caminhos para as próximas etapas. Para cumprir esses objetivos, algumas diretrizes precisam ser consideradas:

- Esta etapa deve ser executada em conjunto com as atividades de captação dos requisitos e o planejamento do projeto do desenvolvimento do sistema a ser testado.

- Testes de verificação deverão ser executados sobre os requisitos do sistema, para minimizar inconsistências, faltas e incorreções.
- Deverá também ser preparada a análise de riscos do projeto de testes.

3.1.3 Preparação

Seu objetivo básico é preparar o ambiente de teste (equipamentos, pessoal, ferramentas de automação, hardware e software), para que os testes sejam executados corretamente. Essa fase ocorre em paralelo com as outras etapas.

3.1.4 Especificação

Os objetivos básicos dessa etapa são a elaboração / revisão dos casos de teste e a elaboração / revisão dos roteiros de teste.

No cumprimento dos objetivos é preciso que se observe o seguinte: os casos de teste e os roteiros de teste devem ser elaborados dinamicamente durante o decorrer do projeto de teste. Isso significa que devem ser elaborados à medida que a equipe de desenvolvimento liberar alguns módulos ou partes dos sistemas para teste.

3.1.5 Execução

Executar os testes planejados e registrar os resultados obtidos são tarefas que precisam obedecer às diretrizes a seguir:

- Os testes deverão ser executados de acordo com os casos de teste e os roteiros de teste.
- Devem ser usados os scripts de teste, caso seja empregada alguma ferramenta de automação de testes.

- Os testes devem ser executados integralmente, por regressão ou parcialmente, sempre que surgir alguma mudança de versão dos programas em teste e nos ambientes de teste preparados (desenvolvimento, testes, homologação, produção), conforme visto na estratégia e nos planos de teste.

3.1.6 Entrega

Nesta etapa, o projeto de teste é finalizado. Será concluída/arquivada toda sua documentação e serão relatadas todas as ocorrências desse projeto que forem consideradas relevantes à melhoria do projeto.

3.2 Conceito “V” de Teste

“Os benefícios advindos da adoção do modelo em ‘V’ são claros: detecção precoce de defeitos, maior envolvimento do time de testes no início do projeto e aumento da qualidade do software” (Richter, Iris, 2008).

O ciclo de vida de testes pressupõe que sejam realizados testes ao longo de todo o processo de desenvolvimento. Em determinados pontos, os produtos intermediários do ciclo de desenvolvimento devem ser revisados com o objetivo de criar as condições necessárias para uma correta implementação, procurando-se identificar defeitos o mais cedo possível.

Segundo Bastos (2006, p. 40), as atividades do ciclo de vida de testes só podem ser bem realizadas se forem executadas por um grupo formalmente definido para fazer os testes. Esse grupo pode ser:

- Interno: formado por profissionais pertencentes ao projeto de desenvolvimento de software ou por profissionais de uma área especialmente criada para exercer as funções de teste para todos os projetos de desenvolvimento do software.

- Externo: pertencente a uma empresa externa e formado por profissionais especializados em testes.

No conceito “V” de teste, os procedimentos de fazer e conferir convergem do início até o fim do projeto, conforme figura 3. O grupo que faz trabalha com o objetivo de implementar o sistema, e a equipe que confere, executa, simultaneamente, os procedimentos de teste visando minimizar ou eliminar os riscos.

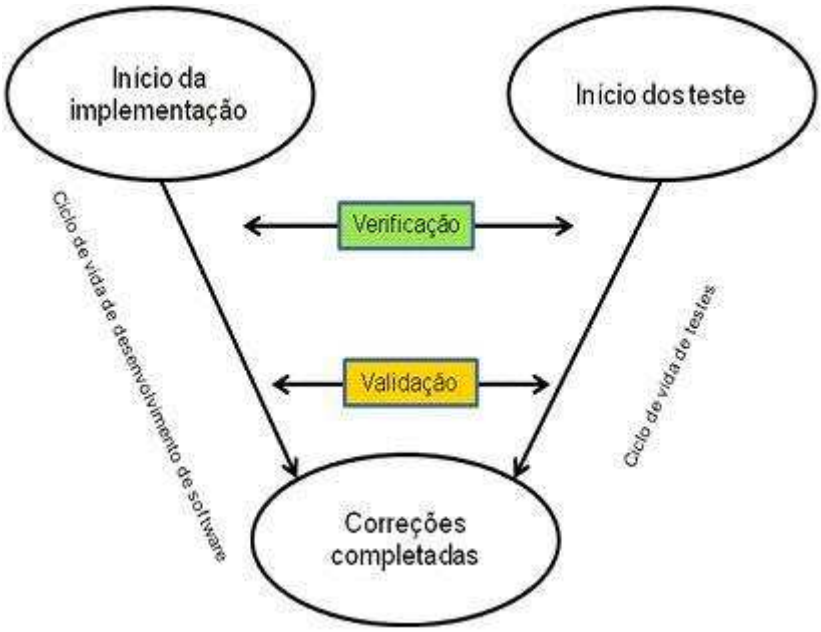


Figura 3 – Conceito “V” de teste de software
Fonte: Base de Conchecimento em teste de software, 2007, p.42

3.3 Processo de teste de software

“A validação é bem sucedida quando o software funciona de uma maneira razoavelmente esperada pelo cliente” (Pressman, 1995).

