FUNDAÇÃO EDSON QUEIROZ

UNIVERSIDADE DE FORTALEZA – UNIFOR

Lívia Nojoza Amorim

GERENCIAMENTO DA QUALIDADE:

UMA NOVA DISCIPLINA PARA O RUP

Fortaleza

2005

FUNDAÇÃO EDSON QUEIROZ

UNIVERSIDADE DE FORTALEZA – UNIFOR

Lívia Nojoza Amorim

GERENCIAMENTO DA QUALIDADE: UMA NOVA DISCIPLINA PARA O RUP

Dissertação apresentada ao Curso de Mestrado em Informática Aplicada (MIA) da Universidade de Fortaleza (UNIFOR), como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Arnaldo Dias Belchior

Fortaleza

2005

Lívia Nojoza Amorim

GERENCIAMENTO DA QUALIDADE: UMA NOVA DISCIPLINA PARA O RUP

Data de Aprovação: _____

Banca Examinadora:

_____

Prof. Arnaldo Dias Belchior, D.Sc.
(Presidente da Banca)

_____

Prof[a]. Ana Cervigni Guerra, D.Sc.

_____

Prof[a]. Maria Elizabeth Sucupira Furtado, *Docteur*

FICHA DE CATALOGAÇÃO

AMORIM, Lívia Nojoza. **Gerenciamento da Qualidade: uma nova disciplina para o RUP.** 2005. 191f. Dissertação (Mestrado em Informática Aplicada) – Universidade de Fortaleza (UNIFOR). Fortaleza, 2005.

Perfil do Autor: Graduada em Bacharelado em Ciência da Computação pela Universidade Estadual do Piauí (UESPI). Graduanda em Administração de Empresas pela Universidade Federal do Piauí/Universidade Federal do Ceará (UFPI/UFC). Consultora de Tecnologia da Informação da Célula de Testes, Homologação e Qualidade de Software do Ambiente de Estratégias de Tecnologia do Banco do Nordeste do Brasil.

RESUMO:
Com a crescente dependência da sociedade em relação a produtos de *software*, há uma maior preocupação em como obter software de qualidade. As organizações têm adotado abordagens para garantir a qualidade de *software*, baseado em modelos de qualidade como CMMI, ou em processos de software, como o Processo Unificado (UP). O UP é um processo que objetiva garantir a produção de *software* de qualidade, atendendo as necessidades de usuários por meio de cronogramas e custos previsíveis. Este trabalho analisa a abordagem de qualidade de *software* do UP em relação a modelos e padrões de qualidade utilizados no mercado e propõe uma nova disciplina para o UP, denominada de Gerenciamento da Qualidade, com o objetivo de estabelecer ações que contribuam para a qualidade do processo de *software* do UP. A disciplina proposta visa suprir deficiências identificadas na abordagem de qualidade de *software* do UP, por meio da execução de atividades de garantia de qualidade em projetos, e está alinhada à ISO/IEC 12207 e ao CMMI. Essa disciplina foi aplicada em projetos de *software* em duas organizações produtoras de *software*, auxiliando a prevenir problemas na qualidade de processo e melhorando a qualidade do *software* desenvolvido.

PALAVRAS-CHAVE:
1) Engenharia de Software; 2) Qualidade de Software; 3) Gerenciamento da Qualidade 4) Garantia de Qualidade de Software (SQA) 5) Processo Unificado (UP)

DEDICATÓRIA

A todos aqueles que nunca tiveram nem nunca terão a oportunidade de estudar e aos que estiveram ao meu lado me apoiando para que este trabalho pudesse ser concluído.

AGRADECIMENTOS

Agradeço a Deus, por ter me auxiliado, de todos os modos, a iniciar e a concluir esse trabalho. Quando tudo era incerto, Deus se mostrou presença certa na minha vida.

Agradeço imensamente ao apoio da minha família que, mesmo de longe, foi fundamental para que esse trabalho pudesse ser levado à frente. Mãezinha, obrigada por tudo. Obrigada, Alexandre, Henrique e Roberto pela torcida e apoio. Pai, obrigada pelas orações. Bian, obrigada por ser minha segunda mãe.

À minha avó (agora *in memorian*), tios e primos pelas orações e incentivos durante esta empreitada.

Agradeço ao Banco do Nordeste do Brasil por contribuir parcialmente para a minha participação no curso de Mestrado.

Agradeço ao Prof. Arnaldo Dias Belchior, pela compreensiva e objetiva orientação, por me proporcionar oportunidades ímpares de aprendizado, além de todo o incentivo para participar do mestrado desde antes de começar esse projeto.

Agradeço à Profa. Elizabeth Furtado e também à Profa. Ana Guerra (que presenciou o nascimento deste trabalho, ao fazer parte da banca no *Workshop* de Dissertações de Qualidade de *Software*, no SBQS de 2003), pela participação na banca de defesa da dissertação, contribuindo para o aperfeiçoamento deste trabalho.

Aos professores do Curso de Mestrado em Informática Aplicada da Universidade de Fortaleza, pelas orientações, conhecimentos, incentivos e experiências de aprendizado.

Aos colegas do Curso de Mestrado em Informática Aplicada (MIA) da Universidade de Fortaleza, pelos exemplos de força de vontade, perseverança, companheirismo e colaboração, em especial à turma IV.

Aos profissionais administrativos do MIA, por tornarem nosso caminho mais fácil, em especial a Tânia Maciel.

Agradeço aos diversos profissionais das empresas que colaboraram para a realização do estudo de caso. Sem vocês este trabalho não poderia ser concluído.

Agradeço também aos meus colegas do BNB pelo apoio e pela troca de idéias, em especial a Ivo Alexandre Lopes, Joy Patrícia da Silva, José Airton Fernandes da Silva, João Batista Ribeiro, Lina Ângela Salles, Leonardo Correa Martins e Márcio Maia Silva. E também a minha equipe da Célula de Testes e Qualidade de *Software*.

Agradeço ainda aos meus colegas da qualidade de *software*, pela troca de idéias, pelas conversas, por compartilharmos vitórias e derrotas, pelos cafezinhos, pela amizade e pela disponibilidade: Raimundo Sales Neto e Azevedo, Pascale Correia Rocha e Tatiana Cavalcanti Monteiro.

Por fim, agradeço a todas as minhas amigas e amigos, pelo carinho e amizade demonstrados.

Somos o que repetidamente fazemos.
A qualidade, a excelência, portanto,
não é um feito, mas um hábito.
Aristóteles

Entrega o teu caminho ao Senhor,
Confia nEle, e o mais Ele fará.
Salmo 37:5

RESUMO

Com a crescente dependência da sociedade em relação a produtos de *software*, há uma maior preocupação em como obter software de qualidade. As organizações têm adotado abordagens para garantir a qualidade de *software*, baseado em modelos de qualidade como CMMI, ou em processos de software, como o Processo Unificado (UP). O UP é um processo que objetiva garantir a produção de *software* de qualidade, atendendo as necessidades de usuários por meio de cronogramas e custos previsíveis. Este trabalho analisa a abordagem de qualidade de *software* do UP em relação a modelos e padrões de qualidade utilizados no mercado e propõe uma nova disciplina para o UP, denominada de Gerenciamento da Qualidade, com o objetivo de estabelecer ações que contribuam para a qualidade do processo de *software* do UP. A disciplina proposta visa suprir deficiências identificadas na abordagem de qualidade de *software* do UP, por meio da execução de atividades de garantia de qualidade em projetos, e está alinhada à ISO/IEC 12207 e ao CMMI. Essa disciplina foi aplicada em projetos de *software* em duas organizações produtoras de *software*, auxiliando a prevenir problemas na qualidade de processo e melhorando a qualidade do *software* desenvolvido.

PALAVRAS-CHAVE: Engenharia de Software; Qualidade de Software; Gerenciamento da Qualidade; Garantia de Qualidade de Software (SQA); Processo Unificado (UP).

ABSTRACT

As the globalized society has software's dependence more and more, there is a great concern about to get the software quality. Organizations have adopted approaches to assure software quality, based on quality models, as CMMI, or software processes frameworks, as Unified Process (UP). UP is a software engineering process that aims to ensure the development of high-quality software that meets the needs of its users, within a predictable schedule and budget This work analyses software quality approach on UP, comparing with market's quality models and standards and proposes a new discipline for UP, named Quality Management, that goal is establishing actions that contributes for quality software process in UP. The proposal discipline aims providing supplements for identified deficiencies on UP's software quality approach, by executing quality assurance activities on projects. The new discipline is aligned to requirements of ISO/IEC 12207 and CMMI. This discipline was applied on three software projects in two different software development organizations, supporting the problems prevention on process quality and improving quality software products.

Keywords: *Software Engineering; Software Quality; Quality Management; Software Quality Assurance (SQA); Unified Process (UP)*

LISTAS DE ILUSTRAÇÕES

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CMM	*Capability Maturity Model*
CMMI	*Capability Maturity Model Integration*
FURPS+	*Functionality, Usability, Reliability, Performance, Supportability*
GEQ	*Good Enough Quality*
GQ	*Gerenciamento da Qualidade*
GQM	*Goal/ Question/ Metric*
GQPP	Garantia de Qualidade de Processo e de Produto
GQS	Garantia de Qualidade de *Software*
IEEE	*International Electronical and Electric Engineering*
ISO/IEC	*International Standardization Organization / International Engineering Committee*
MTBF	*Mean Time Between Failure* ou Tempo Médio entre Falhas
OMG	*Object Management Group*
PA	*Process Area* ou Área de Processo
PPQA	*Process and Product Quality Assurance* ou Garantia de Qualidade de Processo e de Produto
PRA	*Project Review Authority*
PSM	*Practical Software Measurement*
QM	*Quality Management* ou Gerenciamento de Qualidade
QA	*Quality Assurance* ou Garantia de Qualidade
QC	*Quality Control* ou Controle de Qualidade
ROI	*Return Over Investment* (Retorno sobre o Investimento)
RUP	*Rational Unified Process* ou Processo Unificado *Rational*
SCAMPI	*Standard CMMI Appraisal Method for Process Improvement* ou Método de Avaliação do padrão CMMI para melhoria de processos
SEI	*Software Engineering Institute*
SGQ	Sistema de Gestão da Qualidade
SEPA	*Software Engineering Process Authority*
SEPG	*Software Engineering Process Group* ou Grupo de Processo de Engenharia de *Software*
SPICE	*Software Process Improvement Capability Evaluation*
SQA	*Software Quality Assurance* ou Garantia de Qualidade de *Software*
UP	*Unified Process* ou Processo Unificado
UML	*Unified Modeling Language* ou Linguagem de Modelagem Unificada

SUMÁRIO

Capítulo 1

INTRODUÇÃO

Este capítulo apresenta a motivação deste trabalho, assim como seus objetivos e limitações.

1.1. Motivação

Existe uma crescente dependência da sociedade em relação aos produtos de *software*, na medida em que os sistemas computadorizados exercem o papel crucial de suportar iniciativas de negócios e projetos da sociedade (FUGGETTA, 2000). Essa dependência levou a uma preocupação cada vez maior da indústria com a qualidade dos produtos de *software* desenvolvidos. Outra questão envolvida nessa discussão são os custos decorrentes da implantação de um produto de *software* sem qualidade, que vão desde os custos com re-trabalho no desenvolvimento de sistemas, passando pelo custo do *software* desenvolvido não funcionar de forma adequada, até o custo de manutenção do aplicativo e do tempo gasto pelos usuários aguardando que os fornecedores corrijam os defeitos por eles apresentados (HARTER & SLAUGHTER, 2000; MINASI, 2000).

Software deve corresponder às expectativas dos seus usuários e, principalmente, funcionar de forma adequada. Dentro deste contexto, a qualidade emergiu como importante fator para desenvolver e implantar produtos de *software* (PRAHALAD & KRISHNAN, 1999), havendo uma maior disposição para se investir em qualidade de *software* (DUARTE & FALBO, 2000).

Dentro do ponto de vista dos fornecedores de *software* de que a qualidade se tornou condição necessária para competir no mercado (KAN, BASILI & SHAPIRO, 1994), tem-se o cenário descrito a seguir. As organizações dependentes de *software* estão exigindo das organizações desenvolvedoras de *software* a comprovação de que seus processos de *software* são aderentes a determinadas normas e padrões internacionais de qualidade. Em contrapartida, organizações desenvolvedoras de *software* antecipam-se a estas demandas e

buscam-se preparar para a certificação em normas, como a ISO 9001 (2000), e em modelos de qualidade de *software*, como o CMMI (2002a; 2002b).

Assumindo que a qualidade do produto de *software* está diretamente relacionada à qualidade do processo de *software* (FUGGETTA, 2000; SOMMERVILLE, 2003), vem-se buscando melhorar o processo de desenvolvimento de *software*, em um esforço que reúne indústria e academia na especificação de padrões e normas internacionais que tratem da qualidade no processo de desenvolvimento.

Um dos processos que vêm apresentando maior aceitação pelo mercado é o Processo Unificado (UP), da *Rational*/IBM, proposto em 1998 com o objetivo de ajudar organizações dos mais diversos portes a desenvolver *software* com qualidade, dentro de prazos e custos especificados. O Processo Unificado é um processo de desenvolvimento de *software* baseado na UML (*Unified Modeling Language*), que especifica quem deve fazer o quê, quando e qual o produto resultante em cada atividade do processo de desenvolvimento (KROLL & KRUCHTEN, 2003). O Processo Unificado (UP) também é conhecido como RUP (*Rational Unified Process*). Neste trabalho, serão utilizadas ambas as nomenclaturas.

Ulferts (2005) fala a respeito da necessidade de criação de uma disciplina para o Processo Unificado, que trate da garantia de qualidade, já que essa garantia representa um papel importante em modelos de maturidades como o CMMI (2002a; 2002b).

Neste contexto, este trabalho propõe uma nova disciplina para o Processo Unificado, Gerenciamento da Qualidade, aderente a ISO/IEC 12207 (2002) e ao CMMI (2002a; 2002b), para tratar especificamente da qualidade do processo de desenvolvimento no UP. A qualidade do produto no Processo Unificado não é foco deste trabalho, uma vez que isto já está contemplado de alguma forma na disciplina de Testes.

A definição e o uso de um processo de qualidade para o processo de desenvolvimento de *software* foram tratados por Unhelkar (2003), para projetos que utilizam processos de *software* baseados em UML.

1.2. Objetivos e Limitações

O objetivo deste trabalho é contribuir para melhorar a visibilidade sobre a qualidade de processo, propondo um processo de gerenciamento da qualidade de *software* para o Processo Unificado.

Os objetivos específicos desse trabalho são os seguintes:

- Realizar levantamento sobre a abordagem de qualidade de *software* existente no Processo Unificado (RUP, 2003).
- Avaliar a abordagem de qualidade de *software* no UP em relação a alguns padrões e modelos de qualidade de *software* em uso no mercado;
- Identificar, na literatura de qualidade, aspectos positivos e negativos em relação à abordagem de qualidade de *software* do UP;
- Experimentar a abordagem proposta, relatando os resultados.

Este trabalho possui as seguintes limitações:

- As organizações onde foram realizados os estudos de caso, apesar de implantarem a garantia de qualidade por meio da disciplina proposta neste trabalho, não haviam implantado ainda o componente de controle de qualidade (testes) de forma efetiva.
- O estudo não aprofundou o trabalho a respeito de métricas de qualidade, utilizando desse tema o mínimo necessário para possibilitar a compreensão da aplicação do processo proposto.
- O estudo não considerou aspectos de custos de qualidade.

1.3. Organização do Trabalho

Além desta Introdução, este trabalho está organizado como descrito a seguir.

No capítulo 2, são apresentados conceitos sobre qualidade de *software* e qualidade do processo de *software* que nortearam o desenvolvimento deste trabalho, evidenciando o gerenciamento da qualidade. A Garantia de Qualidade de *Software* (*Software Quality Assurance* - SQA) é descrita no capítulo 3. O capítulo 4 apresenta as características do Processo Unificado e qual a abordagem de qualidade de *software* nele existente. O capítulo 5 apresenta uma proposta do processo de gerenciamento da qualidade de *software* para o UP. O capítulo 6 apresenta a implantação da proposta em duas organizações. O capítulo 7

apresenta as conclusões deste trabalho com suas contribuições e trabalhos futuros. Acompanham ainda este trabalho alguns apêndices que complementam as informações e aspectos abordados ao longo do texto.

Capítulo 7

CONCLUSÃO

Este capítulo apresenta as conclusões e as contribuições deste trabalho.

Com a exigência cada vez maior do mercado por *software* de qualidade, várias organizações têm utilizado o Processo Unificado (UP) como processo de desenvolvimento de *software*, visando entregar *software* com o escopo estabelecido, dentro dos prazos e com qualidade. Muitas organizações têm buscado adotar modelos e padrões de qualidade de *software* como forma de se manterem competitivas no mercado. A garantia de qualidade em processos de desenvolvimento de *software* vem-se tornando uma necessidade. Organizações que utilizam o UP também necessitam de garantia de qualidade e essa necessidade não é ignorada pelos fornecedores do Processo Unificado (ULFERTS, 2005).

Este trabalho abordou aspectos da qualidade de processo no UP, como o uso de padrões de qualidade, a importância das políticas de qualidade, as revisões de *software* para melhorar a qualidade do processo. A partir de experiências de implantação de SQA em organizações utilizando RUP foi percebida a necessidade de uma disciplina para o Processo Unificado que tratasse do gerenciamento da qualidade de *software*. Foi definida então essa disciplina, buscando alinhá-la às práticas e aos requisitos da ISO/IEC 12207 e do CMMI, e aderente às práticas do Processo Unificado, contribuindo para a melhoria da qualidade no processo de desenvolvimento. Para validar essa disciplina, foi realizado um estudo de caso em duas organizações que utilizam o UP, abrangendo três projetos de *software*.

A equipe de qualidade alocada aos projetos nas duas organizações identificou os principais problemas que ocorriam com o processo de desenvolvimento, instanciado a partir do *framework* do UP, identificando questões de qualidade comuns a vários projetos. A partir da utilização da disciplina de gerenciamento da qualidade, passou a atuar como mentora de projetos, auxiliando-os na realização de suas atividades.

Os resultados obtidos demonstraram que a implantação da disciplina de gerenciamento da qualidade teve os seguintes aspectos positivos:

- evidenciou pontos de gargalo no Processo Unificado;

- possibilitou ao gerenciamento superior uma percepção realista das necessidades dos projetos, fazendo até com que aumentasse o comprometimento gerencial com as ações de qualidade;
- auxiliou equipes de projeto a visualizarem e corrigirem os principais problemas ocorridos ao longo do processo de desenvolvimento;
- permitiu visualizar que há uma barreira cultural muito grande a ser quebrada para passar do desenvolvimento em cascata para o desenvolvimento iterativo;
- corrigiu e preveniu a ocorrência de não conformidades em uma fase/iteração antes que elas impactassem nas atividades da fase/iteração seguintes;
- mostrou que os indicadores de não conformidades resolvidas e identificadas não podem ser vistos por si só, mas dentro do contexto do desenvolvimento iterativo do UP, em que as atividades realizadas e a ênfase a elas dada dependem dos objetivos da fase onde o projeto se encontra.

As principais dificuldades identificadas nos projetos em relação ao uso do RUP, percebidas com a implantação da nova disciplina de qualidade, foram as seguintes:

- planejamento e utilização das ações de desenvolvimento do ponto de vista do uso de um processo iterativo e incremental (especialmente, quando se tem a cultura do ciclo de vida em cascata);
- dificuldades na elaboração da especificação de casos de uso por parte do analista, de forma a melhorar a compreensão dos demais envolvidos em outras etapas do processo (como arquitetos, desenvolvedores e testadores);
- realização de estimativas com base nos casos de uso especificados;
- gerenciamento adequado dos riscos associados aos casos de uso e dos riscos de projeto como um todo.

Além dos aspectos positivos citados, podem-se enumerar as seguintes contribuições deste trabalho, no que diz respeito ao processo de qualidade de *software* para o UP:

- estruturação de um processo de qualidade de *software*, dentro de um contexto de ciclo de vida iterativo/incremental, para projetos que utilizam o Processo Unificado;

- fortalecimento da perspectiva de qualidade do processo de *software* do RUP, estruturando ações e alinhando-as a requisitos e práticas exigidas por modelos e padrões de qualidade, como a ISO/IEC 12207 (1995; 2002) e o CMMI (2002a; 2002b);
- auxilia a organização na estruturação de seus processos rumo à certificação CMMI, nível 2, já que a proposta considera as práticas exigidas por este modelo.

7.1. Trabalhos Futuros

O processo de qualidade definido nesse trabalho pode ser utilizado por outros processos de desenvolvimento (além do Processo Unificado), bastando, para isso, realizar adaptações nos artefatos do processo de garantia de qualidade e nos *checklist*s de avaliação por fase em função das características do processo de desenvolvimento a ser adotado.

Além disto, poder-se-ia analisar questões relevantes para o planejamento iterativo de projetos, considerando projetos com ciclos de vida de desenvolvimento e suas adaptações para ciclos de manutenção de software utilizando o Processo Unificado.