Segundo Anderson Bastos, no livro Base de Conhecimento em teste de software, o processo de teste de software segue o conceito “V” de teste. A parte esquerda do “V” representa o ciclo de desenvolvimento de software, e o lado direito, alguns dos passos do processo de teste de software.

A seguir, seguem algumas definições de cada passo, segundo o livro Base de Conchecimento em teste de software de Bastos, conforme figura 4.

1. Acesso ao Plano de Desenvolvimento: durante este passo, os testadores verificarão se o plano de desenvolvimento está completo e correto. Isso possibilita a estimativa de quantidade de recursos necessários para testar.
2. Desenvolvimento do Plano de Teste: a preparação do plano de teste segue os mesmos padrões da preparação do plano de desenvolvimento: a estrutura é a mesma, mas o conteúdo variará em função do grau de risco associado com o software que está sendo desenvolvido.
3. Inspeção ou teste dos requisitos de software: avaliação dos requisitos do software mediante o uso da técnica de verificação. Requisitos incompletos, inexatos ou inconsistentes levam ao insucesso de boa parte do desenvolvimento de software.
4. Inspeção ou teste do desenho do software: avaliação do desenho (interno e externo) do software através da técnica de verificação. O interesse da equipe de teste deve estar concentrado em verificar se o desenho atinge os objetivos dos requisitos, bem como se é eficaz e eficiente para operar no hardware previsto.
5. Inspeção ou teste da construção do software: o método escolhido para construir o software a partir do desenho do sistema determinará o tipo e a extensão dos testes que serão necessários.
6. Execução dos testes: envolve a abordagem, as ferramentas e os métodos especificados no Plano de Testes empregados para realizar a execução.
7. Teste de aceitação: avaliação da aplicabilidade e usabilidade do software pelos usuários. Além dos requisitos documentados, os usuários costumam testar outras funções não documentadas e suas expectativas. De modo geral os testes de aceitação devem ser orientados para avaliar se o software está apto a ser implantado com o nível de erros ainda não corrigidos.
8. Informação dos resultados dos testes: esse passo é um processo continuo, podendo ser verbal (não recomendado) ou escrito (formalizado), porém é importante que os defeitos e os tópicos envolvidos sejam relatados aos setores envolvidos o mais rápido possível, de modo que as correções sejam feitas com o menor custo.

9. Teste da instalação do software: visa verificar a interoperabilidade com o sistema operacional, com outros softwares relacionados e com os procedimentos operacionais. O resultado vai determinar se o software está ou não em condições de ser implantado no ambiente de produção.
10. Teste nas mudanças no software: embora seja considerado o décimo passo, as atividades dessa fase cobrem as mudanças durante o processo de implementação e aquelas que irão ocorrer após a implantação do software.
11. Avaliação da eficácia dos testes: as melhorias do processo de teste podem ser verificadas com maior exatidão pela avaliação da eficácia dos testes ao término de um projeto. Deve ser realizada pelos testadores, porém envolve os desenvolvedores, usuários e outros profissionais inseridos no processo de qualidade.