O amadurecimento da disciplina proposta vai acontecer de forma mais apropriada com a aplicação da mesma em um conjunto mais expressivo de projetos em organizações de software.

Capítulo 6
ESTUDO DE CASO

Este capítulo apresenta as informações sobre a implantação da disciplina de Gerenciamento da Qualidade em projetos de software e seus resultados.

A idéia de estruturação da disciplina de qualidade de *software* no Processo Unificado (UP) nasceu a partir da experiência com o uso do Processo Unificado em uma organização que, simultaneamente à implantação do processo, também estava buscando implantar a função de qualidade de *software*, visando certificar-se no modelo CMM-SW nível 2. Essa experiência demonstrou a necessidade do alinhamento da execução das atividades de garantia de qualidade à visão do gerenciamento da qualidade de *software*. A disciplina de Gerenciamento da Qualidade foi aplicada a projetos utilizando o UP, e seus resultados são descritos nas seções a seguir.

6.1. O Método de Pesquisa

A pesquisa realizada neste trabalho tem natureza exploratória e o método de pesquisa utilizado foi o Estudo de Caso. Este método pode ser aplicado nas situações em que o fenômeno estudado é abrangente e complexo e deve ser estudado no seu contexto (BONOMA, 1985). Segundo Benbasat *et al.* (1997), a estratégia de estudo de caso, como método qualitativo, é adequada quando o estudo enfoca eventos contemporâneos e que não podem ser estudados fora do seu contexto natural. Ora, sendo a instanciação de um processo de qualidade em projetos um evento complexo, e com diversas variáveis envolvidas, a estratégia de estudo de caso adequou-se a este contexto, possibilitando estudar as diversas situações de forma prática e avaliar os ajustes que se fizerem necessários.

A pesquisa em estudo de caso é realizada com base na chamada unidade de análise (MARCONI & LAKATOS, 2001). Como a instanciação do processo é realizada em um projeto de *software*, a unidade de análise selecionada é constituída por projetos de

desenvolvimento e manutenção de *software*. Como não é possível falar em qualidade de *software* sem tratar do contexto organizacional, este trabalho caracteriza esse contexto e procura verificar como ele influencia a instanciação do processo de qualidade nos projetos. Além disso, a caracterização do contexto em que os projetos se desenvolveram é importante para compreender as circunstâncias em que o estudo foi procedido e avaliar a aplicabilidade de seus resultados em outros cenários que não os aqui identificados. Buscou-se aplicar o processo de qualidade em projetos de diferentes organizações e avaliar como a proposta se aplicava em cada caso. Os métodos de coleta de dados abrangeram: a análise de documentação e registros de arquivos dos projetos; entrevistas não estruturadas com membros das equipes e observação direta.

Utilizando o método de Estudo de Caso, a pesquisa realizada não estabelece hipóteses a serem testadas, mas é voltada para definir objetivos e buscar mais informações em relação à aplicabilidade da disciplina proposta (YIN, 2002). As questões a serem respondidas ao longo do estudo de caso, ligadas à implantação do processo de qualidade de *software* nos projetos, procuram verificar se a disciplina, em cada projeto:

- Possibilitou identificar e corrigir problemas mais cedo?
- Identificou boas práticas de projetos que poderiam vir a ser utilizadas de forma corporativa?
- Identificou gargalos no processo de desenvolvimento de *software* e formas de corrigir ou prevenir ocorrências destes problemas?
- Obteve o comprometimento, por parte dos envolvidos, com a efetivação das ações corretivas e preventivas propostas?
- Possibilitou que as ações de qualidade realizadas tivessem impacto positivo sobre as métricas coletadas?

Com a finalidade de auxiliar a obtenção de respostas para estas questões foram identificadas algumas métricas de garantia de qualidade, que servem como indicadores para retratar a situação dos projetos durante a implantação da disciplina. As principais métricas identificadas foram:

- quantidade de avaliações planejadas em função da quantidade de avaliações realizadas;

- quantidade de não conformidades identificadas;
- não conformidades mais comuns;
- não conformidades resolvidas no prazo.

Outras questões, também relacionadas a aspectos que influenciam a implantação do processo de qualidade de *software*, podem ser classificadas da seguinte forma:

- *questões envolvidas no contexto organizacional*: qual a importância do comprometimento gerencial para as ações de qualidade de *software* e para a instanciação de um processo de qualidade de *software* na organização? Como a cultura organizacional influencia o êxito das ações de qualidade de *software*? Existe uma preocupação da alta gerência com o processo de desenvolvimento de *software* e sua melhoria ao longo do tempo?
- *questões envolvidas no contexto do projeto*: qual o comprometimento da equipe com a qualidade de *software* e com o processo de desenvolvimento? Qual a capacitação da equipe? Como foi tratada a participação do usuário final nos projetos?
- *questões envolvidas no contexto da qualidade de software*: qual a estrutura de qualidade existente e como ela está relacionada com a estrutura dos projetos de desenvolvimento de *software*? Como foram tratadas as questões relacionadas a políticas e padrões? Qual a participação da equipe de qualidade no apoio aos projetos de desenvolvimento de *software*? Quais as técnicas de qualidade de *software* mais utilizadas em projetos de desenvolvimento de *software*?

Informações relacionadas a estas questões são descritas ao longo do estudo de caso, com a implantação da disciplina nos projetos de desenvolvimento e manutenção de *software*.

6.2. Características das Organizações

Foram selecionadas duas empresas para a realização do estudo de caso. Esta seção apresenta as características de cada organização, tendo em vista:

- o tipo de organização;
- a importância das atividades de desenvolvimento de *software* para a organização;
- a cultura da organização para trabalhar com qualidade de *software*;

- o comprometimento dos envolvidos com as ações de qualidade;
- a preocupação da alta gerência com o processo de desenvolvimento de *software* e sua melhoria;
- a organização de qualidade de *software* na empresa, envolvendo estrutura, pessoas, perfis e experiência da equipe de qualidade, além do uso de políticas e padrões na empresa.

6.2.1. Características da Organização A

Na organização A, a Tecnologia da Informação (TI) é atividade meio, ou seja, a tecnologia dá suporte à realização dos negócios da organização. O desenvolvimento de *software* é uma atividade considerada importante para a consecução dos negócios da empresa. Os principais clientes da atividade de desenvolvimento de *software* são as unidades de negócios da própria organização. As necessidades de *software* desses clientes são traduzidas para a área de desenvolvimento de sistemas pelas unidades de negócios. Um problema que afeta essa organização é a existência de ruídos de comunicação entre a área de desenvolvimento de sistemas e as unidades de negócios. Há um mapeamento inadequado do processo de negócio e das necessidades do negócio, refletido pelo não atendimento de alguns requisitos essenciais no produto de *software* desenvolvido.

Visando minimizar a ocorrência de problemas como este, a organização adotou um processo de *software* instanciado a partir do *framework* do UP e está buscando a certificação CMMI, nível 2. A adoção do UP e utilização da UML visam mapear as necessidades de sistema em casos de uso de sistema, facilitando a comunicação com o cliente unidade de negócio e visando minimizar os problemas de não atendimento dos requisitos. Já o processo de busca da certificação CMMI visa melhorar a atuação do processo de desenvolvimento de *software* como um todo, definindo papéis e responsabilidades e buscando concluir projetos no prazo e dentro dos custos, além de atender requisitos de clientes. Esse processo está na etapa de definição de procedimentos organizacionais para atuação das equipes de projeto em cada área de processo CMMI nível 2, para posterior implantação nos projetos.

A organização possui cerca de 150 profissionais na área de desenvolvimento de sistemas. Essa área está formalmente dividida em dois grupos funcionais, ligados a uma

gerência superior. Um grupo está voltado para o desenvolvimento e a manutenção de projetos de *software*, com cerca de 140 pessoas. O outro grupo é constituído pela equipe de processo e qualidade de *software* da organização. Esses grupos funcionais trabalham de forma coordenada e integrada, sendo independentes entre si, e estão organizados de forma que o grupo preste serviços em relação ao grupo de desenvolvimento de sistemas. O grupo de qualidade é composto atualmente por três pessoas que acumulam as atividades de SQA e SEPG (ou SEPA, segundo o UP). Inicialmente a disciplina proposta foi implantada nesta organização de forma piloto em dois projetos, com o acompanhamento de duas pessoas exercendo o papel de SQA, sendo uma pessoa alocada para cada projeto. A equipe de qualidade da organização A possui experiência com a avaliação de qualidade, uma vez que seus membros trabalharam como SQA de outras organizações e também possuem experiência com a utilização do Processo Unificado. Eles foram treinados em modelos de qualidade, como CMMI, ISO 9000, bem como em técnicas de qualidade.

Com relação à qualidade de *software*, a organização A possui uma cultura de que a qualidade está ligada a entregar produtos dentro do prazo e buscar atender os requisitos dos clientes, isto é, o foco é apenas a qualidade do produto.

A atuação da equipe de qualidade avaliando a qualidade dos projetos antes da implantação das áreas de processo CMMI foi uma estratégia escolhida pela organização A. Essa avaliação prévia de qualidade serviria para identificação de gargalos no processo de desenvolvimento de *software* a serem tratados pela definição e implantação dos procedimentos organizacionais baseados nas áreas de processo do CMMI.

6.2.2. Características da Organização B

Na organização B, uma *software-house*, a TI é atividade fim, sendo a atividade de desenvolvimento de *software* imprescindível para o negócio da empresa. Os principais interessados no resultado das atividades de desenvolvimento de *software* estão fora da organização, são clientes de mercado. A maior preocupação da organização B é o atendimento dos requisitos dos produtos de *software* requeridos pelos clientes, até mesmo por uma questão de sobrevivência da empresa.

A organização adotou um processo de *software* instanciado a partir do *framework* do UP e foi certificada pela ISO 9001, visando manter-se no mercado de forma competitiva. A

organização B é uma pequena empresa com cerca de vinte profissionais na área de desenvolvimento de sistemas. Essa área está dividida em dois grupos com papéis funcionais bem definidos, ligados a uma gerência superior comum. Um grupo é voltado para o desenvolvimento de sistemas. O outro grupo é constituído por uma pessoa que exerce o papel de equipe de qualidade de *software* da organização. Esses grupos trabalham de forma coordenada. O grupo de qualidade é independente do grupo de desenvolvimento de sistemas.

A organização B possui uma forte cultura de qualidade, especialmente por já ser uma organização certificada ISO 9001, amparada no comprometimento gerencial com a melhoria contínua de processos, com a definição da Missão e de Políticas Organizacionais de qualidade, e com o estabelecimento de objetivos de qualidade ligados ao negócio da organização e amparado por ações em diversas áreas. A organização B investiu na utilização de padrões, amparados no uso de *patterns* e no desenvolvimento de um *framework* de componentes que podem ser reutilizados nas diversas aplicações da empresa, minimizando custos e também o tempo de desenvolvimento/manutenção.

A equipe de qualidade da organização B participou de vários treinamentos na área de qualidade de *software*, tendo sido treinada em modelos e padrões de qualidade de *software* e trabalhado como SQA em organização cujo processo de desenvolvimento de *software* foi desenvolvido com base nas práticas do UP. A equipe de qualidade possui também experiência com a implantação de modelos e padrões de qualidade em organizações, tendo participado de processos de implantação do CMM e ISO 9001.

6.3. Implantação da Disciplina de GQ em Projetos

Foram alvos deste estudo de caso três projetos, sendo dois de desenvolvimento de *software* (da organização A) e um de manutenção de *software* (da organização B). Aplicando a disciplina de qualidade aos projetos, buscou-se avaliar sua aderência em relação ao ciclo de vida iterativo/incremental do UP.

Na organização A, inicialmente, foram realizadas algumas ações no âmbito organizacional visando proporcionar o ambiente necessário para realização das atividades de qualidade. Essas atividades começaram a implantação de SQA em dois projetos pilotos

para a organização. Como a organização adotou e implantou o UP, os profissionais da organização A haviam sido treinados em cursos oficiais da *Rational*/IBM sobre o processo de *software* UP. Também houve treinamento sobre o processo de qualidade constituído pela disciplina de Gerenciamento da Qualidade para os *stakeholder*s dos projetos pilotos.

Na organização B, em função da certificação ISO, o ambiente necessário para realização das atividades de qualidade já estava previamente estabelecido, em termos de cultura de qualidade, comprometimento gerencial e estrutura de apoio à execução das atividades de qualidade nos projetos. Com a adoção do UP, os profissionais foram treinados no Processo Unificado e no novo processo de qualidade de *software*, com o esclarecimento sobre a sistemática de avaliação de qualidade ao longo das fases do UP.

A seguir, descreveremos a implantação da disciplina de qualidade nos projetos. É importante informar que as atividades de qualidade acontecem sempre em paralelo com as atividades dos projetos. Sua interferência nas atividades dos projetos dá-se apenas quando os resultados das ações de qualidade influenciam na melhoria na realização das atividades técnicas e gerenciais do projeto, conforme se pode observar durante as avaliações objetivas de qualidade do projeto.

6.3.1. Projeto X

O projeto X, da organização A, trata do desenvolvimento de um sistema para suporte a uma unidade de negócio da organização. O *software* a ser desenvolvido é crítico para o negócio; além disso, seu desenvolvimento é exigência legal de órgão regulador, ao qual a organização A está subordinada. O tempo de duração do projeto foi de cinco meses, estando dividido em quatro iterações, correspondendo cada fase a uma iteração. O projeto possuía ao todo cinco casos de uso e contou com sete pessoas na equipe, sendo um analista, um arquiteto, três desenvolvedores trabalhando em paralelo, e dois testadores. Foi alocada uma pessoa da equipe de qualidade ao projeto, que executou simultaneamente os papéis de gerente de qualidade e engenheiro de qualidade. Apresentamos, a seguir, os dados relativos à instanciação da disciplina de Gerenciamento da Qualidade no Projeto X.

Planejamento da Qualidade do Projeto

A atuação da equipe de qualidade começou após o projeto haver iniciado, através de uma decisão da gerência de utilizar o projeto X como piloto, uma vez que esse projeto possuía poucos casos de uso e que esses casos de uso não eram muito complexos. Como o *software* é crítico para o negócio, e os requisitos legais devem ser atendidos sem quaisquer desvios, o planejamento da qualidade do projeto foi realizado visando atender os objetivos do projeto de identificação e mapeamento de requisitos com a unidade de negócios. O planejamento da qualidade contemplou a realização de cinco avaliações de qualidade (auditorias) ao longo das quatro fases do projeto, sendo uma avaliação a cada fase, à exceção da fase de Elaboração, para a qual foram programadas duas avaliações de qualidade.

As avaliações de qualidade em cada fase não abrangeram todo o conjunto de práticas e artefatos do projeto, mas tiveram como foco os aspectos que eram mais críticos para o projeto em determinado momento da fase. Uma vez que o planejamento do projeto já havia ocorrido, o planejamento da qualidade não pôde acontecer paralelamente a essa ação nesse momento. As informações sobre garantia de qualidade para o projeto foram registradas no Plano de Garantia de Qualidade, o qual foi aprovado pelos *stakeholders* e pelo gerenciamento.

Fase de Iniciação

Quando a equipe de qualidade foi alocada ao projeto, a Fase de Iniciação ainda não havia terminado, mas o projeto já havia feito seu planejamento, em nível macro, das fases e iterações subseqüentes. A equipe de qualidade planejou a realização das atividades de qualidade (auditoria) nessa fase tendo como foco o planejamento do projeto e a definição do escopo do projeto. O planejamento da qualidade foi registrado no Plano de Garantia de Qualidade do Projeto X (Apêndice D).

A auditoria realizada nessa fase identificou que a distribuição inicial das atividades no cronograma do projeto foi realizada sem utilizar a estratégia de divisão por fases de forma adequada, dificultando as ações do projeto de tratamento dos riscos. O erro no planejamento consistiu em colocar apenas atividades de análise na fase de Iniciação, apenas atividades de projeto na fase de Elaboração, apenas atividades de desenvolvimento de código na fase de Construção, e apenas atividades de implantação na fase de Transição, o que representa um resquício cultural extremamente forte do desenvolvimento em cascata.

Outro problema decorrente dessa visão equivocada foi a não realização, nessa fase, pelo projeto, das atividades de estabelecer uma sugestão de arquitetura e de planejar os possíveis testes a serem realizados para a aplicação. Outras não conformidades identificadas na Fase de Iniciação foram:

- Dúvidas de terminologia que tiveram reflexo na documentação do projeto, como por exemplo, a diferenciação entre *stakeholders*, usuários do sistema e atores dos casos de uso, e entre requisitos e regras de negócio.
- Requisitos não funcionais, possíveis sugestões de arquitetura e avaliação do esforço de testes não haviam sido registrados nesta fase.
- A equipe do projeto não conseguiu estabelecer quão detalhadas deveriam ser as informações durante a fase, de forma a permitir que o planejamento do projeto pudesse ser realizado.
- Houve dúvidas sobre a seqüência de passos a serem realizados para gestão do projeto, seguindo o Processo Unificado. As principais dúvidas estiveram relacionadas à distribuição das atividades do projeto ao longo das fases, à identificação dos riscos, e ao planejamento das iterações e fases, seguindo o processo iterativo.

As ações corretivas foram discutidas com a equipe do projeto e estabelecidas no Relatório de Avaliação de Qualidade da Fase de Iniciação. Foi identificada a necessidade de prover orientação aos membros do projeto para diferenciação e identificação de terminologia de engenharia de *software* (por exemplo, diferença entre requisitos e regras de negócios) para o projeto, aplicando-os corretamente na prática, refletindo na melhor compreensão dos requisitos da aplicação. As métricas de qualidade da fase foram coletadas e podem ser visualizadas na Tabela 6.1.

Os resultados da auditoria de qualidade foram reportados para a equipe do projeto e as não conformidades foram resolvidas dentro dos prazos acordados, sem necessidade de escalá-las para o nível superior da gerência. Esses resultados influenciaram, ainda na fase de Iniciação, a realização de atividades técnicas que estavam faltando ser concretizadas (como a definição da sugestão de arquitetura e o planejamento do esforço de testes). Os resultados da auditoria realizada na Fase de Iniciação acabaram impactando o planejamento das próximas fases do projeto também. O planejamento foi re-trabalhado pela equipe do projeto, com a orientação da equipe de qualidade.

Na realização das atividades de monitoramento da qualidade da iteração/fase, percebeu-se que houve mais não conformidades em termos de práticas do processo que não foram seguidas do que em relação ao uso dos artefatos. Os objetivos de qualidade estabelecidos em relação ao planejamento do projeto e à definição do escopo e sua validação com os usuários foram atingidos. As atividades de qualidade planejadas foram realizadas e o esforço alocado da equipe de qualidade foi suficiente para a demanda do projeto, não tendo sido registrados desvios nessa fase.

Fase de Elaboração

Na Fase de Elaboração, inicialmente foi planejada a realização de duas auditorias de qualidade. A primeira teve como objetivos verificar a qualidade das especificações de Casos de Uso e seu mapeamento com os requisitos e verificar o mapeamento dos requisitos em relação à arquitetura definida para o *software*. O Plano de Garantia de Qualidade do Projeto X (Apêndice D) foi atualizado com esses objetivos e com as atividades de qualidade referentes a essa fase.

Os principais resultados da primeira auditoria realizada na Fase de Elaboração podem ser encontrados no Relatório de Avaliação de Qualidade do Projeto X para a fase de Elaboração, disponível no Apêndice E. Um resumo desses resultados é descrito a seguir. Requisitos suplementares (não funcionais) foram finalmente especificados e foi mensurado seu impacto sobre a arquitetura do *software*. As principais não conformidades identificadas estiveram relacionadas aos seguintes pontos do processo:

- avaliações sobre possíveis soluções de arquitetura não foram documentadas, sendo documentada apenas a arquitetura definida para o *software*;
- a arquitetura para o aplicativo não foi descrita utilizando a visão dos modelos 4+1 da UML para a arquitetura de *software* (RUP, 2003);
- o mapeamento e tratamento de riscos foi um dos aspectos de maiores dificuldades. A equipe do projeto consegue identificar os riscos, mas não consegue acompanhá-los utilizando uma estratégia de mitigação e/ou de contingência.
- a equipe apresentou dificuldades em especificar casos de uso adequadamente;
- o projeto apresentou dificuldades para realizar estimativas com base nos Casos de Uso especificados;

- houve dúvidas por parte da equipe, em relação ao nível de detalhamento das informações da documentação de análise e projeto (*design*) feita durante a iteração, não identificando claramente o ponto em que terminava a atuação do analista e onde começava o papel do arquiteto na definição da estrutura do sistema.

As ações corretivas foram discutidas com a equipe do projeto e abrangeram os itens descritos em seguida. A equipe do projeto deveria, além de identificar os demais riscos do projeto, procurar desenvolver e aplicar uma estratégia de mitigação e/ou de contingência aos riscos mapeados. O projeto deveria documentar os aspectos do aplicativo que não haviam sido descritos, como os modelos UML de representação da arquitetura do *software*. Os resultados da auditoria de qualidade foram reportados para a equipe do projeto e as não conformidades foram novamente resolvidas dentro dos prazos, sem necessidade de escalamento para a gerência superior e/ou sênior.

Houve registro de um desvio: foi identificado que o projeto não estava registrando as estimativas requeridas pelo Plano de Desenvolvimento de *Software*. A dificuldade da equipe do projeto em estimar com base nos casos de uso era também uma dificuldade na organização A, uma vez que cada Especificação de Caso de Uso possuía uma forma particular de ser construída em cada projeto. Após aceite dos *stakeholder*s, a não especificação das estimativas para o projeto foi registrada como um desvio.

A segunda auditoria de qualidade buscou avaliar a rastreabilidade entre a documentação de Análise e *Design* e a implementação do *software* e de seus componentes, além do planejamento e da execução dos testes de validação da arquitetura. Os principais resultados foram: os modelos de *design* UML estavam incompletos, ocasionando dúvidas quando da geração dos modelos de implementação do *software*; a não especificação de requisitos não funcionais, identificada como não conformidade na primeira fase, foi providenciada pela equipe do projeto no início da fase de Elaboração, possibilitando que o projeto da arquitetura considerasse estes aspectos e possibilitando que os testes de validação da arquitetura observassem esse tipo de requisito. Não foram identificados problemas em termos de rastreabilidade. As ações corretivas propostas foram realizadas.

Um aspecto interessante foi que, em virtude da equipe do projeto não conseguir identificar adequadamente o nível de detalhamento das informações a ser realizado em cada iteração, houve sobreposição de papéis durante essa fase. Isto resultou em atividades

realizadas de forma parcial, gerando desgaste entre os papéis (analista e arquiteto) e confusão no relacionamento com o cliente (usuário final), já que muitas vezes o arquiteto entrava em contato com o usuário final para obter informações não detalhadas pelo analista na documentação. Outra conseqüência foi de que o papel do arquiteto, ao se envolver com atividades de análise, acabou não realizando as suas atividades com um nível de qualidade adequado. A Revisão Técnica de Requisitos que havia sido planejada para o primeiro momento da fase não foi realizada, por uma decisão da gerência do projeto. Se fosse realizada, como previsto, poderia ter evitado essas ocorrências, minimizando o risco de problemas em requisitos, de ausência de informações nas especificações de Casos de Uso, e impactado menos o trabalho do arquiteto. As métricas coletadas nessa fase foram consolidadas na Tabela 6.1.

As atividades de qualidade possibilitaram que os problemas identificados na fase anterior fossem resolvidos antes que tivessem maior impacto na fase de elaboração. Em função das características de cada fase e de seus respectivos marcos, houve menos dificuldades com as práticas do processo, mas ocorreram mais dificuldades em termos de uso dos artefatos e da ênfase do refinamento das informações sobre o *software* a ser registrada nos artefatos.

Fase de Construção

Já na Fase de Construção, o planejamento de qualidade visou avaliar o impacto das mudanças solicitadas pelos *stakeholders* sobre os requisitos, e a rastreabilidade dessas mudanças sobre os casos de uso, as realizações de casos de uso, o projeto do banco de dados, os componentes já desenvolvidos, e sobre o executável como um todo. O Plano de Qualidade do Projeto X foi refinado com as informações relativas à fase de Construção. Os principais resultados em relação às não conformidades identificadas foram:

- Houve uma evolução da equipe no trabalho de especificar Casos de Uso. Com as especificações de casos de uso com uma melhor qualidade e mais detalhadas, foi possível planejar um esforço de teste para a aplicação que se mostrou bastante acurado e alinhado aos objetivos do projeto.
- O projeto apresentou dificuldades com o planejamento iterativo, no que diz respeito à distribuição de casos de uso para a implementação. Essa dificuldade

existiu em virtude das mudanças nos requisitos e em função da necessidade de paralelizar o esforço de desenvolvimento para que fosse possível cumprir os prazos acordados com o cliente.

- Houve dúvidas sobre como o projeto faria para terminar de tratar requisitos para iniciar a codificação, uma vez que os usuários sempre estavam solicitando mudanças em requisitos existentes e solicitando novos requisitos. A equipe de qualidade e a equipe do projeto responderam a este problema por meio do planejamento iterativo do projeto, com a equipe do projeto negociando com o cliente a importância e a necessidade de cada requisito, para entregar ao usuário final a funcionalidade requerida. O planejamento da iteração foi realizado com base nos casos de uso definidos com o cliente e com a avaliação dos riscos referentes a esses novos requisitos e às solicitações de mudanças para o projeto como um todo.
- Não foi iniciado o planejamento da implantação, conforme recomendado pelas práticas do processo, nem foi evidenciada a realização de testes de unidade pelos desenvolvedores.
- Foi possível rastrear o impacto das solicitações de mudança sobre os itens (atividades e artefatos) que o projeto já havia feito, possibilitando uma melhor avaliação dos riscos de efetuar mudanças solicitadas.

O projeto foi acompanhado pela equipe de qualidade com orientações ao longo da fase. A necessidade de paralelizar o esforço de desenvolvimento foi um ponto que ocasionou a revisão do planejamento da fase/iteração, levando a dúvidas entre planejar apenas com base nos riscos associados aos casos de uso ou tendo em vista a necessidade de disponibilizar funcionalidades ao cliente, em função dos recursos disponíveis alocados ao projeto.

Fase de Transição

Na Fase de Transição, a avaliação de qualidade buscou principalmente avaliar se as providências para implantar o *software* foram tomadas e se os erros críticos identificados nos testes realizados na fase anterior foram corrigidos antes de o *software* entrar em produção. Os principais resultados foram: as dificuldades na prévia organização do projeto para implantar o produto desenvolvido, abrangendo: documentação do usuário, documentação do produto, treinamento do usuário etc; o projeto não apresentou

dificuldades em relação à realização de testes de sistemas e correção dos defeitos antes de disponibilizar o *software* em produção.

Foram estabelecidas como ações corretivas que a documentação do *software* a ser utilizada na implantação fosse desenvolvida, que fossem realizados treinamentos para os usuários e que fosse planejada e realizada a reunião de fechamento do projeto. As ações corretivas foram também realizadas dentro dos prazos acordados.

Lições Aprendidas de Qualidade

Concluindo as atividades de qualidade para o Projeto X, as Lições Aprendidas de Qualidade para o projeto foram:

- Foi bastante positivo vincular os objetivos do projeto e os objetivos da fase aos objetivos de qualidade, para que as avaliações de qualidade pudessem verificar a aderência das ações do projeto aos objetivos estabelecidos, com foco sobre o que era mais importante para o projeto.
- Uma vez utilizando desenvolvimento iterativo, a recorrência de não conformidades é minimizada.
- Conforme entendimentos com a equipe do projeto, as avaliações de qualidade devem ser discutidas para que ocorram em pontos de verificação específicos dentro da fase e da iteração, de forma a minimizar o impacto de não conformidades para a iteração ou fase seguinte. Alguns desses pontos de controle puderam ser percebidos ao longo do Projeto X, corroborando o que já havia sido descrito no capítulo 5 em termos de garantia de qualidade das atividades de cada disciplina, que vêm a compor as atividades executadas em cada fase.
- Percebeu-se que havia dificuldade por parte da equipe do projeto com a documentação e especialmente com o nível de detalhamento das informações ao longo da iteração/fase, ou seja, o quão detalhada deveria ser a informação descrita em cada artefato, ao longo do processo iterativo. A equipe se mostrou confusa com esse aspecto do desenvolvimento iterativo, além de ter dificuldades em dimensionar o nível de detalhamento da documentação de análise e projeto (*design*), não identificando claramente o ponto em que terminava a atuação do analista e onde começava o papel do arquiteto na definição da estrutura do sistema.

As métricas de qualidade para o projeto X são informadas na Tabela 6.1.

Tabela 6.1 – Métricas de Qualidade do Projeto X por fase

	Iniciação	Elaboração	Construção	Transição
Percentual de avaliações de qualidade planejadas versus realizadas	100%	100%	100 %	100%
Quantidade de Não Conformidades identificadas (novas)	• 06 não conformidades de produto • 10 não conformidades de processo	• 10 não conformidades de produto • 06 não conformidades de processo	• 02 não conformidades de produto • 03 não conformidades de processo	• 03 não conformidades de produto • 07 não conformidades de processo
Percentual de Não Conformidades resolvidas no prazo	100%	100%	100%	100%
Índice de Desvios	Nenhum desvio	Um desvio	Nenhum desvio	Nenhum desvio

Durante a Fase de Elaboração, houve um elevado número de não conformidades de produto em relação às demais fases, o que pode ser explicado pelo uso intensivo da UML nessa fase e pela dificuldade apresentada pelo projeto em lidar com a modelagem UML do *software*. O maior número de não conformidades de processo concentrou-se na fase de Iniciação, devido aos resquícios culturais do uso de processo em cascata e a adoção do modelo iterativo/incremental do UP, e na fase de Transição, em função das diversas práticas tradicionalmente relegadas a segundo plano pela gestão de projetos no que diz respeito à preparação do *software* para implantação (suporte, treinamento, testes, migração em paralelo etc.).

Não é possível estabelecer uma relação entre o número de não conformidades de uma fase em relação ao número da fase anterior, já que a ênfase das práticas em cada fase é diferente, e muitas vezes as não conformidades identificadas não são sobre práticas que foram alvo de ação corretiva em iteração anterior, mas são sobre novas práticas executadas de acordo com a fase e com os objetivos do projeto, e que a equipe do projeto apresentou dificuldades em realizar.

6.3.2. Projeto Y

O projeto Y trata da manutenção de um sistema existente, que é o principal produto que a organização B disponibiliza no mercado. Por ser o "carro-chefe" da organização B, quaisquer manutenções nesse produto, sejam evolutivas, adaptativas ou corretivas, são críticas para o sucesso da organização. O sistema possui ao todo quatro casos de uso, mas o projeto de manutenção abrangia dois novos casos de uso que não possuíam impacto sobre a arquitetura do *software*. Esse projeto contou com três pessoas na equipe, sendo um analista/gerente de projeto/arquiteto, um desenvolvedor e um testador. Foi alocada uma pessoa da equipe de qualidade ao projeto, que realizou simultaneamente o papel de gerente de qualidade e engenheiro de qualidade.

O UP não apresenta uma "fase" de manutenção, mas seu ciclo de vida compreendendo as quatro fases pode ser tratado como um ciclo de vida de manutenção, uma vez que todas as atividades de um processo de *software* realizadas durante uma manutenção já são previstas pelo Processo Unificado (KRUCHTEN, 2001). Neste caso, os ciclos de evolução no UP tratam do refinamento de vários artefatos já existentes, recomendando-se a avaliação do impacto dos requisitos que fazem parte do escopo da evolução, a fim de que seja possível dimensionar a duração de cada Fase do projeto. Conforme descrito no Capítulo 4, as fases do UP não podem ser "puladas" durante a execução do projeto. Em projetos cujos requisitos não possuem impacto sobre a arquitetura, a Iniciação pode ser curta o suficiente para definir e validar o escopo do projeto e a Elaboração pode durar de um a dois dias, registrando-se que os critérios de finalização de cada uma das fases foram atendidos.

No caso em que os requisitos possuam impacto sobre a arquitetura, é necessário planejar e executar completamente a Fase de Elaboração, a fim de evoluir e validar a arquitetura do *software*. O projeto Y é um caso de manutenção evolutiva em que os requisitos que fazem parte do escopo do projeto não possuem impacto sobre a arquitetura. Assim, a duração do projeto foi de quatro semanas. Apresentamos, a seguir, os dados relativos à instanciação da disciplina de Gerenciamento da Qualidade no projeto Y.

Planejamento da Qualidade do Projeto

O planejamento da qualidade do projeto foi realizado tendo por base os objetivos do projeto de atendimento das necessidades de alguns novos clientes, sem impactar as funcionalidades já existentes na aplicação e amplamente utilizadas. Outro objetivo que norteou o planejamento das ações de qualidade foi a necessidade de realizar testes de regressão sobre o aplicativo, visando avaliar e reduzir o impacto na implementação da nova funcionalidade coexistindo com as já existentes.

O planejamento da qualidade contemplou a realização de duas avaliações de qualidade ao longo do projeto, sendo uma na fase de Iniciação, tendo por foco o planejamento do projeto e os requisitos, e a outra na fase de Construção, apontando para a implementação da nova funcionalidade e a realização dos testes de regressão.

As informações sobre o planejamento da qualidade foram registradas no Plano de Garantia de Qualidade do Projeto Y. Segundo o ciclo de vida de manutenção, foi acordado com os *stakeholders* que, para as fases em que o esforço do projeto fosse apenas de recuperação de dados para avaliar se os critérios da fase foram atingidos, o esforço de qualidade abrangeria apenas a avaliação da revisão gerencial realizada (revisão de marco de ciclo de vida). Neste caso, pode-se dizer que a disciplina de Gerenciamento de Qualidade foi adaptada para se adequar ao cronograma do projeto de manutenção, não realizando auditorias em determinadas fases, mas apenas avaliando se a revisão gerencial de fechamento da fase havia sido realizada a contento.

Fase de Iniciação

A distribuição de avaliações de qualidade procurou refletir as necessidades do projeto durante a fase. Por exemplo, durante a Iniciação, os objetivos de qualidade voltaram-se para trabalhar as dificuldades do analista com o planejamento do projeto em termos de impacto sobre as próximas fases, como em relação aos novos requisitos e seu relacionamento com a arquitetura da aplicação e o planejamento dos testes.

Um fato interessante nesse projeto, apesar do cronograma curto, é que foi prevista a realização de uma revisão técnica de requisitos, realizada por um analista de outro projeto, numa sistemática de revisão por pares. Apesar de realizada, essa revisão não obteve

resultados muito satisfatórios, uma vez que foram percebidas distorções pela auditoria de qualidade na documentação de requisitos e nas especificações de casos de uso.

A equipe do projeto, imediatamente após ser comunicada dos resultados da avaliação de qualidade, providenciou as ações corretivas. Foi recomendado um treinamento efetivo sobre modelagem UML com foco nas especificações de casos de uso. Entretanto, em virtude do tempo total do projeto, foi definido pela gerência da organização que o treinamento ocorreria após o término do projeto, devendo ter seu impacto avaliado em projetos futuros. Para auxiliar na adequação das especificações de casos de uso após os resultados reportados, a equipe de qualidade orientou os analistas do projeto no sentido de eliminar as distorções que haviam sido identificadas durante a realização da auditoria.

A equipe do projeto demandou à equipe de qualidade para auxiliá-la no planejamento das atividades do projeto para as próximas fases, tendo em vista o ciclo de vida de manutenção. O planejamento das fases e iterações foi realizado e, posteriormente, mostrou-se bem sucedido.

Os objetivos de qualidade para essa fase foram atingidos, inclusive corrigindo distorções identificadas na documentação de requisitos e Casos de Uso, principal gargalo do projeto. Foi percebida a necessidade de ter revisores técnicos mais experientes para realizar as atividades de revisão técnica no nível de qualidade esperado.

Fase de Elaboração

A única ação de qualidade realizada na Fase de Elaboração consistiu em verificar, uma vez realizada a reunião de Revisão de Marco de Ciclo de Vida (Capítulo 4), se os critérios de conclusão da fase haviam sido atingidos. Com os objetivos da Fase de Elaboração atendidos, o projeto passou à realização das atividades da fase seguinte.

Fase de Construção

Na fase de Construção, os objetivos de qualidade abrangeram as ações de detalhamento dos casos de uso remanescentes, a implementação das novas funcionalidades e a realização dos testes sobre o aplicativo, visando verificar o impacto da implantação das novas funcionalidades coexistindo com as demais funcionalidades que o aplicativo já possuía.

Para os dois novos casos de uso foram realizadas as atividades de: detalhamento, análise, *design* e implementação. Foram executados os testes de unidade, integração do *software* e os testes de sistema, além da preparação do ambiente operacional para implantação da nova versão. Um ponto de destaque neste projeto é a reutilização de componentes de *framework* de desenvolvimento, desenvolvido internamente, e que é considerado um padrão de qualidade dentro da organização B. A principal dificuldade identificada durante a auditoria do projeto foi que não foram encontradas evidências de que a documentação de suporte ao produto foi atualizada ou de que foi previsto o treinamento dos usuários em relação às novas funcionalidades.

As não conformidades foram reportadas à equipe do projeto e foram definidos as ações corretivas, os prazos de conclusão e os responsáveis, sendo todas essas informações devidamente registradas no Relatório de Avaliação de Qualidade do projeto Y. O projeto atendeu os prazos, sem necessidade de escalar não conformidades para resolução pelo nível de gerência superior ao da gerência do projeto.

Monitorando os resultados da qualidade nessa fase, observou-se que os objetivos pretendidos para a fase foram atendidos. A equipe de qualidade mostrou-se atenta às necessidades do projeto, orientando o mesmo em suas necessidades em relação ao uso do processo e uso das *templates* dos artefatos. O esforço de qualidade para a fase em termos de alocação de equipe mostrou-se adequado para o escopo do projeto.

Fase de Transição

Houve apenas uma verificação simples nesta Fase, visando observar se os testes validaram o produto em relação aos requisitos solicitados, se os defeitos identificados nos testes foram corrigidos e se a quantidade de defeitos diminuiu após os testes de regressão, além de verificar se os procedimentos para implantar o *software* foram concluídos. Não foram identificadas não conformidades. Os resultados foram relatados e registrados para acompanhamento posterior dos registros de qualidade.

As métricas de qualidade para o projeto Y são informadas na Tabela 6.2 a seguir, para as fases consideradas:

Tabela 6.2 – Métricas de Qualidade do Projeto Y por fase

	Iniciação	**Construção**
Quantidade de avaliações de qualidade planejadas versus realizadas	100%	100 %
Quantidade de Não Conformidades identificadas (novas)	• 4 não conformidades de produto • 3 não conformidades de processo	• 3 não conformidades de produto • 2 não conformidades de processo
Não Conformidades resolvidas no prazo	100%	100%
Índice de Desvios	Nenhum desvio	Nenhum desvio

Lições Aprendidas de Qualidade

As principais Lições Aprendidas de Qualidade para esse projeto foram:

- Projetos curtos e com equipes pequenas não costumam ter muitas variações de objetivos e mudanças de requisitos que tenham impacto sobre o planejamento do projeto e o planejamento da qualidade ao longo das fases.
- A atuação da garantia de qualidade sobre o projeto minimizou a ocorrência de problemas com casos de uso, diminuindo o impacto sobre as demais ações que dependiam deles bem especificados.
- Seria importante alinhar a atuação da garantia de qualidade e dos testes, avaliando as métricas de qualidade do processo (não conformidades de processo) e as métricas de qualidade do produto (índice de defeitos) para se ter uma avaliação do nível de qualidade do projeto como um todo, em termos de processo e produto.
- A disciplina de Gerenciamento de Qualidade mostrou-se adequada para lidar com projetos de manutenção do UP, por meio da adaptação das suas atividades ao ciclo de vida do projeto.

A gerência da organização mostrou-se receptiva às sugestões da equipe de qualidade, apoiando e incentivando o corpo técnico a buscar a melhoria contínua de processos e também apoiando as ações de qualidade.

6.3.3. Projeto Z

O projeto Z da organização A tratou do desenvolvimento de um novo *software* para substituir um sistema já existente na organização. O sistema legado era parcialmente automatizado e ainda apresentava algumas funcionalidades críticas realizadas de forma manual, aumentando os riscos para o negócio. O novo *software* deveria substituir o anterior por completo e automatizar as funcionalidades executadas manualmente. Esse projeto foi considerado de desenvolvimento, e não de evolução do *software* existente (manutenção evolutiva), porque todo o sistema legado seria descontinuado e seria desenvolvida uma nova solução completa para a unidade de negócio cliente do sistema.

Dois riscos para esse projeto eram: a iminente saída da organização A do analista responsável pelo *software*; e a ausência de documentação atualizada sobre o *software* existente (legado), a ser substituído pelo novo. No UP, esses riscos seriam categorizados como risco de equipe e risco técnico, respectivamente. A própria ação de estabelecer um projeto de desenvolvimento da nova aplicação visava mitigar esses dois riscos, documentando os requisitos e funcionalidades antes que o analista do *software* saísse da organização, uma vez que era o único conhecedor das regras de funcionamento do sistema legado e até mesmo dos critérios para execução manual de determinadas funcionalidades.

O tempo de duração do projeto foi em torno de cinco meses. No início do projeto, as funcionalidades do aplicativo ainda não estavam mapeadas em casos de uso. Foi alocada ao projeto uma equipe com cinco pessoas, sendo um gerente de projeto, um analista/arquiteto, dois desenvolvedores e um testador. Para executar simultaneamente os papéis de gerente de qualidade e engenheiro de qualidade foi alocada uma pessoa da equipe de qualidade. Apresentamos, a seguir, os dados relativos à instanciação da disciplina de Gerenciamento da Qualidade no projeto.