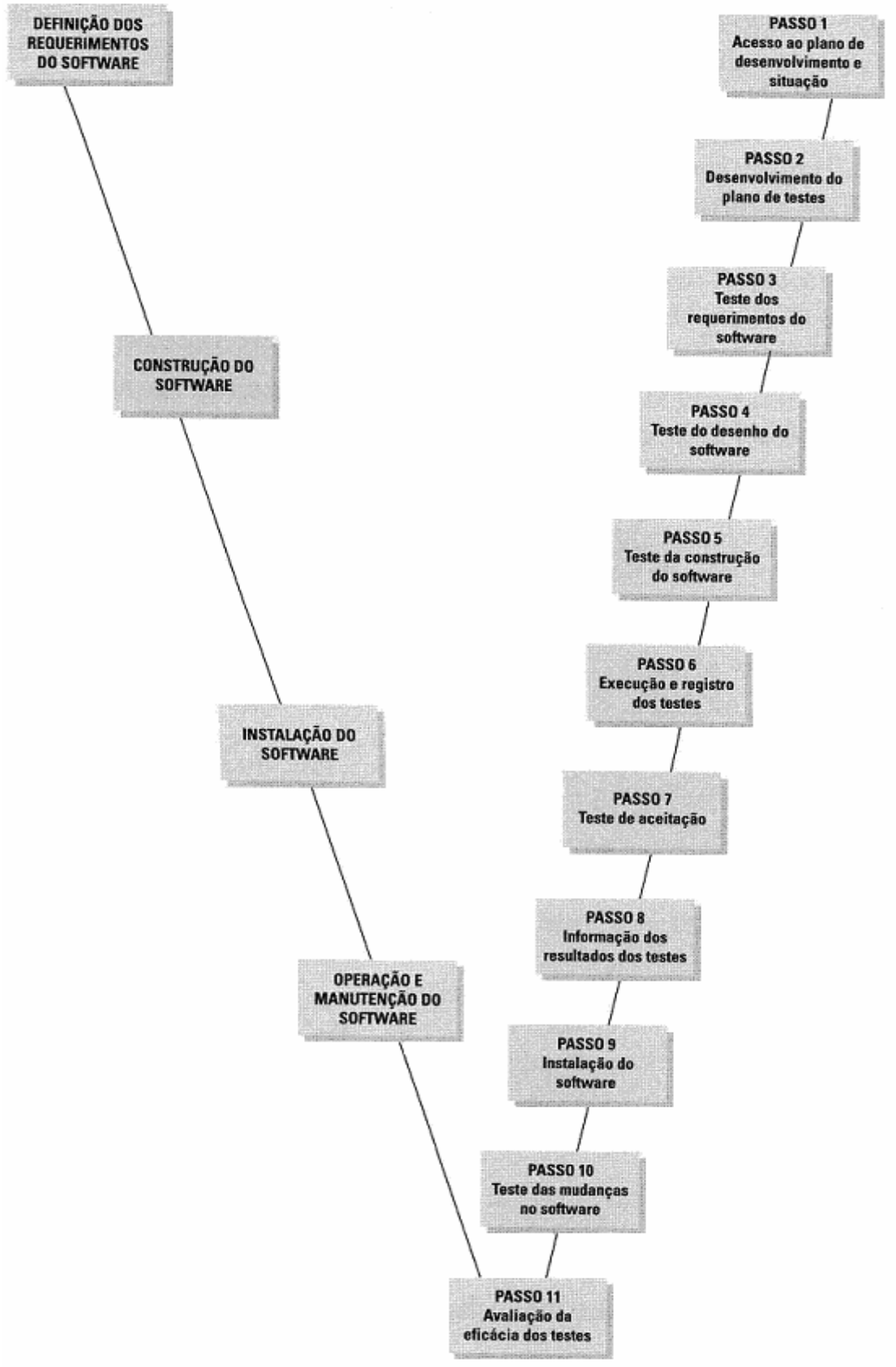


Figura 4 – Onze passos do processo de teste do software

Fonte: Base de Conchecimento em teste de software, 2007, p.42

3.4 Tipos de testes

Segundo Márcio Barros, existem alguns tipos de testes mais utilizados no ciclo de vida de testes, conforme listados abaixo:

- Testes de caixa-preta: geram casos de teste a partir das especificações definidas para os módulos, sem analisar seu código fonte;
- Testes de caixa branca: geram casos de teste a partir do código fonte do módulo que está sendo analisado;
- Testes de estrutura: geram casos de teste a partir de algum conhecimento sobre o projeto do sistema, como as estruturas de dados utilizadas;
- Teste de unidade: análise da funcionalidade provida por um módulo do sistema ou por um trecho de um módulo do sistema;
- Teste de integração (ou sistema): verifica se as unidades do sistema continuam operando de forma adequada quando são integradas. Normalmente é realizado executando-se os casos de teste após a integração dos sistemas;
- Teste de aceitação: determina se os resultados que o sistema está gerando são os resultados que o usuário está esperando. Geralmente é realizado com base em planos de teste gerados a partir dos casos de uso;
- Teste regressivo: execução repetida de um conjunto de casos de teste com o intuito de verificar se modificações causaram alterações inesperadas em um trecho de código previamente testado.

4. ENGENHARIA DE SOFTWARE

“O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais” (Fritz Bauer, 1969).

Segundo Vitório Bruno Mazzola (2003, p.6):

> Nos anos 40, quando se iniciou a evolução dos sistemas computadorizados, grande parte dos esforços, e conseqüentes custos, era concentrada no desenvolvimento do hardware, em razão, principalmente das limitações e dificuldades encontradas na época. À medida que a tecnologia de hardware foi sendo dominada, as preocupações se voltaram, no início dos anos 50, para o desenvolvimento dos sistemas operacionais, onde surgiram então as primeiras realizações destes sistemas, assim como das chamadas linguagens de programação de alto nível ou alta plataforma, como FORTRAN e COBOL, e dos respectivos compiladores. Já no início dos anos 60, com o surgimento dos sistemas operacionais com características de multiprogramação, a eficiência e utilidade dos sistemas computacionais tiveram um considerável crescimento, para o que contribuíram também, de forma bastante significativa, as constantes quedas de preço do hardware.

Ainda segundo o autor Vitório Bruno Mazzola:

> Uma conseqüência deste crescimento foi a necessidade, cada vez maior, de desenvolver grandes sistemas de software em substituição aos pequenos programas aplicativos utilizados até então. Desta necessidade, surgiu um problema nada trivial devido à falta de experiência e à não adequação dos métodos de desenvolvimento existentes para pequenos programas, o que foi caracterizado, ainda na década de 60 como a "crise do software", mas que, por outro lado, permitiu o nascimento do termo "Engenharia de Software".

Atualmente, apesar da constante queda dos preços dos equipamentos, o custo de desenvolvimento de software não obedece a esta mesma tendência. Pelo

contrário, corresponde a uma percentagem cada vez maior no custo global de um sistema informatizado. A principal razão para isto é que a tecnologia de desenvolvimento de software implica, ainda, em grande carga de trabalho, os projetos de grandes sistemas de software envolvendo em regra geral um grande número de pessoas num prazo relativamente longo de desenvolvimento. O desenvolvimento destes sistemas é realizado, na maior parte das vezes, de forma "ad-hoc" (expressão utilizada para designar ciclos completos de construção de softwares que não foram devidamente projetados em razão da necessidade de atender a uma demanda específica do usuário, ligada a prazo, qualidade ou custo), conduzindo a freqüentes desrespeitos de cronogramas e acréscimos de custos de desenvolvimento. (www.inf.ufsc.br)

Pensando nisso, foram criados os Modelos de Qualidade de Software que têm como objetivo garantir a qualidade do produto através da definição e normatização de processos de desenvolvimento. Os mais conhecidos são: ISO, CMM, CMMI.