Planejamento da Qualidade do Projeto

Paralelamente ao planejamento do projeto, a equipe de qualidade realizou o planejamento de qualidade para o projeto. Inicialmente, foram definidos os objetivos de qualidade, instanciados os padrões de qualidade para o projeto como um todo, e foi

estabelecido o uso da técnica de Auditoria para identificar o atual estágio do projeto de desenvolvimento em termos de qualidade de processo de *software*.

Fase de Iniciação

Na fase de Iniciação, as principais preocupações da equipe eram: o fechamento do escopo do projeto, identificando as funcionalidades com base no sistema existente (legado) e nas atividades realizadas de forma manual por alguns dos usuários do sistema; e a documentação das funcionalidades, por meio dos modelos de casos de uso. Outra preocupação era identificar os riscos associados a cada funcionalidade descrita, de forma a permitir um adequado planejamento do projeto. Foi realizada uma auditoria na Fase de Iniciação e as principais não conformidades identificadas foram:

- Existiram não conformidades em relação à identificação dos casos de uso de sistema pelo analista, com base no *software* existente. O analista fez um mapeamento em que um caso de uso do sistema novo correspondia a um item do menu do *software* antigo. A representação dos casos de uso ficou distorcida, uma vez que havia uma grande quantidade de casos de uso para representar ações pontuais que, analisando, podia-se perceber que faziam parte de uma mesma funcionalidade.
- Houve dúvidas sobre a seqüência de passos a serem realizados para gerenciar o projeto, seguindo o Processo Unificado. As principais dificuldades estiveram ligadas à identificação dos casos de uso de maior risco e os tipos de riscos associados aos casos de uso, e em como seriam planejadas as futuras iterações a partir dos riscos identificados e tendo como base os casos de uso.
- Houve confusão no uso da documentação de riscos. Apesar de identificados os riscos relacionados aos Casos de Uso e também os riscos de negócio, eles foram documentados, respectivamente, na Especificação de Requisitos de *Software* e no documento de Visão, ignorando completamente o artefato Lista de Riscos e resultando em uma certa dispersão da equipe no tratamento dos riscos, pois uma vez registrados, os riscos não foram acompanhados de forma adequada.
- Não foi definida uma estratégia de testes para o aplicativo.

As não conformidades identificadas foram reportadas à equipe do projeto para resolução, juntamente com as ações corretivas sugeridas, utilizando o artefato Relatório de Avaliação de Qualidade. As não conformidades foram solucionadas pela equipe do projeto, dentro dos prazos acordados, dispensando o escalamento e a intervenção dos níveis superiores de gerência.

Orientações sobre os casos de uso foram prestadas ao projeto e distorções em casos de uso documentados a partir do sistema legado foram corrigidas. O projeto então identificou cinco casos de uso, sendo dois com impacto sobre a arquitetura e três menos complexos.

Fase de Elaboração

O planejamento da qualidade nessa fase foi refinado para buscar antecipar a ocorrência de problemas que tivessem impacto para a arquitetura do *software.* Os casos de uso foram detalhados e foi avaliado se as novas funcionalidades que seriam automatizadas teriam ou não impacto sobre a arquitetura da aplicação. Os padrões de qualidade a serem utilizados nessa fase foram instanciados para o projeto. Foi planejada a realização de uma auditoria nessa fase que, uma vez realizada, demonstrou a existência das seguintes não conformidades:

- Depois dos problemas com a identificação e descrição preliminar dos casos de uso na fase anterior, que foram corrigidos a contento e dentro dos prazos esperados de modo a não impactar a Fase de Elaboração, o projeto apresentou dificuldades em *detalhar* a especificação dos casos de uso. Ao detalhar os casos de usos críticos para o trabalho da fase, o analista misturou informações dos passos do caso de uso com informações de interface gráfica de usuário. Deste modo, uma alteração no caso de uso gerava alteração no artefato protótipo de interface gráfica e vice-versa. A funcionalidade não ficou bem descrita, podendo ser percebida ainda uma falta de estruturação dos casos de uso detalhados, traduzidas pela ausência de fluxos alternativos de casos de uso. Alguns casos de uso também foram “explodidos”, como quando se faz projeto utilizando a técnica de Diagrama de Fluxo de Dados.
- Houve não conformidades em relação ao uso dos artefatos do processo para documentar as características da arquitetura da aplicação. O artefato Documento de Arquitetura não havia sido utilizado, e a equipe acabou criando alguns

documentos adicionais para registrar algumas informações sobre a arquitetura do *software*, proliferando informações que deveriam estar em um único documento.

- O projeto apresentou dificuldades com o ciclo de vida iterativo, no que diz respeito ao refinamento de informações em cada fase. É necessário que, a cada mudança nos requisitos, seja identificado pela equipe do projeto os impactos para a documentação já existente, e que a documentação existente seja atualizada para contemplar as modificações. A equipe do projeto Z, a cada solicitação de mudança recebida, criava documentos adicionais para registrar as mudanças ocorridas, ao invés de atualizar a versão dos documentos já existentes (de requisitos, de arquitetura, de componentes etc.) que registravam as informações sobre o *software* sendo desenvolvido.

As não conformidades identificadas foram reportadas por meio do Relatório de Avaliação de Qualidade do Projeto Z para a fase, relacionando também os prazos e responsáveis. Uma não conformidade (referente ao caso de uso mal especificado e com erros de estruturação) não foi resolvida dentro do prazo acordado, tendo sido escalada pela equipe de qualidade para resolução pelo nível de gerenciamento superior ao da gerência do projeto. A resolução pelo nível de gerência não aconteceu prontamente, impactando o prazo de algumas ações que o projeto precisava implementar e que dependiam do correto detalhamento da especificação de caso de uso. A equipe de qualidade escalou, então, para a gerência sênior que procurou resolver a não conformidade, determinando que a equipe do projeto realizasse as correções necessárias o mais rapidamente possível. Este fato serviu para que a gerência sênior percebesse o valor da atividade de garantia de qualidade para o projeto e para que a organização A compreendesse que deveria fazer alguns ajustes para melhorar seu processo, incluindo o comprometimento gerencial com as ações de qualidade.

O projeto demandou a modificação de um padrão existente relacionado a banco de dados, de forma que o projeto pudesse utilizá-lo de forma adequada às suas necessidades. A demanda de adequação do padrão foi encaminhada ao SEPA, que trabalhou na adaptação do padrão existente. A equipe de qualidade auxiliou o SEPA nesse trabalho e auxiliou a equipe do projeto na aplicação do padrão adaptado para as necessidades do projeto Z.

O projeto havia planejado a realização de uma revisão de *walkthrough* nessa fase para verificar os requisitos com os clientes. A realização da revisão de *software* foi constatada

pela equipe de qualidade, conforme havia sido planejado no Plano de Garantia de Qualidade do Projeto Z. Essa revisão ocorreu após a implementação das correções nos casos de uso detalhados nesta fase, e possibilitou uma maior aproximação com o cliente.

Um ponto interessante observado no projeto Z é uma prática do projeto que poderia ser estendida a toda a organização, em relação ao protótipo de interface gráfica de usuário. O projeto Z se reúne com o cliente e especificamente conjuntamente uma versão da interface gráfica que depois de validado, é passado aos *designers* da organização para acabamento e documentação definitiva.

Fase de Construção

Os objetivos de qualidade na Fase de Construção visavam checar: se os casos de uso remanescentes estavam sendo detalhados (sem a ocorrência dos problemas identificados na fase anterior), projetados, implementados e testados; se estava sendo mantida a rastreabilidade entre os artefatos de requisitos, análise, *design* e implementação, se estava havendo o rastreamento entre as solicitações de mudança e os demais itens do projeto

Executada a auditoria para essa fase, as principais dificuldades apresentadas pelo projeto foram:

- A rastreabilidade entre os artefatos do projeto deixava ainda um pouco a desejar. Durante a Fase de Construção foi possível perceber que o projeto ficou mais disciplinado em relação ao desenvolvimento iterativo e à filosofia de refinamento de informações fase a fase.
- Em função da dificuldade acima, a análise do impacto das solicitações de mudanças recebidas dos usuários sobre os casos de uso existentes e sua documentação associada foi prejudicada. O projeto teve dificuldades para avaliar o impacto das solicitações de mudança sobre os requisitos.
- Não foi constatado que houve planejamento da sistemática de uso em paralelo do sistema antigo com o sistema novo, indicando que ações deveriam ser tomadas e os respectivos prazos, dentro das ações de implantação do *software*. O projeto sequer havia iniciado o Plano de Implantação.

Pode ser percebido que houve melhoria nas ações de detalhamento das especificações dos casos de uso remanescentes. Esse trabalho continuou contando com a atenção da equipe de qualidade também durante esta fase. E com relação as ações de implantação, a única ação realizada foi a participação do cliente na realização dos testes dos *builds* produzidos nesta fase, embora sem registro no Plano de Implantação, que não havia sido feito pelo projeto.

Os resultados foram reportados à equipe do projeto, que providenciou a correção de não conformidades, sem necessidade de escalar para os níveis superiores de gerenciamento. As não conformidades relatadas foram corrigidas dentro dos prazos acordados. O projeto conseguiu atingir os objetivos para a fase, inclusive iniciando os preparativos para a migração do *software* antigo para o novo, com o estabelecimento das ações e do cronograma das etapas de migração.

Fase de Transição

Na fase de Transição, os objetivos de qualidade para o projeto abrangeram avaliar a realização dos testes de regressão sobre o aplicativo, além de verificar o progresso das ações de implantação e se elas estavam sendo conduzidas a contento. A auditoria identificou que as atividades do projeto para implantação do produto estavam sendo realizadas conforme previsto no Plano de Implantação, desenvolvido na fase anterior. Também identificou que, embora o plano de migração da aplicação antiga para a nova estivesse em andamento, não estavam sendo registradas informações sobre esse processo, de forma que viesse auxiliar tanto o projeto Z, como outros projetos com características similares no futuro.

A auditoria também constatou que o projeto apresentou dificuldades para registrar mudanças efetuadas e manter atualizada a documentação do projeto em relação a essas mudanças, originadas da realização dos testes e correção dos defeitos. Uma vez reportados os resultados aos envolvidos e acordados com os responsáveis e respectivos prazos de resolução, o projeto providenciou as ações corretivas. A equipe de qualidade foi comunicada da resolução das não conformidades e constatou o fechamento das mesmas sem maiores problemas.

As métricas de qualidade para o projeto Z são informadas na Tabela 6.3.

Tabela 6.3 – Métricas de Qualidade do Projeto Z por fase

	Iniciação	Elaboração	Construção	Transição
Quantidade de avaliações de qualidade planejadas versus realizadas	100%	100%	100 %	100%
Quantidade de Não Conformidades identificadas (novas)	• 03 não conformidades de produto • 06 não conformidades de processo	• 04 não conformidades de produto • 03 não conformidades de processo	• 03 não conformidades de produto • 06 não conformidades de processo	• 05 não conformidades de produto • 02 não conformidades de processo
Não Conformidades resolvidas no prazo	100%	80%	100%	100%
Índice de Desvios	Nenhum desvio	Nenhum desvio	Nenhum desvio	Nenhum desvio

Lições Aprendidas de Qualidade

As Lições Aprendidas de Qualidade para o projeto como um todo são as seguintes:

- Há dificuldade pelos projetos em utilizar o conceito de iteração, e principalmente de trabalhar de forma iterativa, definindo o que será tratado na iteração, refinando documentação que já foi criada, e mantendo-a atualizada. Isso foi sobremaneira forte no projeto Z e constitui uma visão cultural que deve ser quebrada aos poucos, na medida em que a equipe vai adquirindo mais experiência com o processo iterativo.
- O número de não conformidades não pode ser visto de forma isolada, já que é necessário checar qualitativamente o tipo de não conformidade dentro da iteração e dentro da fase. As não conformidades identificadas numa fase e numa iteração foram resolvidas, mas foram identificadas novas não conformidades que dependiam essencialmente da necessidade do projeto naquela fase e da ênfase do uso das disciplinas de engenharia do UP na fase, já que cada fase possui objetivos específicos. As não conformidades de processo identificadas, apesar de em número menor na segunda iteração, abrangeram atividades mais críticas para a fase.
- A realização das avaliações de qualidade, em pontos de verificação específicos do projeto dentro da fase e da iteração, ajudaram a minimizar o impacto de não conformidades para a iteração ou fase seguinte.

6.4. Comparando Resultados entre Projetos

Descreveremos agora os resultados comparando organizações e projetos e os resultados observados de modo geral com a instanciação da disciplina. Os resultados estão categorizados da seguinte maneira: conclusões em relação aos aspectos culturais, conclusões acerca do Processo Unificado, conclusões sobre processos de desenvolvimento de *software* em geral e conclusões em relação à disciplina de Gerenciamento de Qualidade.

6.4.1. Conclusões em Relação aos Aspectos Culturais

As ações de qualidade tiveram maior apoio na organização B do que na organização A, ao longo deste trabalho. A organização B já havia percebido os benefícios de se trabalhar com qualidade de *software*, pois já possuía experiência com qualidade, em função da certificação ISO 9001. Inicialmente, o comprometimento dos diversos níveis gerenciais da organização A com as ações de qualidade foi bastante insipiente. A realização de ações corretivas e preventivas encontrava-se dependente da boa vontade das pessoas que faziam parte das equipes dos projetos. Esta situação foi mudando à medida que a organização A, percebia os ganhos reais da implantação do processo de qualidade (disciplina Gerenciamento da Qualidade), identificando problemas e gargalos no processo de desenvolvimento e ajudando a gerência a ter uma visão mais acurada do processo de desenvolvimento. Desde então começou a apoiar as ações de qualidade com maior comprometimento, sensibilizada inclusive em relação à realização de treinamentos.

A experiência da organização B com qualidade, sua visão de mercado e da importância da qualidade de *software* para manter-se competitiva no mercado de *software* auxiliou bastante a realização de ações corretivas e preventivas no menor tempo possível. A organização B não mediu esforços para apoiar as atividades de qualidade, uma vez que o produto sob manutenção no projeto avaliado era o principal ativo da empresa. A implantação da disciplina de qualidade na organização A está conduzindo essa organização no sentido de aprender a obter mais informações sobre o seu modo de trabalhar e a utilizar essas informações para melhorar seu processo de desenvolvimento, seu processo de gerenciamento e até mesmo outros processos que dão apoio ao processo de desenvolvimento, como processos de aquisição e adoção de novas tecnologias e ferramentas.

Outro aspecto interessante diz respeito à motivação das pessoas e à sua postura quando realizadas orientações pela equipe de qualidade. Observou-se uma melhora no aspecto motivacional, já que as dúvidas apresentadas pelos projetos, além de serem elucidadas em trabalho conjunto com a equipe de qualidade, geraram discussões e a realização de eventos (palestras/*workshops*) para aprofundar a discussão dos temas de interesse de outros projetos na organização que apresentavam as mesmas dificuldades.

6.4.2. Conclusões em Relação ao Processo Unificado

A distribuição das avaliações de qualidade ao longo do projeto possibilitou identificar alguns pontos chave do Processo Unificado em que a qualidade deve ser avaliada, antes que problemas impactassem atividades futuras. Esses pontos foram identificados com a experiência do processo de qualidade aplicada aos projetos: avaliar a qualidade dos requisitos e das especificações de casos de uso com os clientes, avaliar a arquitetura na documentação técnica avaliar os modelos de requisitos, análise, *design* e implementação, atuação sobre riscos, atuação sobre o planejamento do projeto e sobre as revisões gerenciais de fechamento de fase, por exemplo, são exemplos de pontos de controle dentro da abordagem do UP.

Os principais problemas identificados no processo de desenvolvimento UP e comum às organizações estudadas dizem respeito a:

- Diferenciação entre requisitos funcionais e não funcionais e regras de negócio, sua importância e utilização.
- Problemas na identificação e estruturação de casos de uso, indo da identificação equivocada de atores até a estruturação de casos de uso com problemas como não identificação de fluxos básicos e alternativos, “explosões” de casos de uso, mapeamento de casos de uso como itens de menu do *software* quando se faz documentação a partir de sistemas legados, etc.
- Planejamento do projeto, da fase e da iteração com base nos casos de uso especificados e no mapeamento de riscos realizado foi um grande ponto de gargalo.
- Dificuldades com o nível de detalhamento das informações dos artefatos em cada iteração, e seu posterior detalhamento nas iterações e fases seguintes. É

culturalmente complicado para as equipes este aspecto do desenvolvimento iterativo, de trabalhar com refinamentos sucessivos de informação e manter a documentação existente atualizada, sem necessidade de criar novos documentos.

- Problemas com a especificação e compreensão da arquitetura do *software* na visão 4+1 da UML. O modo como a arquitetura do UP é descrita, baseado na visão dos modelos UML 4+1, acaba-se tornando confuso para as pessoas que exercem os papéis cujas atividades são impactadas pela arquitetura do *software*, apesar de elas conhecerem e utilizarem UML.
- Dificuldades em realizar estimativas com base nos casos de uso especificados.
- Dificuldades em lidar com riscos por não serem identificados apropriadamente. Quando são identificados, não conseguem ser categorizados apropriadamente. Muitas vezes nem são estabelecidas estratégias de mitigação e de contingência.

As principais formas de prevenir as ocorrências destes gargalos é a realização de treinamentos adequados com os envolvidos, e o acompanhamento da equipe de qualidade e orientações sempre que o projeto necessitasse. Isto, entretanto, deveu-se ao perfil técnico das pessoas que compunham a equipe de qualidade em ambas as organizações, e que dispunham de condições de apoiar os projetos nas ações necessárias. É possível que isso não seja aplicável em outras organizações, uma vez que depende do perfil técnico das pessoas que executam atividades de SQA.

6.4.3. Conclusões em Relação à Disciplina de Gerenciamento de Qualidade

A identificação e correção de problemas mais cedo em um processo de desenvolvimento como o UP não apenas foi provido pela disciplina de gerenciamento da qualidade, como é fundamental nesse tipo de processo em que o encadeamento entre informações e atividades ao longo das iterações e fases é pré-requisito para a consecução das atividades de desenvolvimento de *software*.

As avaliações de qualidade em cada fase não abrangeram todo o conjunto de práticas e artefatos do projeto, mas tiveram como foco os aspectos mais críticos em cada fase. Procurou-se avaliar e refinar o planejamento da qualidade ao longo do projeto, a fim de que as avaliações de qualidade pudessem ocorrer em pontos de verificação importantes na fase, para minimizar o impacto das não conformidades ocorridas na fase e em relação a fases ou

iterações posteriores. Com o planejamento da qualidade para a iteração, com base nos resultados da fase anterior, as avaliações ficaram distribuídas em pontos importantes do processo cujas atividades impactavam as ações seguintes.

Outro ponto percebido nas duas organizações foi a postura com relação à realização de revisões. Nos projetos houve a realização de Revisão Técnica, especialmente sobre os requisitos, entretanto o que se observou foi que, em alguns casos, os revisores não possuíam a experiência e o conhecimento requerido para realizar a atividade. Com relação às Revisões Gerenciais, foi identificado que elas são fundamentais para o processo, uma vez que determinam o encerramento ou não de uma fase e a percepção sobre o andamento do projeto.

A disciplina se mostrou aderente às iterações e às fases, na medida em que pode ser utilizada pelos projetos, auxiliando-os a obter um nível melhor de qualidade. Mostrou-se também aplicável e adaptável a ciclos de manutenção, conforme pode ser percebido com a instanciação da mesma no projeto Y.

Foi percebida a necessidade de adaptação dos *checklists* usados inicialmente para as atividades gerenciais, especialmente as de planejamento e as reuniões de marco de ciclo de vida, que avaliam ou não se os critérios de fechamento da fase foram concluídos.

Capítulo 5

GERENCIAMENTO DE QUALIDADE:

UMA NOVA DISCIPLINA PARA O PROCESSO UNIFICADO

Este capítulo propõe uma nova disciplina para o Processo Unificado, denominada Gerenciamento da Qualidade, a qual visa tratar de aspectos de qualidade de processo de software.

No Processo Unificado (UP), a qualidade do produto é tratada de forma abrangente pela disciplina de Testes, e não é dada a relevância necessária à qualidade do processo de *software* (SMITH, 2003). Neste contexto, é proposta a disciplina de Gerenciamento da Qualidade (GQ) para o UP, que visa estabelecer um conjunto de ações que possam contribuir para a qualidade do processo de *software*, uma vez que a qualidade do produto já é tratada pela Disciplina de Testes. A disciplina proposta trata a qualidade do processo, em termos de verificação do uso do processo definido e da preocupação com a melhoria do processo utilizado pelos projetos, por meio do planejamento, da garantia, e da monitoração da qualidade ao longo do processo de desenvolvimento. As ações propostas nessa disciplina visam melhorar a abordagem de qualidade de processo de *software* do UP, à medida que estão alinhadas às práticas propostas por alguns dos principais modelos e padrões de qualidade do mercado.