Qualidade é um termo que faz parte do dia a dia e desempenha papel importante em todos os tipos de organização.

Isso levou as organizações a procurar novos caminhos para melhorar a qualidade do software desenvolvido. Um desses caminhos foi o aprimoramento da atividade de teste, através de um processo paralelo, contando com técnicos especializados, que pudesse suportar os projetos de teste, que, por sua vez, estavam vinculados aos projetos de desenvolvimento. Isso tem trazido resultados imediatos sobre a qualidade do produto entregue.

Cada vez mais, empresas tem adotado esse processo, que significa, num curto prazo, também a redução de custos com manutenção dos softwares.

Na Engenharia de Software, Qualidade é uma das áreas de conhecimento que tem como objetivo garantir a conformidade do software com as expectativas do cliente, através da definição e normatização de processos de desenvolvimento. Os modelos aplicados na garantia da Qualidade de Software atuam principalmente no processo e no produto.

Segundo Friedrich Ludwig Bauer, "Engenharia de Software é a criação e a utilização de sólidos princípios de engenharia a fim de obter software de maneira econômica, que seja confiável e que trabalhe eficientemente em máquinas reais".

Para Ricardo de Almeida Falbo, “A Engenharia de Software trata de aspectos relacionados ao estabelecimento de processos, métodos, técnicas, ferramentas e ambientes de suporte ao desenvolvimento de software”.

5. APLICAÇÃO DOS TESTES DE SOFTWARE NA INTEGRAÇÃO ENTRE DOIS BANCOS: ESTUDO DE CASO

5.1 Banco Beta

O Banco Beta* foi criado em 1925 e adquiriu 8 instituições financeiras no país entre 1934 e 1971. Em 1973, dois anos após transferir sua sede para São Paulo, a organização passou a adotar o nome que é utilizado até hoje.

Em 1998, um importante grupo adquiriu as operações do Banco Beta, além das aquisições dos bancos Bandepe (1998), Paraiban (2001) e Sudameris (2003). Em 2007 o consórcio formado pelos bancos Gama, RBS e Fortis adquiriu o ABN AMRO, controlador do Beta.

5.2 Grupo Gama

O Banco Gama* chegou ao Brasil no ano de 1957, por meio de um acordo operacional, porém teve as portas abertas ao público somente no ano de 1982, quando foi inaugurada sua primeira agência. Contudo, em 1991, o banco Gama decide se tornar um banco apenas de Investimentos. A seqüência de aquisições de vários e tradicionais bancos brasileiros, a partir dos anos 1990, também foi estrategicamente ponderada e planejada para levar o banco Gama a uma posição de grande destaque. Uma de suas maiores aquisições foi efetuada no ano de 2000, quando venceu no leilão de privatização de outro banco muito tradicional que contava, então, com mais de 570 agências e quase três milhões de clientes.

---

*Os nomes das empresas foram alterados, devido à política de confidencialidade da consultoria que disponibilizou essas informações.

Esses acontecimentos e outros levaram-no a se tornar uma das principais instituições financeiras do mercado nacional.

Modernização tecnológica - O banco Gama implementou grandes projetos para tornar-se uma corporação líder. No início deste ano, foi desencadeado o plano de renovação tecnológica para troca de sistemas, plataformas e equipamentos de informática.

Já em 2002, preocupado em satisfazer ainda mais seus clientes, foi iniciada a implantação de uma nova plataforma tecnológica cuja função era flexibilizar e agilizar a operação do sistema de informática e o lançamento de produtos.

Plano de atendimento com foco no cliente – o Grupo Gama iniciou projetos para aperfeiçoar os canais de contato, melhorar a qualidade dos serviços e estabelecer padrões de trabalho, buscando a excelência no atendimento. Com uma base de clientes diversificada, em função das várias aquisições realizadas, o maior desafio estava na conquista e fidelização deste grupo. A estratégia foi investir na formação dos funcionários e desenvolver produtos inovadores para atender às necessidades de todos os segmentos de clientes.

Em agosto de 2008, foi realizada a incorporação das ações do Banco Beta e respectivas empresas controladas, que passaram a fazer parte do banco Gama.

5.3 A Integração

5.3.1 Beta e Gama

O banco Gama e o Banco Beta juntos significam a união de duas culturas de extraordinário valor no mercado. Cada um, inovador a seu modo, conquistou a confiança e a preferência de seus clientes, trazendo aperfeiçoamentos e evoluções significativas tanto na eficiência da prestação de serviços financeiros como na atuação junto ao mercado e à sociedade.

Em março, uma nova etapa foi iniciada nos dois bancos: a primeira entrega da união do Banco Gama e Banco Beta ofereceu muito mais aos seus clientes e funcionários. Além de saques e consultas, cerca de 8 milhões de correntistas e 500

mil empresas atendidas pelo Grupo Gama já podem fazer pagamento de contas de concessionárias e títulos a vencer e/ou vencidos do Banco Gama, do Banco Beta e da Aymoré. E mais, além dos 18 mil caixas eletrônicos, os clientes contam também com 6 mil agências e postos de atendimento bancário em todo o País.

Orgulhosos da união feita em 2008 e concretizada agora no ano de 2009, o grupo Gama hoje disputa pelo posto de melhor e mais eficiente banco do País e esse primeiro passo demonstra que podem ser apresentadas soluções mais completas e com mais conveniência, facilidade e comodidade.

O lema utilizado para reforçar a eficiência da integração é justamente:

“Juntos, seremos ainda mais fortes”.

5.3.1.1 Serviços integrados disponíveis

O Grupo Gama teve que passar por um período de adaptação de seus sistemas para oferecer aos clientes do Banco Beta todos os serviços que lhes eram oferecidos antes da integração, com a mesma qualidade ou até melhor. Ainda existem diversos sistemas que estão sendo adaptados ou criados para atender seus novos clientes, uma vez que, por se tratar de diferentes bancos, podem ser oferecidos os mesmos serviços, mas de formas diferentes. Muitos sistemas já foram entregues e estão em utilização, contudo há outros serviços em desenvolvimento com data prevista para entrega para o final de 2009 e início de 2010.

Dentre estes, já estão disponíveis serviços como:

Caixas eletrônicos e Agências:

- Saque de conta corrente ou conta poupança;
- Consulta de saldo da conta corrente ou conta poupança;
- Extrato de conta corrente ou conta poupança;
- Pagamento de tributos e contas de concessionárias (água, luz, telefone, gás etc.);
- Pagamento de títulos vencidos do Grupo Gama (inclusive Aymoré);
- Pagamento de títulos a vencer do Grupo Gama e de outros bancos;

- Recarga de celular pré pago disponível, somente em caixas eletrônicos;
- Transferência entre contas (débito em conta do Banco Beta e crédito em conta do banco Gama ou vice-versa).

Em canais eletrônicos (Internet Banking, Superlinha, Correspondentes Bancários,entre outros):

- Pagamento de tributos e contas de concessionárias, vencidos e a vencer;
- Pagamento de títulos vencidos do banco Gama e do Banco Beta;
- Pagamento de títulos a vencer do banco Gama, do Banco Beta (inclusive Aymoré) e de outros bancos.

5.3.2 Débito Direto Autorizado (DDA)

O objetivo deste estudo de caso é demonstrar a eficiência do sistema de teste integrado para garantir a qualidade e o sucesso do projeto. O DDA foi um serviço no qual, entre as fases de desenvolvimento e homologação, foi investida uma fase de testes integrados para garantir a qualidade do sistema. Hoje, já em implantação, apesar de gerar grandes dúvidas entre seus usuários por se tratar de uma novidade, o serviço tem sido um sucesso.

5.3.2.1 Mas o que é o DDA?

O Débito Direto Autorizado é uma iniciativa da FEBRABAN (Federação Brasileira de Bancos) de um novo serviço disponível para todos os bancos, permitindo que os clientes recebam e paguem os seus boletos de cobrança de forma eletrônica.

5.3.2.2 Funcionamento

Os clientes cadastrados no DDA não mais receberão seus boletos de pagamento impressos. Estes serão apresentados de forma eletrônica nos canais de relacionamento do Banco, Internet Banking, Caixas Eletrônicos e Superlinha.

Um ponto importante em relação ao DDA: O cadastramento ao DDA não significa que as contas serão pagas automaticamente. Para que o pagamento seja realizado é necessário que o cliente acesse um dos canais de relacionamento do banco, verifique se há boletos a serem pagos e autorize ou agende o pagamento dos respectivos boletos.

Vantagens de se cadastrar no DDA

Ao se cadastrar ao DDA e concentrar seus pagamentos no Grupo Gama o cliente tem as seguintes vantagens:

- É possível consultar boletos e realizar pagamentos pala Internet, Telefone e Caixas Eletrônicos de qualquer lugar e sem a necessidade de ter em mãos o boleto em papel ou ir até uma agência.
- Através dos canais de relacionamento pode-se programar e realizar seus pagamentos até às 22hs.
- Estão disponíveis mais de 18.000 caixas dos bancos do Grupo Gama para realização pagamentos, saques, consultas, extratos, recarga de celulares e transferências.
- Ao optar pelo DDA o cliente corre menos risco de fraude e extravio do boleto, não precisa mais capturar ou digitar os dados do boleto e ainda contribui para a redução do uso de papel e para a preservação do meio ambiente.

5.3.2.3 A implantação do sistema

Durante o período de desenvolvimento do sistema que atende aos serviços prestados pelo DDA, uma consultoria multinacional, com grande experiência de

mercado e conhecida pela alta qualidade de seus serviços prestados, aderiu aos serviços de uma Fábrica de Testes, termo utilizado pela empresa por se tratar de uma fábrica que produz qualidade, para a execução de testes integrados.