Há vários argumentos que podem ser citados para a criação de uma disciplina para o gerenciamento da qualidade para o Processo Unificado:

- Smith (2003), conforme relato no capítulo anterior, sugere a criação de uma disciplina no UP para tratar qualidade do processo.
- Ulferts (2005) reconhece a existência de uma lacuna no UP, uma vez que inexiste uma disciplina para tratar SQA.
- Apenas incluir as atividades de qualidade propostas na disciplina de gerenciamento de projetos não seria razoável, a medida em que esta disciplina trata do aspecto de qualidade apenas do ponto de vista do gerente do projeto, enquanto a disciplina proposta considera o contexto organizacional da qualidade.

- As atividades de garantia de qualidade propostas por meio da disciplina GQ, pela natureza da atividade de SQA, devem ser realizadas atendendo os critérios de independência e objetividade do avaliador em relação ao objeto de avaliação. Existiria uma contradição em tratar SQA dentro da própria disciplina de gerenciamento de projetos, uma vez que o gerente de projeto, principal papel executor das atividades nessa disciplina, por seu envolvimento com o projeto, não atenderia a esses dois critérios básicos de SQA elencados acima.

5.1. Objetivos da Disciplina

A disciplina de Gerenciamento da Qualidade está estruturada de modo a atender aos seguintes objetivos:

- Prover mais visibilidade para a qualidade do processo de *software* no UP, destacando a importância da mesma em cada iteração.
- Tratar sistematicamente as não conformidades no processo, facilitando sua identificação, seu relato aos envolvidos, a identificação de ações corretivas e, quando possível, de ações preventivas. Isto não é tratado no UP, segundo Manzoni & Price (2003), e a identificação de ações corretivas e preventivas é uma ação recomendada pelo CMMI (2002a; 2002b) e ISO 9001 (2000).
- Possibilitar que as ações de qualidade estejam alinhadas aos objetivos de qualidade especificados para a organização (CMMI, 2002a; CMMI, 2002b). A execução das atividades de garantia de qualidade é realizada de acordo com o planejamento da qualidade de *software* para o projeto e para a iteração. O planejamento da qualidade, por sua vez, considera os objetivos de qualidade especificados para a organização, para o projeto e para a iteração.
- Possibilitar a utilização de métricas de processo relacionadas aos objetivos de qualidade para a iteração, em cada fase, para o projeto e para a organização, com base nos conceitos do padrão IEEE 1061 (1998).
- Estabelecer o registro das lições aprendidas de qualidade de *software* ao longo do processo. Esta atividade, aliada ao uso sistemático de métricas e às atividades de SQA nos projetos, facilitará a identificação da aderência de boas práticas de engenharia de *software* do projeto em relação ao processo configurado a partir do *framework* do UP.

5.2. Ações Organizacionais para o Gerenciamento da Qualidade

Para que a nova disciplina proposta para o UP, Gerenciamento da Qualidade, exerça efetivamente seu papel nos projetos de *software*, é necessário que seja estabelecido um conjunto de atividades no âmbito organizacional, que auxilie a estruturação da função de qualidade na organização, definindo:

- Como serão implementadas as ações de qualidade de *software* na organização;
- Qual a estrutura de apoio às atividades de qualidade ao longo do processo de *software*.

Preferencialmente, estas atividades devem ser realizadas necessariamente antes da instanciação da disciplina em nível de projeto. São elas:

- Definição dos objetivos de qualidade e de políticas de qualidade para a organização: as Políticas de Qualidade podem ser expressas no Manual de Qualidade, conforme proposto pela ISO 9001 (2000). A organização deve estabelecer seus objetivos de qualidade e buscar alcançá-los. Esses objetivos devem ser mensuráveis e coerentes com as Políticas de Qualidade.
- Estabelecimento de padrões de qualidade organizacionais: os padrões de qualidade são definidos para revelar as melhores práticas de uma organização, auxiliando o trabalho dos projetos com um nível de qualidade adequado às suas necessidades (SOMMERVILLE, 2003).
- Definição de métricas organizacionais de qualidade: as métricas organizacionais de qualidade são definidas com base nos objetivos organizacionais de qualidade. Elas auxiliam a organização a verificar como estão sendo conduzidas as ações de qualidade e se essas ações estão atingindo os objetivos de qualidade definidos.
- Disseminação da cultura de qualidade: estabelecimento de uma cultura de qualidade na organização possibilita que todos os envolvidos com o processo de *software* estejam comprometidos com a qualidade em cada ação realizada (SCHULMEYER & MCMANUS, 1999), sejam eles gerentes, engenheiros de *software* e até mesmo clientes.

Para efeito de simplificação, as referências ao conjunto de artefatos constituído pelo Manual da Qualidade, Procedimentos, e Padrões de Qualidade organizacionais serão feitas ao artefato denominado de Documentos Organizacionais de Qualidade.

5.3. Papéis e Responsabilidades

São propostos dois novos papéis para o UP: o Gerente de Qualidade e o Engenheiro de Qualidade. A criação desses dois novos papéis visa suportar a atuação da qualidade na organização. De acordo com Sommerville (2003), a equipe responsável por gerenciar a qualidade deve ser independente e se reportar à Gerência Superior, no nível acima da gerência de projeto. Assim, o Gerente de Qualidade e o Engenheiro de Qualidade possuem como principais características: a responsabilidade pelo gerenciamento da qualidade na organização e a independência em relação aos projetos que estão sendo avaliados. O Gerente de Qualidade e o Engenheiro de Qualidade devem reportar-se ao Gerenciamento Sênior da organização, garantindo a existência de um canal de comunicação independente para relatar e tratar não conformidades identificadas nos projetos de *software* (CMMI, 2002a; CMMI, 2002b; UNHELKAR, 2003).

O Gerente de Qualidade deve administrar a qualidade de *software* na organização, mantendo os objetivos organizacionais da qualidade alinhados aos objetivos de negócios organizacionais e monitorando se os projetos estão alcançando os objetivos previstos de qualidade. Deve possuir experiência com gestão de processos e de pessoas, conhecimento da organização e conhecimento sobre modelos e padrões de qualidade de *software*. Deve estimular os envolvidos com o processo de desenvolvimento a cultivarem o compromisso com a qualidade de *software* em suas ações.

O Engenheiro de Qualidade é responsável por avaliar a qualidade dos projetos de *software*, em termos de processo e de produto, verificando a conformidade das práticas dos projetos aos procedimentos e padrões estabelecidos. Deve ter forte embasamento em engenharia de *software*, mormente em modelos, padrões e técnicas de SQA, e também com o processo de desenvolvimento de *software*.

Ambos, Gerente de Qualidade e Engenheiro de Qualidade, devem estar cientes do papel da qualidade de *software* no contexto organizacional e dos projetos, podendo inclusive agir como consultores de engenharia de *software* para projetos da organização, provendo orientação sobre padrões organizacionais, engenharia de *software* e qualidade de *software*.

5.4. Fluxo da Disciplina de Gerenciamento da Qualidade

O fluxo da disciplina Gerenciamento da Qualidade é apresentado na Figura 5.1.

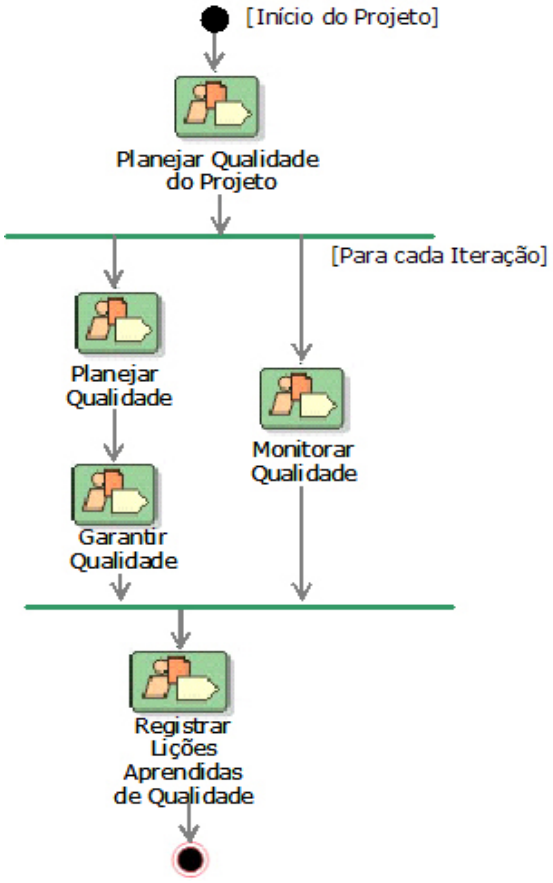

Figura 5.1 - Disciplina Gerenciamento da Qualidade

Cada disciplina do UP é composta por macro-atividades (*workflow detail*). As macro-atividades da disciplina proposta são:

i. Planejar Qualidade do Projeto;

ii. Planejar Qualidade;

iii. Garantir Qualidade;

iv. Monitorar Qualidade; e

v. Registrar Lições Aprendidas de Qualidade.

O conjunto de atividades referentes aos itens (*ii*), (*iii*) e (*iv*) repete-se a cada iteração do UP. As atividades de qualidade ao longo de cada iteração são realizadas de acordo com os objetivos da fase na qual a iteração está inserida.

A seguir, são detalhadas as macro-atividades e atividades da disciplina. Cada uma das atividades constantes da disciplina de Gerenciamento da Qualidade é descrita em termos de propósitos, passos, artefatos de entrada e de saída, freqüência de execução da atividade, e atores envolvidos, seguindo a estrutura do UP.

5.4.1. Macro-atividade: Planejar Qualidade do Projeto

O propósito dessa macro-atividade é garantir que as informações necessárias para o planejamento da qualidade do projeto estejam estabelecidas e registradas no Plano de Garantia de Qualidade. A Figura 5.2 detalha as atividades da macro-atividade Planejar Qualidade do Projeto, que devem ser executadas em paralelo com as atividades de planejamento do próprio projeto.

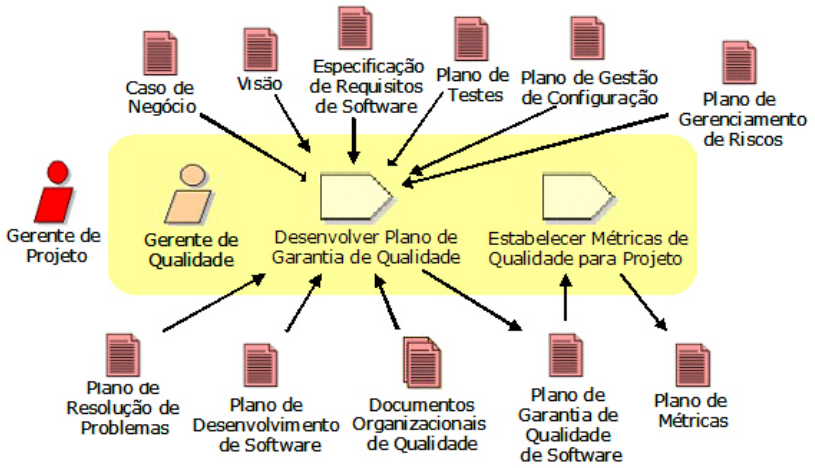

Figura 5.2 - Macro-Atividade: Planejar Qualidade do Projeto

Descrição

Nesta macro-atividade são definidos os objetivos de qualidade de *software* para o projeto e são estabelecidas as métricas de qualidade associadas a cada objetivo de qualidade especificado, para que seja possível monitorar se os objetivos foram atingidos. O Gerente de Qualidade planeja, a partir dos objetivos de qualidade, que atividades de qualidade serão executadas ao longo do projeto e constrói o cronograma dessas atividades, desenvolvendo o Plano de Garantia de Qualidade do projeto. Ele também orienta os integrantes do projeto a respeito dos padrões de qualidade e das práticas de engenharia de *software* aplicáveis ao projeto, instanciando os padrões organizacionais para a realidade do projeto. Por fim, o Plano de Garantia de Qualidade desenvolvido para o projeto é aprovado pelos *stakeholders*.

Agenda

As atividades que compõem a macro-atividade Planejar Qualidade do Projeto são realizadas em paralelo com as atividades de planejamento do projeto.

Recomendações

As atividades são recomendadas para quaisquer projetos e têm sua execução amparada pela Política de Qualidade definida e seguida pela organização.

Equipe

O Gerente de Qualidade e o Engenheiro de Qualidade são responsáveis por definir, a partir de discussões conjuntas com a equipe de projeto e os *stakeholders*, qual a visão de qualidade estabelecida para o projeto. Eles atuam como consultores de qualidade de *software* para a equipe do projeto.

Atividades

A macro-atividade Planejar Qualidade do Projeto é composta pelas seguintes atividades:

- Desenvolver Plano de Garantia de Qualidade
- Estabelecer Métricas de Qualidade para o Projeto

5.4.1.1. Desenvolver Plano de Garantia de Qualidade

O Quadro 5.1 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Desenvolver Plano de Garantia de Qualidade</th></tr>
<tr><td colspan="2">Propósito: desenvolver um plano que trate dos aspectos de garantia de qualidade de software no projeto.</td></tr>
<tr><td colspan="2">Passos:
- definir os objetivos de qualidade para o projeto;
- definir papéis e responsabilidades relacionados à garantia de qualidade;
- definir as atividades de garantia de qualidade e o cronograma;
- instanciar padrões de qualidade para o projeto;
- coordenar as ações de qualidade junto aos demais planos de projeto.</td></tr>
<tr><td>Artefatos de Entrada:
- Caso de Negócio
- Plano de Gestão de Configuração
- Plano de Resolução de Problemas
- Plano de Gerenciamento de Riscos
- Plano de Testes
- Documento de Visão
- Documentos Organizacionais de Qualidade
- Plano de Desenvolvimento de Software
- Especificação de Requisitos de Software</td><td>Artefatos Resultantes:
- Plano de Garantia de Qualidade de Software</td></tr>
<tr><td colspan="2">Freqüência: nas reuniões de planejamento do projeto.</td></tr>
<tr><td colspan="2">Papel: Gerente de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Planejar Qualidade do Projeto</td></tr>
</table>

Quadro 5.1 – Atividade: Desenvolver Plano de Garantia de Qualidade

Definir os Objetivos de Qualidade para o Projeto

Com base nos Documentos Organizacionais de Qualidade (Políticas de Qualidade, Padrões de Qualidade) e nos requisitos de qualidade do projeto, o Gerente de Qualidade define os objetivos de qualidade para o projeto, em conjunto com a equipe do projeto. Os objetivos de qualidade do projeto devem ser válidos dentro do escopo definido para o projeto, guiando as ações que serão realizadas para que se tenha um processo adequado ao projeto e um produto atendendo os requisitos dos *stakeholders*.

Os requisitos de qualidade para o projeto, que podem estar categorizados segundo o modelo FURPS+ (GRADY, 1992), são definidos a partir:

- da experiência da organização;
- dos modelos de qualidade da organização aplicáveis ao projeto;
- de alguma exigência legal ou regulamentadora aplicável ao projeto;
- de restrições contratuais;
- das expectativas externadas pelo usuário e/ou cliente em relação à qualidade de *software*.

O Plano de Garantia de Qualidade de *Software* deve ser atualizado com as informações sobre os objetivos de qualidade, e sobre a estrutura organizacional.

Definir Papéis e Responsabilidades relacionados À Garantia de Qualidade

Devem estar descritos no Plano os papéis envolvidos e suas responsabilidades no contexto das atividades de garantia de qualidade de *software*. Segundo o RUP (2003), o SEPA (*Software Engineering Process Authority*) pode ser constituído por uma única pessoa, por um gerente ou por uma equipe de representantes, mas não especifica qual é o relacionamento entre essa entidade e a estrutura de projetos da organização. Um requisito para execução de ações de qualidade é a independência do avaliador em relação ao objeto avaliado. Neste sentido, a atuação do Gerente de Qualidade e do Engenheiro de Qualidade, definidos como papéis completamente independentes da estrutura de projetos, se apresenta como adequada para executar as revisões e auditorias nos projetos de *software*.

Definir as Atividades de Garantia de Qualidade e o Cronograma

Os objetivos de qualidade a serem atingidos auxiliam o Gerente de Qualidade a definir que atividades de garantia de qualidade devem ser realizadas ao longo do processo de desenvolvimento de *software*. As atividades de qualidade a serem realizadas no projeto são definidas pelo Gerente de Qualidade, em conjunto com a equipe de projeto. As atividades de qualidade propostas nessa

disciplina abrangem as Revisões de *Software*, segundo o padrão IEEE 1028 (1997), além de avaliações de *benchmarking* do processo (SWEBOK, 2004), conforme a seguir:

- Auditorias de garantia de qualidade: avaliam os projetos de *software* para verificar a adequação de processos e de padrões às necessidades dos projetos (IEEE 1028, 1997).
- Revisões Técnicas (IEEE 1028, 1997): avaliam artefatos técnicos.
- Revisões Gerenciais (IEEE 1028, 1997): avaliam o status das atividades gerenciais realizadas versus planejadas ao longo do processo de *software*.
- Inspeções (IEEE 1028, 1997): são revisões formais, realizadas sobre código e outros artefatos.
- *Walkthrough* (IEEE 1028, 1997) – revisões de caráter menos formal, mais utilizada para validação de requisitos com usuários.
- Avaliações qualitativas de melhoria de processo (*assessments*) (SWEBOK, 2004): avaliam o uso do Processo Unificado em seus projetos, comparando suas práticas em relação às práticas exigidas por modelos de qualidade específicos.

As ações de Auditorias, Inspeções, *Walkthroughs*, Revisões Técnicas e Revisões Gerenciais podem utilizar os *checkpoints* propostos no UP para validar artefatos do processo.

O cronograma das atividades de qualidade para o projeto é definido, estabelecendo em cada fase do projeto as atividades que serão realizadas para alcançar os objetivos de qualidade. Informações sobre as atividades de qualidade planejadas e o cronograma associado devem ser registradas no Plano de Garantia de Qualidade de *Software*.

Instanciar os Padrões de Qualidade adequados ao Projeto

As políticas, os procedimentos e os padrões organizacionais de qualidade, definidos quando do estabelecimento das ações de qualidade no âmbito organizacional, são utilizados pelo Engenheiro de Qualidade na tarefa de, em conjunto com a equipe de projeto, estabelecer os padrões de qualidade aplicáveis ao projeto.

No trabalho de seleção dos padrões de qualidade aplicáveis ao projeto, são considerados os seguintes fatores:

- características dos projetos;
- domínio da aplicação;
- tecnologias utilizadas no desenvolvimento da aplicação;
- técnicas a serem utilizadas ao longo do processo de desenvolvimento.

Os padrões de qualidade instanciados para o projeto são utilizados como critérios para a avaliação de qualidade.

Coordenar as Ações de Qualidade junto aos demais Planos do Projeto

O Plano de Garantia de Qualidade referencia uma série de outros planos que descrevem padrões do projeto e como processos de suporte devem ser executados. Essas informações auxiliam na definição das atividades de garantia de qualidade que devem ser executadas, e sua freqüência adequada. Os planos referenciados incluem:

- Plano de Métricas
- Plano de Gerenciamento de Riscos
- Plano de Resolução de Problemas
- Plano de Gestão de Configuração
- Plano de Desenvolvimento de *Software*
- Plano de Iteração
- Plano de Testes

5.4.1.2. Estabelecer Métricas de Qualidade para o Projeto

O Quadro 5.2 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Estabelecer métricas de qualidade de software para o projeto</th></tr>
<tr><td colspan="2">Propósito: estabelece, para cada objetivo de qualidade definido para o projeto, as métricas de qualidade que serão utilizadas para avaliar se os objetivos de qualidade estão sendo atingidos.</td></tr>
<tr><td colspan="2">Passos:
- identificar as métricas de qualidade de software;
- implementar as métricas de qualidade de software.</td></tr>
<tr><td>Artefatos de Entrada:
- Plano de Garantia de Qualidade de Software</td><td>Artefatos Resultantes:
- Plano de Métricas</td></tr>
<tr><td colspan="2">Freqüência: sempre que forem definidos objetivos de qualidade do projeto.</td></tr>
<tr><td colspan="2">Papel: Gerente de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Planejar Qualidade do Projeto</td></tr>
</table>

Quadro 5.2 – Atividade: Estabelecer métricas de qualidade de *software* para o projeto

Identificar as Métricas de Qualidade de *Software*

Métricas podem ser identificadas para os objetivos de qualidade, a fim de verificar se os objetivos estão sendo atendidos. Para cada objetivo de qualidade, pode-se associar uma ou mais métricas. Em seguida, para cada métrica estabelecida, identifica-se: um valor esperado, um valor crítico e uma faixa de valores limite no qual o valor associado à métrica deverá ou poderá variar ao longo do

processo de desenvolvimento. Depois, deve ser avaliado o custo de implementação de cada métrica, em termos de procedimentos de coleta de dados, cálculo de métricas, e aplicação, interpretação e análise dos resultados. Avalia-se o benefício de aplicação das métricas. Finalmente, todos os aspectos de implementação das métricas devem ser discutidos com os *stakeholders* para que sejam definidas, em comum acordo, quais as métricas que serão utilizadas (IEEE 1061, 1998; CMMI, 2002a), registrando as informações no Plano de Métricas.