Ainda, no processo de criação de novos sistemas e customização dos sistemas já existentes, a equipe especializada em testes integrados gerou uma documentação com 234 casos de teste para serem executados antes de entrar na fase de homologação. Do período de 01/06/2009 a 10/07/2009, esses testes foram executados dentro das instalações do Grupo Gama, onde foi possível acessar seu TFC (Terminal Financeiro Corporativo) e sua base de dados. Nesse período, foram investidas cerca de 160 horas de execução dos testes e retestes, pois os defeitos corrigidos deveriam ser retestados em paralelo às execuções.

O TFC trata-se do sistema utilizado pelos funcionários do banco para efetuar as transações solicitadas pelos clientes, tais como cadastros de convênios, clientes, fornecedores, cadastrar serviços, tal como o DDA, efetuar consultas, transferências de valores, abrir contas, etc.

Todas essas transações, entre muitas outras, foram disponibilizadas à equipe para executar os devidos testes.

Esses testes foram focados apenas no escopo dos sistemas dedicados à consultoria citada neste estudo de caso, uma vez que o DDA é constituído de diversas interfaces entre diversos sistemas.

Com a execução desses testes, pode-se analisar os seguintes resultados:

- 234 casos de teste para serem executados;
- 67 incidências (defeitos) abertas em 7 semanas, pela equipe de teste integrado, além dos erros encontrados nos testes unitários, executados pelos próprios desenvolvedores;
- 95% dos erros identificados puderam ser corrigidos devido ao prazo estipulado, antes de congelar o sistema para início da fase de homologação;

5.3.2.4 O Processo

Como já citado, dentro do escopo dos sistemas confiados à consultoria analisada neste estudo, foi elaborado um processo para execução dos testes. É gerado um documento pela equipe funcional do projeto, nomeado de Especificação Funcional, que informa todas as funcionalidades do sistema, dentre essas, constam informações desde o tipo de caracter que os campos do TFC e banco de dados devem aceitar até as interfaces que devem ser feitas entre o sistema do DDA e os demais sistemas (Contas Correntes, Personas, etc), tabelas de dados, gráficos, layouts, protótipos de interfaces, dentre outras informações.

A Especificação Funcional é entregue à equipe de testes e a partir deste momento é iniciado um estudo sobre o sistema a ser testado. Após análises e reuniões com a equipe funcional para um perfeito entendimento das funcionalidades a serem testadas, a equipe de testes começou a identificar os requisitos de testes.

A equipe de teste se baseia em um conceito de teste que pode ser melhor esclarecido através da seguinte figura:

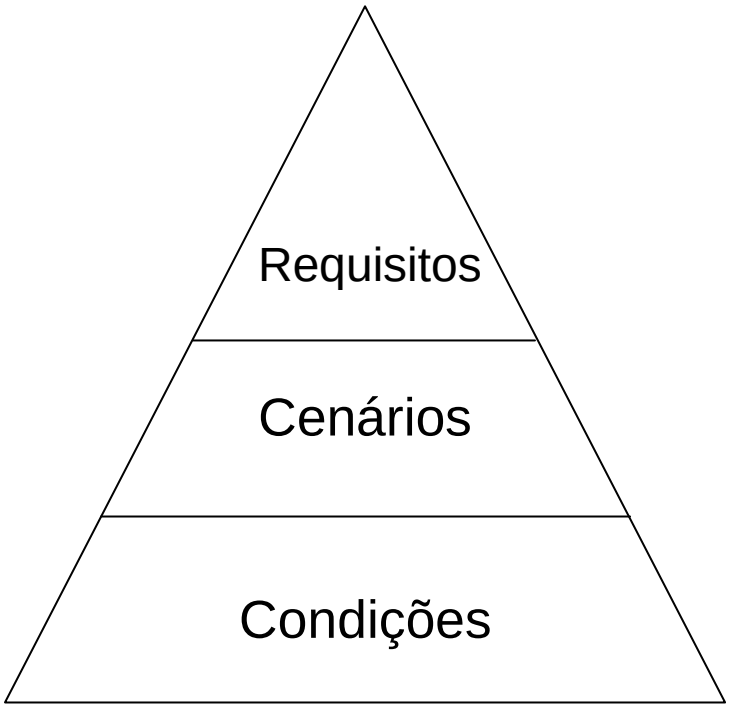


Figura 5 – Hierarquia do processo de testes

Fonte: Consultoria envolvida no Projeto

Requisitos: Os requisitos identificam o que será testado. Eles são o objetivo específico de um teste. Há algumas regras gerais que se aplicam à geração de requisitos de teste:

- O requisito de teste deve ser um comportamento mensurável e observável. Se não for possível observar nem medir o requisito de teste, ele não poderá ser avaliado para determinar se foi atendido.
- Não há um relacionamento um-para-um entre cada caso de uso ou requisito suplementar de um sistema e um requisito de teste. Geralmente, os casos de uso têm mais de um requisito de teste, enquanto alguns requisitos suplementares geram um ou mais requisitos de teste e outros não geram nenhum.

Cenários: Cenários de teste são descrições de alto nível das áreas funcionais e técnicas a serem testadas. Os cenários podem ser subdivididos em sub-cenários como necessário, até detalhados, condições de ser testadas e os resultados esperados podem ser determinados. Como resultado, cada cenário de teste (ou sub-cenário) é descrito por um número de condições de teste associados. Juntos, um conjunto de condições de teste torna-se uma lista de itens a ser usado para testar completamente o cenário de teste associados.

Condições: Condições de teste definem os testes a serem realizados e devem sempre combinar as fases de desenvolvimento que se referem e os requisitos e projetos que pretendem provar:

- Condições de ensaio de design de produto para testar os requisitos do produto.
- Condições de ensaio Design de montagem para testar os projetos.
- Condições de ensaio Design componente para testar as especificações detalhadas.

A condição de teste inclui as seguintes propriedades:

- Nome e descrição da condição de teste. Começa com um verbo de ação: criar, validar, verificar, etc.;
- Resultado esperado. Para cada condição de teste, definir um resultado correspondente esperado. O nível de resultados esperados criado durante o planejamento do teste pode ser relativamente elevado. Por

exemplo, um resultado esperado para a condição de teste “Criar Fatura”, pode ser simplesmente, uma fatura é criada. Enquanto se prepara os scripts de teste, o atual step (entrada de dados, apresentar fatura, etc) para executar a condição de teste é adicionado e os resultados esperados (dados introduzidos são validados, na fatura é enviada via e-mail, etc) são mais detalhados para cada etapa.

- Requisito ou fonte de design. Cada condição de teste é derivado de uma obrigação ou um item de design. Isso torna mais fácil para verificar se todos os requisitos e elementos de design são abrangidos e para controlar os defeitos que são detectados durante a execução do teste. Deve-se usar a Matriz de rastreabilidade de requisitos para cruzar as condições de teste de referência de volta para os requisitos e elementos de design.

Após a identificação dos requisitos de teste, são extraídas, destes mesmos requisitos, as condições de teste e é gerada uma documentação conhecida como plano de testes. Segue abaixo um exemplo de Matriz de Rastreabilidade.

ID #	Lista de Requisitos	ID #	Lista de Cenários	ID #	Lista de Scripts (Condições de Testes)
Requisitos Funcionais					
1.0.0	Validar Cadastro de Convênios				
		1.1.1	Validação do Parâmetro para Captura de Títulos (Cobrança)		
				1.1.1.1	Selecionar opção "Não Captura" como Parâmetro de Captura de Títulos e Perfil de autorização "Tem módulo de autorização do IB"
				1.1.1.2	Selecionar opção "Não Captura" como Parâmetro de Captura de Títulos e Perfil de autorização "Lista de Débito"
				1.1.1.3	Selecionar opção "Não Captura" como Parâmetro de Captura de Títulos e Perfil de autorização "Nenhum"
				1.1.1.4	Selecionar opção "Somente Envia Arquivo" como Parâmetro de Captura de Títulos e Perfil de autorização "Tem modulo de autorização do IB"
				1.1.1.5	Selecionar opção "Somente Envia Arquivo" como Parâmetro de Captura de Títulos e Perfil de autorização "Lista de Débito"
				1.1.1.6	Selecionar opção "Somente Envia Arquivo" como Parâmetro de Captura de Títulos e Perfil de autorização "Nenhum"
				1.1.1.7	Selecionar opção "Agenda Conforme Convênio" como Parâmetro de Captura de Títulos e Perfil de autorização "Tem módulo de autorização do IB"
				1.1.1.8	Selecionar opção "Agenda Conforme Convênio" como Parâmetro de Captura de Títulos e Perfil de autorização "Lista de Débito"
				1.1.1.9	Selecionar opção "Agenda Conforme Convênio" como Parâmetro de Captura de Títulos e Perfil de autorização "Nenhum"
				1.1.1.10	Selecionar opção "Agenda Bloqueado" como Parâmetro de Captura de Títulos e Perfil de autorização "Tem módulo de autorização do IB"

Figura 6 – Modelo de Matriz de Rastreabilidade
Fonte: Consultoria envolvida no Projeto

Como o sistema do DDA faz interface com diversos sistemas, as condições de teste criadas têm como foco situações em que o sistema reflete em outros

sistemas. Por exemplo, ao efetuar o cadastro de um cliente, são necessárias diversas parametrizações e preenchimento de diversas informações. No momento do cadastro, ao inserir o CPF do cliente (caso seja pessoa física) ou o CNPJ (para pessoa jurídica), o sistema do DDA deve fazer uma validação junto ao sistema de cadastro chamado Personas, para garantir a integridade da informação. Para casos como esse, foram testados todos os tipos de parametrização possíveis, validando sempre se o sistema responde às expectativas, no caso, chamadas de Resultado Esperado.

Abaixo alguns exemplos das condições criadas no plano de testes da consultoria mencionada:

1. Efetuar novo cadastro no sistema DDA para verificar se o sistema recebe arquivos dos sistemas XX e YY após um dia útil;
2. Efetuar novo cadastro no sistema DDA para verificar se o sistema recebe arquivos dos sistemas XX e YY após um dia de final de semana;
3. Consultar arquivo de títulos de cobrança de clientes com a opção "Não Acata e Somente Envia Arquivos" selecionada para verificar se foi enviado arquivo de títulos contra ele;
4. Verificar se o parâmetro de “Melhor Data” se aplica a todos os títulos capturados dos sistemas XX e YY;
5. Acessar cadastro de cliente e verificar se o sistema DDA envia arquivo com títulos do próprio e de outros bancos não agendados;
6. Efetuar agendamento de entrada de títulos vencidos;

Hoje, o serviço de Débito Direto Autorizado foi implantado com sucesso e já está disponível para atender os milhares de clientes do Grupo Gama.