Implementar as Métricas de Qualidade de *Software*

Neste passo, são definidos os procedimentos de coleta de dados para as métricas. Esses procedimentos envolvem determinar, para cada métrica, que dados serão coletados, como eles serão analisados, registrados e reportados aos interessados. Os dados coletados sobre as métricas devem ser analisados em relação aos valores esperados para cada uma delas. Deve ser verificado, por meio da análise das métricas, se os objetivos de qualidade estão sendo atingidos (IEEE 1061, 1998; CMMI, 2002a). As definições sobre as métricas de qualidade devem ser registradas no Plano de Métricas.

5.4.2. Macro-Atividade: Planejar Qualidade da Iteração

O propósito desta macro-atividade é definir como a qualidade de *software* será tratada na iteração, de acordo com cada fase do Processo Unificado, segundo os objetivos da iteração e da fase. A Figura 5.3 mostra as atividades para o planejamento da qualidade na iteração.

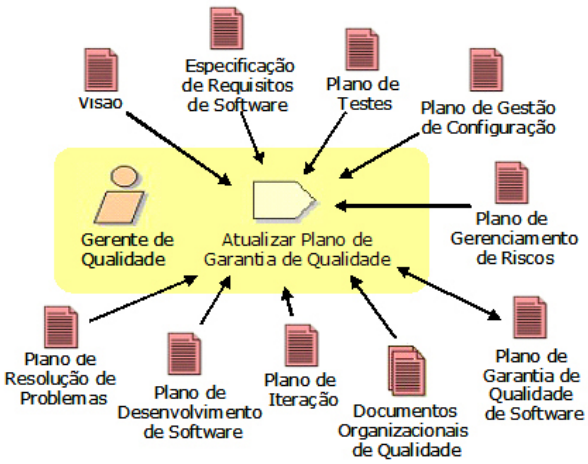

Figura 5.3 - Macro-atividade: Planejar Qualidade da Iteração

Descrição

Paralelo à realização do planejamento da iteração, há o planejamento da qualidade na iteração. Ao se iniciar uma iteração, devem ser identificados, dentre os objetivos de qualidade do projeto levantados durante o planejamento do projeto, aqueles que serão abordados durante a

iteração. Em seguida, são elencadas as atividades de qualidade de *software* que serão executadas durante a iteração, atualizando-se o Plano de Garantia de Qualidade.

Agenda

Esta atividade é realizada em paralelo com as atividades de planejamento da iteração, atualizando o Plano de Garantia de Qualidade de *Software* do projeto com as informações sobre as atividades de qualidade.

Recomendações

Esta atividade é recomendada para quaisquer projetos da organização. Pode ser opcional para projetos de curta duração.

Equipe

O Gerente de Qualidade define, a partir de discussões conjuntas com a equipe de projeto e com os *stakeholders*, qual a visão de qualidade estabelecida para a iteração.

Atividades

A atividade Atualizar Plano de Garantia de qualidade é apresentada a seguir.

5.4.2.1. Atualizar Plano de Garantia de Qualidade

O Quadro 5.3 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Atualizar Plano de Garantia de Qualidade</th></tr>
<tr><td colspan="2">Propósito: Identificar os objetivos de qualidade que devem ser atingidos durante a iteração e quais as atividades devem ser conduzidas.</td></tr>
<tr><td colspan="2">Passos:
- Identificar os objetivos de qualidade da iteração;
- Identificar as ações de qualidade para a iteração;
- Atualizar o Plano de Garantia de Qualidade de Software.</td></tr>
<tr><td>Artefatos de Entrada:
- Documento de Visão
- Plano de Gerenciamento de Riscos
- Plano de Gestão de Configuração
- Especificação de Requisitos de Software
- Plano de Testes
- Plano de Garantia de Qualidade de Software
- Plano de Desenvolvimento de Software
- Plano de Iteração
- Documentos Organizacionais de Qualidade
- Plano de Resolução de Problemas</td><td>Artefatos Resultantes:
- Plano de Garantia de Qualidade de Software.</td></tr>
<tr><td colspan="2">Freqüência: a cada iteração.</td></tr>
<tr><td colspan="2">Papel: Gerente de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Planejar Qualidade da Iteração</td></tr>
</table>

Quadro 5.3 – Atividade: Atualizar Plano de Garantia de Qualidade

Identificar os Objetivos de Qualidade da Iteração

Os objetivos da iteração dependem dos objetivos do projeto tratados na fase na qual a iteração está inserida. Portanto, faz-se necessário identificar os objetivos de qualidade que devem ser alcançados na iteração, selecionando-os dentre os já especificados quando houve o planejamento do projeto/fase, ou mesmo identificando outros novos ou modificando alguns dos objetivos já existentes.

Identificar as Ações de Qualidade para a Iteração

São identificadas ações de garantia de qualidade a serem realizadas na iteração visando atender as necessidades do projeto, e atender os objetivos de qualidade da iteração. São planejadas as avaliações de qualidade a serem realizadas na iteração, definindo-se os critérios de avaliação. São definidas, quando for o caso, ações preventivas, discutindo com a equipe do projeto como as ações serão implementadas.

Atualizar o Plano de Garantia de Qualidade de *Software*

As informações sobre os objetivos de qualidade da iteração e as atividades de qualidade, a serem realizadas para garantir a qualidade do processo de *software*, são registradas no Plano de Garantia de Qualidade de *Software* do projeto, inclusive atualizando os cronogramas, se necessário.

5.4.3. Macro-Atividade: Garantir Qualidade da Iteração

O propósito desta macro-atividade é garantir a qualidade das atividades de desenvolvimento de *software* realizadas na iteração. A Figura 5.4 abaixo apresenta o detalhamento da macro-atividade.

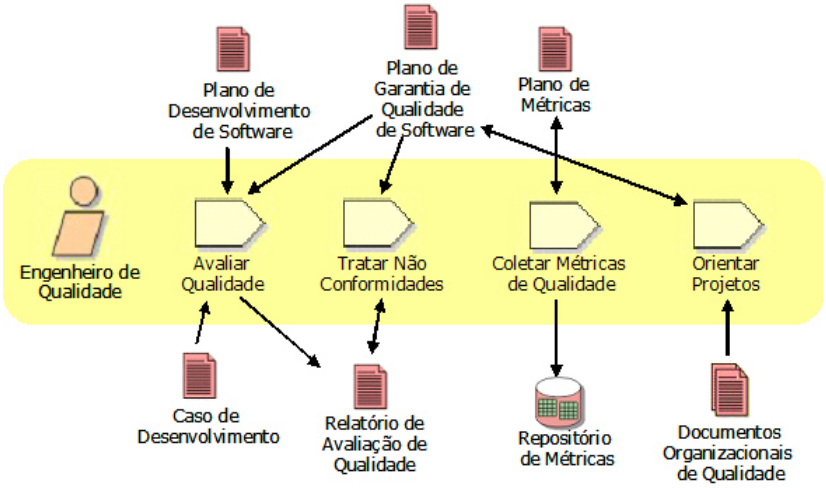

Figura 5.4 – Macro-atividade: Garantir Qualidade da Iteração

Descrição

A garantia de qualidade é realizada por meio de avaliações executadas para determinar se requisitos técnicos de desenvolvimento foram estabelecidos, e se produtos e processos estão em conformidade com os requisitos técnicos (SCHULMEYER & MCMANUS, 1999). A avaliação de qualidade pode ser conduzida utilizando a técnica de Auditoria (Apêndice A). Ao serem utilizadas ao longo de cada iteração para verificar a qualidade do processo de *software*, as auditorias podem suprir a deficiência relatada por Smith (2003) (capítulo 4).

Uma das atividades da garantia de qualidade é a identificação de não conformidades. O artefato Relatório de Avaliação da Qualidade do projeto registra as informações sobre as avaliações realizadas, os itens avaliados, as não conformidades identificadas, os graus de severidade a elas associados, os responsáveis e os prazos de resolução. As não conformidades identificadas e reportadas devem ser acompanhadas pelo Engenheiro de Qualidade até seu desfecho. Ainda são coletadas métricas de qualidade da iteração e fornecidas orientações sobre padrões e práticas de engenharia de *software*, quando necessário.

Agenda

As atividades de garantia de qualidade ocorrem ao longo da iteração.

Recomendações

As atividades são recomendadas para quaisquer projetos e têm sua execução amparada pela Política de Qualidade definida e seguida pela organização.

Equipe

O Engenheiro de Qualidade possui atuação fundamental nesta macro-atividade. Ele deve conhecer políticas, modelos e técnicas de qualidade. Deve atuar como um orientador/mentor dos projetos e como um avaliador independente em relação aos projetos sob avaliação.

Atividades

A macro-atividade de Garantir Qualidade na iteração possui as seguintes atividades:

- Avaliar qualidade
- Tratar não conformidades
- Coletar métricas de qualidade
- Orientar projetos

5.4.3.1. Avaliar Qualidade

O Quadro 5.4 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Avaliar qualidade</td></tr>
<tr><td colspan="2">Propósito: avaliar a iteração em relação às práticas do processo de desenvolvimento, uso dos artefatos e aplicação de técnicas ao processo, identificando e registrando as não conformidades.</td></tr>
<tr><td colspan="2">Passos:
- preparar avaliação de qualidade;
- realizar avaliação de qualidade;
- identificar e registrar não conformidades;
- reportar resultados da avaliação de qualidade.</td></tr>
<tr><td>Artefatos de Entrada:
- Plano de Garantia de Qualidade de Software
- Plano de Desenvolvimento de Software
- Caso de Desenvolvimento</td><td>Artefatos Resultantes:
- Relatório de Avaliação de Qualidade.</td></tr>
<tr><td colspan="2">Freqüência: a cada iteração, conforme cronograma estabelecido no Plano de Garantia de Qualidade de Software do Projeto ou quando for solicitado pela equipe de projeto.</td></tr>
<tr><td colspan="2">Papel: Engenheiro de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Garantir Qualidade da Iteração</td></tr>
</table>

Quadro 5.4 – Atividade: Avaliar qualidade

Preparar Avaliação de Qualidade

O objetivo desta atividade é preparar as informações que serão necessárias para realizar a avaliação de qualidade. Baseando-se nos objetivos da iteração e da fase, na qual é realizada a avaliação, devem ser definidas as atividades do processo que serão revisadas, a partir dos artefatos gerados, e que técnicas e métodos serão utilizados.

Realizar Avaliação de Qualidade

A avaliação de qualidade contempla a garantia de qualidade em relação aos seguintes pontos:

- avalia se atividades do processo definido estão sendo seguidas pela equipe do projeto;
- verifica se o processo e as atividades são adequados às necessidades do projeto;
- avalia se os artefatos de apoio ao processo estão sendo utilizados e se estão sendo elaborados de forma adequada;

Podem participar das avaliações de qualidade do projeto: o Gerente de Projetos, membros da equipe do projeto, a equipe de qualidade alocada ao projeto, e quaisquer *stakeholders* interessados na avaliação de qualidade. A seleção dos participantes de cada avaliação é feita durante o planejamento das avaliações, em cada iteração, de acordo com os objetivos de qualidade a serem alcançados.

O Engenheiro de Qualidade também poderá participar de outras atividades de qualidade executadas durante a iteração, como as Revisões Técnicas (revisão de artefatos como Especificação de Requisitos de *Software* e de Código) e as Revisões Gerenciais (Revisões de

Planejamento do Projeto e Revisões de Marcos de Ciclo de Vida). Caso não participe, verifica se há evidências de que essas Revisões foram executadas e que atingiram os seus objetivos.
O artefato Relatório de Avaliação de Qualidade contém todas as informações relacionadas às avaliações de qualidade realizadas para o projeto e seus resultados.

Identificar e Registrar não Conformidades
Quando é identificada a falta de aderência de determinado item que está sob garantia de qualidade (atividade, artefato, técnica etc) em relação aos padrões de qualidade, diz-se que há uma não conformidade (NC). As não conformidades identificadas podem ser categorizadas como NC de processo e NC de produto, com a identificação dos pontos do processo em que elas ocorreram. A cada NC deverá ser atribuído um grau de severidade, de acordo com a sua criticidade para o projeto, e são identificados os responsáveis pela resolução, as ações corretivas e os prazos em que as não conformidades deverão estar solucionadas. Todas essas informações são registradas pelo Engenheiro de Qualidade no Relatório de Avaliação de Qualidade. A ocorrência de não conformidades é comunicada à equipe do projeto para que sejam tomadas as providências cabíveis. O Engenheiro de Qualidade deve verificar com o Gerente de Projeto qual o potencial de uma não conformidade ocorrer novamente e quais as suas possíveis causas. O Engenheiro de Qualidade deve avaliar se as não conformidades identificadas podem indicar alguma tendência do projeto com relação à qualidade e registrar isto no Relatório de Avaliação da Qualidade.

Reportar Resultados da Avaliação de Qualidade
Os resultados das avaliações de qualidade são reportados à equipe do projeto e aos *stakeholders* afetados pela avaliação.

5.4.3.2. Tratar não Conformidades

O Quadro 5.5 apresenta o detalhamento desta atividade.

Reportar não Conformidades
As não conformidades identificadas no projeto são reportadas ao gerente de projeto pelo engenheiro de qualidade.
No processo de solução de uma não conformidade pode haver necessidade de se efetuar mudanças em modelos e padrões, que deverá ser feito pelo SEPA (*Software Process Engineering Authority*).

<table>
<tr><th colspan="2">Atividade: Tratar não conformidades</th></tr>
<tr><td colspan="2">Propósito: possibilitar que as não conformidades identificadas sejam solucionadas, escalonando-as para resolução quando necessário.</td></tr>
<tr><td colspan="2">Passos:
- reportar não conformidades;
- acompanhar resolução de não conformidades;
- escalar não conformidades não resolvidas;
- encerrar não conformidades.</td></tr>
<tr><td>Artefatos de Entrada:
- Plano de Garantia de Qualidade de Software
- Relatório de Avaliação de Qualidade</td><td>Artefatos Resultantes:
- Relatório de Avaliação de Qualidade</td></tr>
<tr><td colspan="2">Freqüência: em cada iteração</td></tr>
<tr><td colspan="2">Papel: Engenheiro de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Garantir Qualidade da Iteração</td></tr>
</table>

Quadro 5.5 – Atividade: Tratar Não Conformidades

Acompanhar Resolução de não Conformidades

O Engenheiro de Qualidade deve acompanhar as não conformidades para verificar se foram solucionadas pela equipe do projeto, de acordo com os prazos estabelecidos para resolução e com a severidade associada a cada não conformidade.

Escalar não Conformidades não Resolvidas

O Engenheiro de Qualidade deve identificar as não conformidades ainda não resolvidas e que extrapolaram seu prazo de resolução. O Engenheiro de Qualidade é responsável por escalar essas não conformidades abertas para o nível gerencial superior adequado, conforme previsto em procedimento.

Encerrar não Conformidades

Após verificar a solução das não conformidades identificadas, o Engenheiro de Qualidade providencia o encerramento das mesmas.

5.4.3.3. Coletar Métricas de Qualidade

O Quadro 5.6 apresenta o detalhamento desta atividade.

Coletar Métricas

De acordo com os procedimentos de coleta definidos para as métricas estabelecidas no Planejamento do Projeto, os dados para cálculo são coletados pelo Engenheiro de Qualidade e registrados no Plano de Métrica. As medidas podem ser coletadas de forma manual ou automatizada.

Atividade: Coletar métricas de qualidade	
Propósito: obter métricas relacionadas com a qualidade de *software* para suportar a tomada de decisões a respeito do processo de qualidade e de seus resultados.	
Passos: - coletar métricas;	
Artefatos de Entrada: - Plano de Métricas; - Plano de Garantia de Qualidade de *Software*	**Artefatos Resultantes**: - Repositório de Métricas. - Plano de Métricas
Freqüência: a cada iteração.	
Papel: Engenheiro de Qualidade	
***Macro-Atividade*:** Gerenciamento da Qualidade Garantir Qualidade da Iteração	

Quadro 5.6 – Atividade: Coletar Métricas de Qualidade

5.4.3.4 Orientar Projetos

O Quadro 5.7 apresenta o detalhamento desta atividade.

Atividade: Orientar projetos	
Propósito: prover orientação aos projetos de *software*, ao longo da iteração, com relação à engenharia de *software* e à qualidade de *software*.	
Passos: - identificar necessidade de orientar projetos; - prover orientação aos projetos.	
Artefatos de Entrada: - Documentos organizacionais de qualidade; - Plano de Garantia de Qualidade de *Software*; - Plano de Desenvolvimento de *Software*.	**Artefatos Resultantes**: - Plano de Garantia de Qualidade de *Software*.
Freqüência: ao longo da iteração.	
Papel: Engenheiro de Qualidade	
***Macro-Atividade*:** Gerenciamento da Qualidade Garantir Qualidade da Iteração	

Quadro 5.7 – Atividade: Orientar projetos

Identificar Necessidade de Orientar Projetos

A identificação da necessidade de orientar projetos em relação à qualidade de *software* pode ser originada de duas formas: o Engenheiro de Qualidade pode identificar a necessidade a partir dos resultados das avaliações de qualidade realizadas; ou a necessidade pode ser externada pela própria equipe de projeto.

Prover Orientação aos Projetos

A orientação aos projetos pode ser realizada de duas formas. A primeira caracteriza-se quando a própria equipe do projeto solicita a orientação da equipe de qualidade. A segunda é identificada pela equipe de qualidade com base em resultados da avaliação de qualidade realizada no projeto. A

orientação pode ser concretizada por meio de técnicas de *mentoring* ou mesmo realização de reuniões. Informações sobre as ações de orientação realizadas podem ser registradas no artefato Plano de Garantia de Qualidade. A equipe de qualidade realiza *mentoring* de apoio aos projetos, dando suporte ao trabalho do SEPA (*Software Engineering Process Authority*).

5.4.4. Macro-Atividade: Monitorar Qualidade da Iteração

O propósito desta macro-atividade é prover uma avaliação contínua na iteração sobre as atividades de qualidade realizadas, visando a melhoria do processo de qualidade de *software*. A Figura 5.5. apresenta o detalhamento dessa macro-atividade.

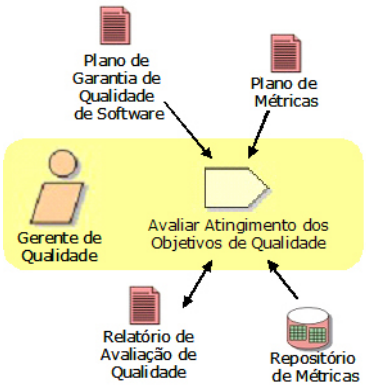

Figura 5.5 – Macro-atividade: Monitorar Qualidade da Iteração

Descrição

O Gerente de Qualidade analisa dados sobre as atividades de qualidade realizadas e sobre as métricas coletadas, para verificar se os objetivos de qualidade da iteração foram atingidos. A análise auxilia na adequação ou melhoria do processo de qualidade, possibilitando que ações preventivas possam ser empreendidas.

Agenda

As atividades de monitoração da qualidade são realizadas ao longo da iteração.

Recomendações

As atividades são recomendadas para quaisquer projetos e têm sua execução amparada pela Política de Qualidade definida e seguida pela organização.

Equipe

O Gerente de Qualidade é responsável pelas análises dos dados que visam melhorar o processo de *software*, avaliando o cumprimento dos objetivos e o desempenho das atividades de qualidade realizadas ao longo do processo.

Atividades

Esta macro-atividade é composta pela atividade Avaliar o atingimento dos objetivos de qualidade.

5.4.4.1. Avaliar o Atingimento dos Objetivos de Qualidade

O Quadro 5.8 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Avaliar o atingimento dos objetivos de qualidade</th></tr>
<tr><td colspan="2">Propósito: Avaliar a efetividade das ações de qualidade realizadas para o alcance dos objetivos de qualidade propostos para o projeto.</td></tr>
<tr><td colspan="2">Passos:
- Verificar se os objetivos de qualidade da iteração foram atingidos.
- Registrar informações para melhorar o processo de software e o processo de qualidade.</td></tr>
<tr><td>Artefatos de Entrada:
- Plano de Garantia de Qualidade de Software
- Repositório de métricas do projeto
- Plano de Métricas
- Relatório de Avaliação de Qualidade</td><td>Artefatos Resultantes:
- Relatório de Avaliação de Qualidade</td></tr>
<tr><td colspan="2">Freqüência: ao final de cada iteração.</td></tr>
<tr><td colspan="2">Papel: Gerente de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Monitorar Qualidade da Iteração</td></tr>
</table>

Quadro 5.8 – Atividade: Avaliar atingimento dos objetivos de qualidade

Verificar se os Objetivos de Qualidade da Iteração foram Atingidos

O Gerente de Qualidade verifica se os objetivos de qualidade estabelecidos para a iteração foram atingidos pelas ações de qualidade empreendidas na iteração. Um dos pontos de verificação são os resultados constantes no artefato Relatório de Avaliação de Qualidade, considerando a freqüência com que as avaliações de qualidade foram realizadas no projeto e quão aderentes foram as ações de qualidade em relação aos objetivos da fase na qual o projeto se encontra. Alguns exemplos de informações analisadas são:

- Atividades realizadas de qualidade versus atividades planejadas;
- Não conformidades mais críticas;
- Pontos do processo em que o projeto possui mais dificuldades;
- Identificação de ações preventivas.

A análise sobre esses pontos possibilita identificar tendências em relação à qualidade para o projeto como um todo, em termos de processo e de produto.

5.4.5. Macro-Atividade: Registrar Lições Aprendidas de Qualidade

O propósito desta macro-atividade é, ao final do projeto, registrar as lições aprendidas do processo de qualidade executado no projeto. A figura 5.6 abaixo mostra a macro-atividade:

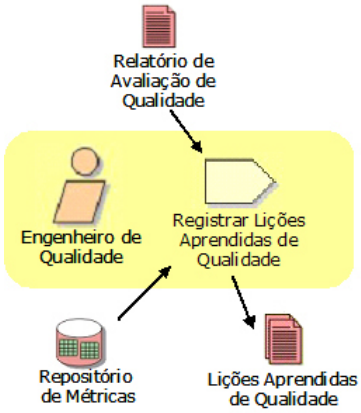

Figura 5.6 – Macro-atividade: Registrar Lições Aprendidas de Qualidade

Descrição

As lições aprendidas de qualidade, identificadas ao longo de cada iteração com as atividades de qualidade realizadas, podem:

- auxiliar o Engenheiro de Processo, na disciplina de Ambiente, a selecionar a instância do *framework* do UP que constituirá o processo mais adequado para projetos similares, atualizando o artefato Caso de Desenvolvimento;
- identificar tipos de não conformidades mais comuns e como elas podem ser prevenidas e corrigidas;
- possibilitar avaliar como determinados tipos de não conformidades podem ser resolvidos e em que espaço de tempo;
- ajudar a quantificar a equipe de qualidade adequada por tipo de projeto, dentre outras informações válidas.

Agenda

A atividade é realizada ao final do projeto, embora informações sobre lições aprendidas possam ser coletadas ao término das iterações.

Recomendações

A atividade é recomendada para quaisquer projetos.

Equipe

O Engenheiro de Qualidade é responsável por definir, a partir da experiência com as atividades de qualidade ao longo das diversas iterações e fases do projeto, as lições aprendidas que irão auxiliar projetos futuros.

Atividades

A única atividade desta macro-atividade é Registrar lições aprendidas de qualidade.

5.4.5.1. Registrar Lições Aprendidas de Qualidade

O Quadro 5.9 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Registrar Lições Aprendidas de Qualidade</th></tr>
<tr><td colspan="2">Propósito: Identificar e registrar as lições aprendidas de qualidade ao longo do projeto com as atividades de qualidade de software desenvolvidas ao longo do processo.</td></tr>
<tr><td colspan="2">Passos:
- Identificar e registrar as lições aprendidas de qualidade.</td></tr>
<tr><td>Artefatos de Entrada:
- Relatório de Avaliação de Qualidade.
- Repositório de Métricas</td><td>Artefatos Resultantes:
- Lições Aprendidas de Qualidade.</td></tr>
<tr><td colspan="2">Freqüência: no encerramento do projeto.</td></tr>
<tr><td colspan="2">Papel: Engenheiro de Qualidade</td></tr>
<tr><td colspan="2">Macro-Atividade:
Gerenciamento da Qualidade
Registrar Lições Aprendidas de Qualidade</td></tr>
</table>

Quadro 5.9 – Atividade: Registrar Lições Aprendidas de Qualidade

Identificar e Registrar as Lições Aprendidas de Qualidade

O Engenheiro de Qualidade é responsável por identificar e registrar as lições aprendidas de qualidade que poderiam melhorar processos para futuros produtos e serviços (CMMI, 2002a). As lições aprendidas de qualidade podem estar relacionadas:

- ao processo de qualidade de *software* executado no projeto;
- ao processo de desenvolvimento de *software* definido para o projeto (instância do *framework* do UP);
- às técnicas e métodos utilizados pelo projeto;
- à distribuição de avaliações de qualidade ao longo das fases e iterações.
- à melhoria do processo de qualidade de *software* e à melhoria do processo de desenvolvimento de *software*.

Engenheiro de Qualidade e Gerente de Qualidade devem tratar com os demais papéis ligados ao gerenciamento das equipes de desenvolvimento sobre as tendências de qualidade relativas a um projeto ou a vários projetos.

5.5. Artefatos

Os artefatos relacionados com a qualidade de *software* que o UP já propõe foram especificados no Cap. 4 deste trabalho. Esta seção apresenta com mais detalhes os artefatos utilizados anteriormente que sofreram algumas adaptações e o conjunto de artefatos propostos para dar suporte à disciplina de Gerenciamento da Qualidade. Os *templates* para os artefatos do UP são apresentadas nos Apêndices deste trabalho.

Plano de Garantia de Qualidade

O Plano de Garantia de Qualidade proposto pelo RUP (2003) foi revisado para estar em conformidade com o padrão IEEE 730 (2002). Além disso, o Gerente de Qualidade e não mais o Gerente do Projeto passa a ser o papel responsável por este artefato, apresentado no Apêndice B.

Relatório de Avaliação de Qualidade

O Relatório de Avaliação de Qualidade é um novo artefato proposto (Apêndice C), que é utilizado para registrar as informações relacionadas à avaliação de qualidade realizada pelo Engenheiro de Qualidade no projeto, ao longo das iterações.

Lições Aprendidas de Qualidade

As Lições Aprendidas de Qualidade são utilizadas para registrar as informações para melhoria dos processos, produtos e serviços, ao longo das iterações e de acordo com os objetivos das fases do Processo Unificado. Os artefatos que compõem o conjunto denominado Documentos Organizacionais de Qualidade são originados da ISO 9001 (2000) e estão vinculados à visão organizacional de qualidade.

5.6. A disciplina Gerenciamento da Qualidade ao longo das fases do UP

As atividades da disciplina de Gerenciamento da Qualidade (GQ) foram concebidas para serem aderentes às fases do Processo Unificado, visando dar suporte, por meio das ações de qualidade, ao atendimento do conjunto de objetivos de cada fase. É apresentado, nesta seção, o escopo das atividades da disciplina de GQ que são realizadas ao longo das iterações, em cada fase.

5.6.1. Fase de Concepção

Na fase de Concepção, o principal objetivo é compreender o que construir. Ela possui 5 objetivos específicos, derivados do objetivo principal. Na Tabela 5.1, é detalhado o

escopo das atividades da disciplina GQ, de acordo com os objetivos específicos da fase de Iniciação.

Tabela 5.1 – Gerenciamento da Qualidade na fase de Iniciação – adaptado de Kroll & Kruchten (2003)

Fase de Iniciação		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender o que construir (determinar o escopo do sistema, para quem o sistema servirá e qual é a melhor solução).	• se a aplicação a ser desenvolvida (*software*), no artefato Visão, foi definida em termos de: - benefícios e oportunidades; - problemas resolvidos pela aplicação; - usuários-alvo; - necessidades e características. • se há descrição de casos de uso e atores; • se há descrição dos requisitos não funcionais da aplicação; • se os protótipos de interface foram gerados, se previstos;	• Visão • Especificação de requisitos de *software* • Especificação suplementar de requisitos • Especificação de caso de uso
2. Identificar casos de uso críticos.	• se as funcionalidades-chave (casos de uso críticos) foram identificadas e priorizadas.	• Especificação de requisitos de *software*
3. Determinar pelo menos uma solução possível.	• se foi identificada pelo menos uma solução arquitetural; • se, para cada solução arquitetural proposta: - a estratégia de evolução do *software* foi estabelecida; - as tecnologias a serem utilizadas foram identificadas e os riscos associados a sua utilização; - decisões relativas à aquisição ou desenvolvimento interno foram consideradas, e os riscos associados a essas decisões foram especificados; - os componentes a serem utilizados (database, middleware) foram considerados em termos de aquisição, reuso, custos de aquisição e se os riscos associados a essas decisões foram especificados; • se o protótipo funcional (se existir) foi avaliado pela equipe de testes. • Se a estratégia de testes foi definida.	• Documento de arquitetura de *software* • Lista de riscos • Realizações de caso de uso • Plano de Testes (Estratégia de Testes) • Sumário de Avaliação de Testes
4. Estimar custos, cronograma e riscos associados ao projeto.	• se foram especificadas informações de: - estimativas; - recursos; - cronograma; - custo do projeto; - listas de riscos.	• Caso de Negócio • Lista de Riscos • Plano de Desenvolvimento de *Software*
5. Decidir que processo seguir e que ferramentas utilizar.	• se foram especificadas informações de: - processo definido para o projeto; - ferramentas a serem utilizadas; - *templates* e artefatos. • Se o ambiente (processo, ferramentas, *template*s) foi preparado para uso pelo projeto.	• Caso de Desenvolvimento • Plano de Desenvolvimento de *Software*

Durante esta fase, as atividades de gerenciamento da qualidade envolvem ainda a participação em revisões técnicas ou gerenciais, podendo contemplar a Revisão do Marco dos Objetivos do Ciclo de Vida ou verificar se há evidências de que foi realizada e atingiu os objetivos pretendidos. Verifica também se os artefatos do projeto encontram-se sob controle de versão e sob gestão de configuração.

5.6.2. Fase de Elaboração

Na fase de Elaboração, o principal objetivo é compreender como construir, definindo uma *baseline* da arquitetura estável e avaliando os riscos do projeto. O escopo das atividades da disciplina proposta, de acordo com os objetivos específicos da fase de Elaboração, são descritos na Tabela 5.2:

Tabela 5.2 – Gerenciamento da Qualidade na fase de Elaboração – adaptado de Kroll & Kruchten (2003)

Fase de Elaboração		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender requisitos detalhadamente.	• se os requisitos funcionais foram descritos e detalhados e representados em especificações de casos de uso; • se todos os casos de uso arquiteturalmente significativos foram detalhados; • se há evidências de que os casos de uso foram testados; • se o Glossário está sendo atualizado; • se mudanças nos requisitos são gerenciadas e rastreadas.	• Especificação de requisitos de *Software* • Especificação de caso de uso • Realização de caso de uso • Glossário • Registros de revisão
2. Projetar, implementar e validar a arquitetura, estabelecendo uma *baseline* de arquitetura.	• se o sistema foi definido em termos de blocos de construção, identificando: - subsistemas e componentes; - interfaces entre os componentes; - decisões e riscos de implementação dos componentes (construção, aquisição, ou reuso); - descrição da interação entre os componentes ao longo dos cenários. • se há evidências da realização dos casos de uso, com as devidas correspondências entre as classes de análise e de *design*; • se há evidências de que as classes foram empacotadas de forma apropriada. • se há evidências da descrição de modelos abrangendo concorrência, processos e *threads*. • Se há modelo de dados do sistema, e se foi identificada a estratégia de armazenamento e recuperação de dados. • se foram utilizados na solução os mecanismos arquiteturais padrões da organização adequados aos problemas. • se há evidências de que o protótipo foi implementado, contemplando cenários críticos. • se há evidências de que houve revisão do código implementado, e se houve teste de código, com implementação de classes de teste pelos desenvolvedores. • se há evidências de que o integrador combinou as partes adequadas (componentes e subsistemas) do sistema; • se há evidências de que foram realizados testes do protótipo (*build*) pela equipe de testes, segundo o Plano de testes.	• Documento de Arquitetura de *software* • Realizações de caso de uso • Especificações de requisitos de *software* • Especificação de caso de uso • Plano de testes • Relatório de testes • Modelo de dados • Modelos UML • Registros de revisão • Plano de integração • Sumário de avaliação de testes
3. Mitigar riscos e produzir estimativa acurada de custo e cronograma	• se os dados do plano de projeto e estimativas de custo foram atualizados.	• Plano de Desenvolvimento de *Software* • Lista de Riscos
4. Refinar Caso de	• se, no artefato Caso de Desenvolvimento, foram identificadas melhorias e adaptações no uso de processos e	• Caso de Desenvolvimento

Desenvolvimento e ambiente de desenvolvimento.	ferramentas; • se o ambiente de desenvolvimento foi ajustado para a iteração.	

Durante a fase de Elaboração, as atividades de gerenciamento da qualidade podem contemplar ainda a participação em revisões técnicas ou gerenciais. Também pode abranger a participação da Revisão do Marco da Arquitetura do Ciclo de Vida ou a verificação da existência de evidências de que ela foi realizada e atingiu os objetivos pretendidos. Também verifica:

- se o material de treinamento para usuários e mantenedores já começou a ser planejado;
- se houve necessidade de aquisição e instalação de novo hardware.

5.6.3. Fase de Construção

Na fase de Construção (Tabela 5.3), o principal objetivo é desenvolver uma versão operacional do sistema que possa ser implantada.

Tabela 5.3 – Gerenciamento de Qualidade na fase de Construção – adaptado de Kroll & Kruchten (2003)

Fase de Construção		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Minimizar custos de desenvolvimento e atingir algum grau de paralelismo.	• se há evidências nos modelos UML do projeto de que as responsabilidades do sistema foram divididas em subsistemas bem definidos; • se os artefatos do projeto encontram-se sobre controle de versão e sob gestão de configuração; • se há evidências de que é realizado o planejamento da integração, identificando que determinada versão do *software* é composta por determinados componentes ou subsistemas definidos para o sistema. • se os mecanismos arquiteturais planejados foram considerados entre as opções de projeto; • se há evidências de que os desenvolvedores testam o código (teste de unidade).	• Modelos UML do projeto • Realização de Caso de Uso • Plano de Gestão de Configuração • Mecanismos Arquiteturais • Sumário de Avaliação de Testes
2. Desenvolver um produto completo que esteja pronto para ser entregue ao cliente.	• se há evidências de que todos os casos de uso remanescentes foram projetados, implementados e testados; • se houve revisão do *design* dos Casos de Uso remanescentes; • se todos os cenários dos Casos de Uso foram projetados, implementados e testados. • se as mudanças relacionadas a banco de dados abrangem apenas criação de colunas, de visões e de índices. • são realizados testes de integração e testes sistêmicos; • se há evidências de que são realizados testes de usuários; • se há evidências de que são preparados os artefatos para implantação do *software*.	• Realização de caso de uso • Modelos UML • Especificação de Requisitos de *Software* • Casos de uso • Registro de Revisão • Plano de Testes • Casos de Testes • Sumário de Avaliação de Testes • Solicitações de mudanças de Bancos de Dados • *Build* • Instruções de Instalação • Manuais de Usuários • Material de Treinamento

São realizados: o projeto detalhado, a implementação e os testes do sistema como um todo, com base na arquitetura estabilizada durante a fase de Elaboração. Seus dois objetivos específicos visam garantir a integridade da arquitetura, minimizar custos por meio do desenvolvimento em paralelo, controlar o impacto das mudanças e realizar testes automatizados.

Durante a fase de Construção, as atividades de gerenciamento de qualidade podem contemplar ainda a participação em revisões técnicas (de código) ou gerenciais (de iteração). Também pode abranger ou a participação durante a Revisão do Marco da Capacidade Operacional Inicial do Ciclo de Vida ou a verificação, durante a avaliação de qualidade, se há evidências de que essa reunião foi realizada e atingiu os objetivos pretendidos. Também contemplam a verificação dos seguintes itens:

- o material de treinamento para usuários e mantenedores está sendo construído;
- o material de suporte para o usuário final começou a ser desenvolvido;
- se houve necessidade de aquisição e instalação de novo *hardware* e se esse processo foi concluído;
- se foi definida estrutura de empacotamento do *software* (*Bill of Materials*).

5.6.4. Fase de Transição

Na fase de Transição, o principal objetivo é garantir que o sistema implementou por completo as necessidades do cliente, gerando uma versão beta operacional do sistema. Na tabela 5.4 a seguir, é detalhado o escopo das atividades da disciplina GQ, de acordo com os objetivos da fase de Transição.

Tabela 5.4 – Gerenciamento de Qualidade na fase de Transição – adaptado de Kroll & Kruchten (2003)

Fase de Transição		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Realizar testes beta.	• se os usuários realizaram testes de avaliação do *software* em sua versão beta; • se solicitações de mudança do usuário foram implementadas antes de implantar a versão final do *software*; • se solicitações de mudança para correção de defeitos foram implementadas; • se foram realizados testes de regressão pelo desenvolvedor e pela equipe de testes; • se a documentação do *software* a ser utilizada na implantação foi desenvolvida; • se os testes de sistema planejados foram realizados; • se os testes de aceitação foram realizados; • se *patch releases* foram produzidos para corrigir defeitos significativos.	• Sumário de avaliação de testes • Relatórios de defeitos • Solicitações de mudanças • *Help On line* • Material de treinamento • Manual do usuário • Manual de instalação

2. Treinar usuários.	• se o material de treinamento foi concluído; • se a documentação do usuário foi concluída; • se os manuais operacionais foram preparados e concluídos; • se houve treinamento para os usuários.	• Registros de treinamento • Material de treinamento • Manual de instalação • Manuais do usuário
3. Preparar implantação e converter bancos de dados operacionais.	• se houve necessidade de migração de dados e se ela realmente aconteceu; • se houve necessidade de que o sistema anterior (se for o caso) e sistema novo executarem em paralelo, e se foram registradas análises relacionadas a isto; • se o hardware novo necessário foi adquirido e instalado.	• Documento de Arquitetura de *Software* • Plano de Implantação
4. Preparar empacotamento.	• Se foi definida a estrutura de empacotamento do *software* e se ela está completa.	• Empacotamento de *software* • Plano de Implantação
5. Avaliar as *baselines* de implantação.	• se foram realizados testes de aceitação do produto, considerando a completude das *baselines* de implantação.	• Sumário de Avaliação de Testes • Plano de Gerenciamento de Configuração
6. Registrar lições aprendidas.	• se a Revisão *Post Mortem* foi realizada; • se o artefato Caso de Desenvolvimento foi atualizado com base nas lições aprendidas do projeto.	• Registros de Revisão • Caso de Desenvolvimento

Nesta fase, as atividades de GQ contemplam ainda a participação na Revisão *Post Mortem*, e na Revisão do Marco de *Release* do Produto ou a verificação da existência de evidências de que essas revisões foram realizadas e atingiram os objetivos pretendidos.

5.7. Gerenciamento da Qualidade e as demais disciplinas do UP

A disciplina Gerenciamento de Qualidade (GQ) está relacionada a todas as demais disciplinas do UP, conforme a seguir:

- As atividades de GQ da **Modelagem de Negócios** buscam assegurar a qualidade dos artefatos gerados, como os Modelos de Casos de Uso do Negócio, Modelos de Análise de Negócio e Regras de Negócio. Também verifica a realização das revisões técnicas sobre artefatos: modelos de casos de uso de negócios e modelos de análise de negócio.
- As atividades de GQ de **Requisitos** visam verificar se os requisitos são claramente definidos, priorizados e implementados pelo processo de desenvolvimento de *software*, assegurando que o produto de *software* atende os requisitos especificados. Também são avaliados os artefatos da disciplina, como Visão, Modelos de Casos de Uso, Especificação de Requisitos de *Software* e Plano de Gerenciamento de Requisitos. Também verifica a realização das revisões técnicas sobre os requisitos.
- As atividades de GQ da **Análise e Projeto** visam garantir rastreabilidade entre os requisitos, os casos de uso, e os modelos UML de realização desses casos de uso, traduzindo os requisitos em componentes do sistema a ser construído. Dentre os artefatos avaliados estão o Documento de Arquitetura de *Software*, Realizações de

Casos de Uso, Modelos UML de Análise, de projeto (*design*) e de implantação, além do protótipo de interface do usuário e modelos de dados. Também verifica a realização das revisões técnicas sobre a arquitetura e sobre o projeto (*design*) de *software*.

- As atividades de GQ de **Implementação** avaliam se os elementos de implementação correspondem aos modelos de projeto (*design*) resultantes das atividades da disciplina Análise e Projeto (*Design*) e que eles são integrados corretamente constituindo a versão operacional do produto de *software*. Dentre os artefatos avaliados estão: Plano de Integração de *Build* e Modelos UML de Implementação. Também verifica a realização da revisão técnica do código.
- As atividades de GQ de **Testes** são auditadas para verificar se as ações planejadas correspondem às ações implementadas e se os testes realizados podem ser repetidos sob as mesmas condições.
- As atividades de GQ da **Implantação** verificam se o processo de implantação de *software* foi devidamente seguido, avaliando as ações do Plano de Implantação e se outros artefatos associados foram desenvolvidos, como o Material de Treinamento e o Material de Suporte.
- As atividades de GQ da **Gestão de Configuração e Mudanças** verificam se os projetos estão cumprindo as políticas de gestão de configuração e se elas são adequadas às necessidades dos projetos. Avalia se as auditorias de gestão de configuração estão sendo realizadas e se o fluxo de solicitação de mudanças é adequado às necessidades dos projetos e da organização.
- As atividades de GQ do **Gerenciamento de Projetos** envolvem o planejamento de atividades, realização de estimativas, acompanhamento das ações, gerenciamento de riscos, e a realização das revisões gerenciais do projeto.
- As atividades de GQ do **Ambiente** verificam a adequação do processo definido para o projeto, as necessidades de adaptação que precisam ser realizadas, e reporta as questões identificadas, visando a melhoria do processo.