6. CONCLUSÃO

No estudo de caso mencionado, verifica-se que a estratégia da empresa de consultoria em aplicar um processo de testes no projeto DDA paralelo ao processo de desenvolvimento/alteração, foi de extrema importância para garantir a qualidade dos sistemas que estavam sendo alterados e/ou criados para atender aos serviços solicitados. Segundo o estudo de caso analisado anteriormente, o resultado deste investimento foi simplesmente de 95% de erros identificados e corrigidos antes da implementação, além dos erros encontrados pelos próprios desenvolvedores durante os testes unitários.

Este trabalho foi desenvolvido com a finalidade de demonstrar a importância de entregar um sistema com qualidade. Além disso, durante este projeto foram definidos conceitos de algumas normas de qualidade, dentre as mais utilizadas, e também as técnicas de testes, tais como documentações geradas, quais tipos de testes são utilizados atualmente, onde se aplicam e quais os benefícios de investir nesta área.

Após diversas pesquisas e diante do que foi exposto acima, é possível concluir que a qualidade é fundamental na produção e entrega de um produto e quando se trata de um software, onde a probabilidade de ocorrerem erros é muito grande, os testes são grandes aliados.

REFERÊNCIAS

BASILI, Victor. **A validation of object-oriented design metrics as quality indicators, IEEE Transactions on Software Engineering**. 1996.

Bastos, Anderson et al. **Base de conhecimento em teste de software.** São Paulo: Martins, 2007.

Caetano, Cristiano. Comentários sobre o evento: O Conceito 'V' de Testes: O Segredo do Software de Qualidade - Porto Alegre, 27 jun. 2008. Disponível em: http://www.testexpert.com.br/?q=node/776. Acessado em: 12 nov. 2009.

FILHO, Wilson de Pádua Paula. **Engenharia de Software**. 2. ed.. (1º reimpressão - 2005). São Paulo: LTC, 2003.

Fórum da Qualidade de Software O Caminho Para A Excelência Do Software Brasileiro Passa Pela Qualidade, Disponível em: http://qualidadesoftware.org.br/forum/, 2009 Acessado em: 03 out. 2009.

GENTLEMAN, Morven. **If the software quality is a perception, how do we measure it? Quality of Numerical Software**. 1997.

Implementando o CMMI (Capability Maturity Mode Integration) como ferramenta para gerenciamento de projetos de Software, Disponível em: http://kplus.cosmo.com.br/materia.asp?co=30&rv=Vivencia, 2009. Acessado em: 03 out. 2009.

Instituto Brasileiro de Qualidade em Testes de Software, Disponível em: http://www.ibqts.com.br. Acessado em: 03 out. 2009.

Introdução à Qualidade de Software, 2007. Disponível em: http://www.inf.ufes.br/~falbo/download/aulas/tengsoft/2007-1/Aula1.ppt#355,5,O que é Qualidade?. Acessado em: 03 out. 2009.

ISHIKAWA, Kaoru. **Introduction to Quality Control.** New York, 1993.

KITCHENHAM B.; PFLEEGER, S. L. Software Quality**: The Elusive Target**. 1996.

Koscianski, A.; Soares, Michel dos Santos. **Qualidade de Software: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software**; São Paulo, 2007.

Modelos de Desenvolvimento de Software. Disponível em: http://inf.unisul.br/~vera/egs/aula01.htm. Acessado em 04 out. 2009.

Pressman, Roger S. **Engenharia de Software.** São Paulo: Makron Books, 1995.

Qualidade de Software, Disponível em: http://www2.unemat.br/rhycardo/download/qualidade_em_software.pdf. Acessado em: 03 out. 2009.

Qualidade de Software, Disponível em: http://www.slideshare.net/alsimoes/qualidade-de-software, 2009. Acessado em: 03 out. 2009.

Qualidade de Software, Disponível em: http://www2.dem.inpe.br/ijar/SWQ.htm. Acessado em: 03 out. 2009.

Qualidade de Software. Disponível em: http://www.pdf-search-engine.com/qualidade-de-software-html-www2.unemat.br/rhycardo/download/qualidade_em_software.html. Acessado em: 04 nov. 2009.

Software Quality Assurance. Disponível em: http://imasters.uol.com.br/artigo/1050/des_de_software/software_quality_assu, 2005. Acessado em: 15 nov. 2009.

Textos e Notas sobre a Engenharia de Software e Sistemas de Informação

Disponível em: http://www.cic.unb.br/~jhcf/MyBooks/iess/index.html, 2005. Acessado em 04 out. 2009.

Universidade Federal do Espírito Santo. Disponível em: http://www.inf.ufes.br/~falbo/download/aulas/es-g/2005-1/NotasDeAula.pdf. Acessado em: 03 out. 2009.

Universidade Federal de São Carlos. Modelo de Qualidade – CMMI. Disponível em: http://www.comp.ufscar.br/~bruno_abrahao/Artigos/CMMI.pdf, 2007. Acessado em: 03 out. 2009.