5.8. Conclusão

A disciplina Gerenciamento da Qualidade foi proposta visando melhorar a abordagem de qualidade do UP, especialmente no que diz respeito à garantia de qualidade do processo. A disciplina pode ser considerada aderente aos seguintes padrões e modelos de qualidade, conforme descrito a seguir.

- ISO/IEC 12207: atende aos seguintes processos/atividades de qualidade da norma:
  - *Garantia de Qualidade: Implementação do Processo* - um processo contemplando a realização de auditorias é especificado, exigindo que o papel que as executará seja independente do projeto avaliado.
  - *Garantia de Qualidade: Garantia do Processo* - provê um processo com atividades visando garantir a qualidade do processo.
  - *Garantia de Qualidade: Garantia do Produto* – o processo proposto verifica a realização dos testes de *software* e avaliação da qualidade dos artefatos gerados.
  - *Garantia de Qualidade: Sistema de Gestão da Qualidade* - provido parcialmente pela proposta, ao estabelecer políticas, procedimentos, padrões, e tratamento de não conformidades.
  - *Auditoria: Implementação do processo* - provido pela proposta, utilizando a técnica de auditoria nas avaliações de garantia de qualidade.
  - *Auditoria: Auditoria* – provida com a realização das avaliações de garantia de qualidade, utilizando a técnica de auditoria e obedecendo ao critério de independência em relação aos projetos avaliados.
  - *Gerenciamento: Gerenciamento de Qualidade* - provê uma infra-estrutura para o estabelecimento de um processo organizacional para gerenciamento da qualidade.
- Em relação ao modelo CMMI (2002a), atende 100% dos objetivos, implementando 100% das práticas requeridas em relação à área de processo PPQA (*Process and Product Quality Assurance*) ou Garantia de Qualidade do Processo e do Produto.

Além disso, a disciplina proposta supre as deficiências encontradas no UP e disponíveis na literatura da área, conforme Capítulo 4 deste trabalho:

- Em relação às deficiências identificadas por Manzoni e Price (2003), a disciplina de GQ provê tratamento sistemático de não conformidades através de revisões, e especificando como são definidas as ações corretivas que devem ser realizadas e os responsáveis pela resolução. A proposta também especifica como as não conformidades são identificadas, documentadas e acompanhadas até a conclusão, assim como são tratados os desvios.
- Em relação às deficiências identificadas por Smith (2003), elas especificam os tipos de revisão providos pelo UP, acrescidos dos tipos de Revisões de *Software* utilizadas, conforme o padrão IEEE 1028 (1997). Também abordam o tratamento de desvios, com o processo de identificação e a aprovação pelos *stakeholders*.

Capítulo 4

QUALIDADE DE *SOFTWARE* NO PROCESSO UNIFICADO

Este capítulo aborda o Processo Unificado e suas características com relação à qualidade de software, com foco na qualidade do processo.

O UP (*Unified Process*), também chamado de Processo Unificado, foi desenvolvido pela *Rational Software* (empresa adquirida pela IBM) e apresenta-se como um processo de desenvolvimento de *software* apoiado em modelos e nas denominadas melhores práticas de desenvolvimento de *software* do mercado.

4.1 Características do Processo Unificado

O Processo Unificado é uma abordagem de desenvolvimento de *software* iterativa, centrada em arquitetura e dirigida a casos de uso (JACOBSON *et al.*, 1998).

O UP é orientado a riscos, que são gerenciados de forma integrada ao processo de desenvolvimento; as iterações do processo são planejadas com base nos riscos de alta prioridade. O desenvolvimento é orientado a *casos de uso*, que expressam uma funcionalidade do sistema em seu mais alto grau de abstração. Isso significa que os casos de uso são definidos para expressar os requisitos funcionais dos sistemas e modelar o negócio e, a partir de então, são utilizados como base para todo o processo de desenvolvimento. As atividades de projeto do sistema são centradas em arquitetura, que é utilizada para conceituar, construir, gerenciar e desenvolver o sistema, por meio de visões ou modelos múltiplos coordenados, baseados na UML (*Unified Modeling Language*) (KRUCHTEN, 1999; KROLL & KRUCHTEN, 2003).

O Processo Unificado procura garantir a integração entre as equipes de desenvolvimento, facilitando o acesso a bases de conhecimento comuns sobre o processo de desenvolvimento de *software* na organização. Essa integração é particularmente facilitada pela utilização da UML, que atualmente é mantida pela OMG (*Object Management Group*). As atividades do UP são baseadas na criação e manutenção de modelos UML, que representam semanticamente o *software* que está sendo desenvolvido e as atividades a ele associadas (BOOCH, 1995; JACOBSON *et al.*, 1997; BOOCH *et al*., 1998).

Além de ser uma abordagem de desenvolvimento de *software* iterativa, o UP é um processo de engenharia de *software* que estabelece o que deve ser feito, como deve ser feito, quando e por quem, abrangendo o ciclo de vida de *software* de um projeto. Ele provê um *framework* de processos, utilizados por uma organização em seus projetos de desenvolvimento por meio da seleção dos processos adequados à realidade dos projetos (KROLL & KRUCHTEN, 2003).

O Processo Unificado é constituído por elementos de processo classificados como *elementos essenciais*, quando fazem parte do núcleo do processo, e como *elementos de suporte*, quando provêem informações adicionais aos elementos essenciais, conforme Tabela 4.1 (RUP, 2003).

Tabela 4.1 - Elementos de processo do UP

Elemento	Tipo	Conceito
Papel	Essencial	Definição do comportamento e das responsabilidades de um indivíduo ou grupo de indivíduos trabalhando juntos em equipe.
Atividade	Essencial	Constitui uma unidade de trabalho, executada por um papel com um propósito definido, geralmente criando ou atualizando um artefato. Pode ser dividida em passos.
Artefato	Essencial	Um pedaço de informação que é produzido, modificado ou criado por um processo. São usados como entrada na execução de atividades e como produto das atividades executadas.
Conceito	Suporte	Introduz definições e princípios chave.
Guia	Suporte	Provê regras, recomendações e heurísticas que suportam atividades, passos e artefatos.
Mentor de Ferramenta	Essencial	Orientação de uso das ferramentas de apoio ao processo.
Disciplina (*workflow*)	Essencial	Fluxo de atividades de alto-nível que produz um resultado de valor observável.
Macro-atividade (*workflow detail*)	Essencial	Agrupamento de atividades que são realizadas em conjunto.
Fase	Essencial	Período entre marcos do tempo com objetivos a serem atingidos, em termos de papéis, artefatos e macro-atividades.
Template	Suporte	Associados aos artefatos, provêem informações sobre como estes devem ser produzidos.
Report	Suporte	Extraem informações sobre um ou mais artefatos a partir de uma ferramenta.
Checkpoint	Suporte	Critérios de finalização de artefatos.
Whitepaper, *Roadmap*, e exemplo.	Suporte	Provêem informações sobre o UP em vários níveis de detalhes.

O UP compõe-se de duas dimensões ou estruturas: uma estática e outra dinâmica. A *dimensão estática* representa a estrutura estática do processo, descrevendo como os elementos do processo são agrupados logicamente em termos de disciplinas ou fluxos de trabalho (*workflow*). A *dimensão dinâmica* é representada pelo tempo e expressa o processo por meio de ciclos, decompostos em Fases. As fases são divididas em iterações com marcos de conclusão. São elas: *Iniciação* (ou *Concepção*), *Elaboração*, *Construção*, e *Transição*

(MOHAGHEGI, 2002). As dimensões estática e dinâmica podem ser visualizadas na Figura 4.1.

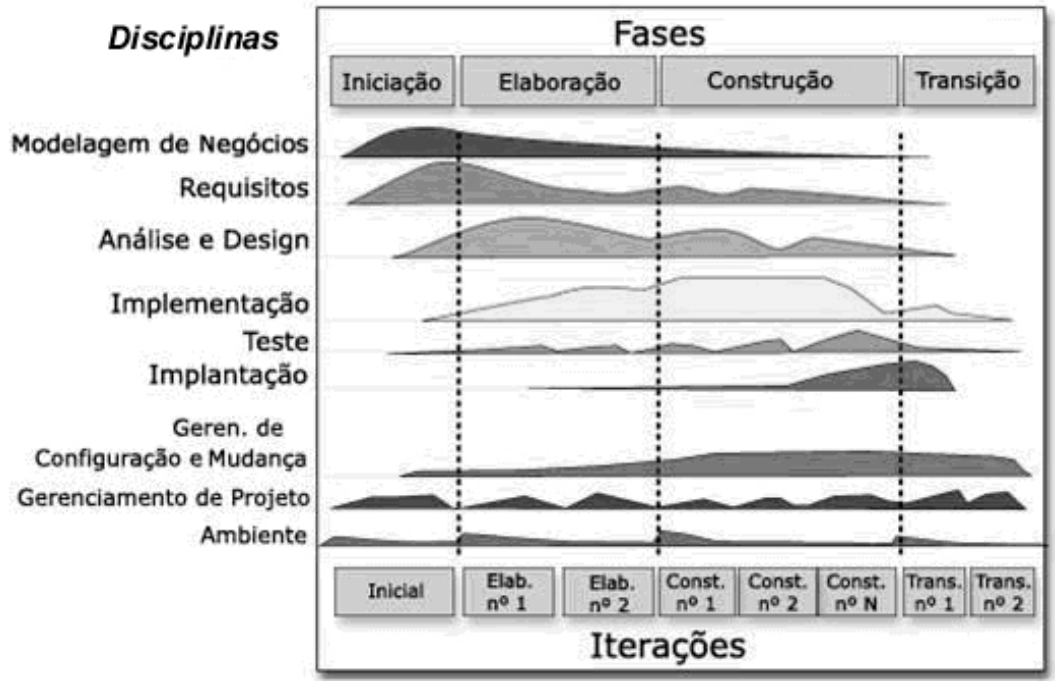

Figura 4.1 - Dimensões do Processo Unificado (RUP, 2003)

Cada Fase do UP possui objetivos de negócio bem definidos a serem atingidos em cada marco, por meio da produção ou evolução de artefatos técnicos. As descrições das fases são sumarizadas na Tabela 4.2:

Tabela 4.2 - Fases do Processo Unificado (RUP, 2003)

Fase	Descrição
Iniciação	Estabelece uma boa compreensão do sistema a ser construído, por meio de requisitos de alto nível e do estabelecimento do escopo do projeto. Obtém um acordo com os envolvidos (*stakeholders*), para prosseguir ou não com o projeto. Também é preparado o ambiente para o projeto, estabelecendo o artefato Caso de Desenvolvimento.
Elaboração	Objetiva a compreensão de como construir o sistema, analisando o domínio do problema, estabelecendo uma arquitetura básica, desenvolvendo um plano de projeto e eliminando os maiores riscos do projeto.
Construção	Os componentes da solução são desenvolvidos e integrados em um produto. Custos, cronograma e qualidade são otimizados pelo gerenciamento dos recursos e o controle das operações. Os componentes são desenvolvidos e testados baseados em critérios de avaliação definidos. *Releases* do produto também são avaliadas.
Transição	O principal objetivo desta fase é construir a versão final do produto e disponibilizá-la para os usuários finais. São desenvolvidos os materiais de suporte à instalação e ao uso pelo usuário final.

Os marcos de conclusão da fase são pontos onde decisões críticas devem ser tomadas e onde deve ser verificado se os objetivos estabelecidos para a fase foram atingidos. Segundo Kruchten (2001), "*o que importa nas fases não é tanto o que você faz durante as fases, ou quanto tempo elas demoram, mas o que você necessita atingir*" ou ainda "*a fase é 'julgada' pelo marco que a conclui*". As fases são orientadas à mitigação de riscos e não são opcionais. Para Kruchten (2001), o esforço de uma fase pode ser reduzido se os objetivos relacionados àquela fase já tiverem sido atingidos. Isso pode ser verificado quando um projeto confronta elementos de que já dispõe (como, por exemplo, requisitos compreendidos, arquitetura estável etc), com os critérios de finalização da fase (fase de Elaboração, por exemplo). Nesse caso, o projeto poderia concluir que possui os elementos necessários para considerar que a fase foi concluída. Ocorrências como essa seriam aplicáveis a projetos de manutenção de sistemas existentes (KRUCHTEN, 2001).

Fases são divididas em iterações. Uma iteração é uma seqüência de atividades distintas, passando por várias disciplinas, que constitui um ciclo completo de desenvolvimento e do qual resulta um produto de *software* executável (versão interna ou externa). Essa versão vai sendo incrementalmente acrescida, iteração a iteração, até constituir o produto de *software* final do projeto (KROLL & KRUCHTEN, 2003; RUP, 2003). O foco de uma iteração e sua ênfase sobre várias atividades muda através do ciclo de vida. Dependendo do projeto, a duração de uma iteração pode variar de 2 até 6 semanas, sendo afetada por fatores como o tamanho da organização, a experiência da equipe, e o tamanho e a complexidade do projeto (RUP, 2003).

Visando assegurar o desenvolvimento de produtos de *software* de alta qualidade, satisfazendo as necessidades dos usuários finais, e dentro de cronogramas e orçamentos previsíveis, o UP utiliza-se das seis "melhores práticas" em desenvolvimento de *software*. Descritas por Kroll & Kruchten (2003), as chamadas melhores práticas possibilitariam a adaptação do processo para uma grande variedade de projetos e organizações. São elas (RUP, 2003):

- Desenvolver *software* iterativamente – os sistemas estão cada vez maiores e mais complexos, o que não justifica realizar a definição de todo o sistema, a modelagem e o desenvolvimento seqüencialmente. A abordagem iterativa possibilita uma compreensão do que deve ser construído por meio de sucessivos refinamentos, atacando os riscos do projeto. O projeto é dividido em iterações, onde, em cada iteração, são atacados os riscos mais graves para o projeto, e são produzidas versões executáveis dos produtos, com a equipe focada na obtenção de resultados.

- Gerenciar requisitos – trata-se da elicitação, da descrição, do registro e do acompanhamento de requisitos e restrições sobre requisitos. A abordagem de casos de uso e o uso de *cenários* auxiliam na identificação de requisitos funcionais, facilitando a captura de requisitos e a rastreabilidade dos mesmos ao longo do processo de desenvolvimento. Além disso, assegura que a arquitetura, o projeto (*design*), a implementação e os testes sejam realizados com base nos casos de uso definidos.
- Usar arquitetura baseada em componentes – o projeto foca no estabelecimento de uma arquitetura de *software* executável e robusta, suportando o desenvolvimento de *software* baseado em componentes. A abordagem do UP provê a definição de arquitetura utilizando componentes novos e reutilizando componentes existentes.
- Modelar *software* visualmente – o processo mostra como modelar *software* visualmente por meio da UML, como uma forma de capturar a estrutura e o comportamento dos componentes da arquitetura. As abstrações visuais auxiliam a comunicação de diferentes aspectos do *software*.
- Verificar a qualidade de *software* – O UP relata que a qualidade deve ser verificada com base em critérios como funcionalidade, confiabilidade, e performance. A qualidade é tratada com ênfase sobre o produto de *software* executável, possibilitando que a organização planeje, projete, implemente, execute e avalie vários tipos de testes sobre o executável.
- Controlar mudanças no *software* – o sucesso no esforço de desenvolver *software* está fortemente ligado à capacidade da organização em gerenciar e controlar as modificações que acontecem ao longo do processo de desenvolvimento. O UP trata do controle, do rastreamento e da monitoração de mudanças ocorridas ao longo do processo de desenvolvimento, facilitando a integração entre os membros das equipes de desenvolvimento ao longo do processo iterativo e mantendo a integridade dos produtos de *software*.

Os objetivos de cada *workflow*, ou disciplina, no UP são mostrados na Tabela 4.3 a seguir:

Tabela 4.3 - Disciplinas do Processo Unificado (RUP, 2003).

Disciplinas	Objetivos
Modelagem de Negócios	Compreende o entendimento da estrutura e da dinâmica da organização na qual o sistema será implementado.
Requisitos	A disciplina captura e gerencia os requisitos, além de projetar uma interface focada nas necessidades e nos objetivos do usuário final.
Análise e Projeto	Traduz os requisitos em uma especificação que descreve como implementar o sistema.
Implementação	Cria, desenvolve e integra componentes e subsistemas em *software* executável.
Testes	Avalia a qualidade do produto.
Implantação	Direciona o produto de *software* finalizado para seus usuários.
Gestão de Configuração e Mudanças	Rastreia e mantém a integridade dos ativos do projeto em desenvolvimento.
Gerenciamento de Projetos	Planeja um processo iterativo, decidindo a duração e o conteúdo de cada iteração.
Ambiente	Dá suporte à organização de desenvolvimento com processos configurados e ferramentas.

4.2. Visão de Qualidade de *Software* do Processo Unificado

No que diz respeito à qualidade de *software*, é analisada a visão apresentada pelo UP na versão 2003.06.03 (RUP, 2003), como se segue: conceitos de qualidade de *software*, qualidade de produto de *software*, qualidade de processo de *software* – tratando da definição do processo, garantia de qualidade do processo e melhoria do processo – e, finalmente, a visão sobre o gerenciamento da qualidade no Processo Unificado.

4.2.1. Conceitos de qualidade de *software* no Processo Unificado

O UP utiliza alguns critérios que devem ser usados para medir a qualidade, como o progresso, a confiabilidade e a performance, dentre outros. Para o UP, a qualidade pode ser resumida como sendo o critério para aceitabilidade, podendo ser negociada com base em fatores tais como: risco, oportunidades de mercado, requisitos, restrições de cronograma, restrições de pessoal, e custos (RUP, 2003).

Segundo o RUP (2003), qualidade do produto é construir o produto certo, enquanto que qualidade do processo é construir o produto corretamente. Estes conceitos assim definidos estão aderentes aos conceitos sobre Validação e Verificação (ISO, 2002; SWEBOK, 2004). Entretanto, o Processo Unificado salienta a necessidade de definir a qualidade de *software* como algo que possa ser mensurado e alcançado. Em linhas gerais, a qualidade do processo no UP seria implementada por meio de revisões, ao longo do ciclo de vida do projeto, ligadas à disciplina de Gerenciamento de Projeto.

O RUP (2003) afirma que um erro comum das organizações é a criação de grupos denominados de “Garantia de Qualidade”, “Testes”, “Controle de Qualidade” ou “Engenharia de Qualidade”, que são responsáveis pela qualidade. Schulmeyer & McManus (1999) ponderam que o erro das organizações não é a criação de um grupo em si responsável pela qualidade, mas a atribuição, ao grupo criado, com a responsabilidade *única* pela qualidade, ao invés de inserir na organização a cultura da qualidade como responsabilidade de todos. Embora considere a qualidade de *software* como responsabilidade de todos os envolvidos no processo, há carências no Processo Unificado quanto à preocupação com o estabelecimento de uma cultura de qualidade na organização.

4.2.2. Qualidade de Produto de *Software*

A ênfase da qualidade de *software* no UP está voltada para o produto de *software* executável. A qualidade de *software* constitui algo verificável por meio de Testes realizados sobre o produto de *software*, ao longo do ciclo de vida do projeto. O UP apresenta uma disciplina de Testes, que é estruturada para realizar o planejamento, o projeto, a execução e a avaliação dos testes sobre o produto de *software* executável em seus diversos estágios (*builds*, *releases*, produto). Como a abordagem é iterativa e os testes são realizados em cada fase, a qualidade do produto é verificada continuamente, sendo possível identificar defeitos mais cedo (KROLL & KRUCHTEN, 2003).

O modelo FURPS+ (GRADY, 1992), aplicado no UP para a caracterização de requisitos, é utilizado também para avaliar a qualidade do produto. FURPS+ significam categorias em que os requisitos de *software* são classificados: funcionalidade (*Functionality*), usabilidade (*Usability*), confiabilidade (*Reliability*), desempenho (*Performance*) e suportabilidade (*Supportability*), além de outros critérios simbolizados pelo sinal +, como restrições de projeto (*design*) de *software*, requisitos de implementação e de interface e requisitos físicos.

O UP adotou para os testes de *software* o conceito de “Qualidade Boa o Suficiente” (GEQ – *Good Enough Quality*), proposto por Bach (1997). De acordo com esse conceito, todo *software* apresenta erros e não é possível exaurir os testes sobre um *software* em função do tempo e do custo para realizá-los. Assim, os testes que serão realizados devem ser determinados de forma a prever os tipos de erros que o *software* venha a apresentar e eliminar a ocorrência de erros que são mais críticos para a aplicação.

O UP faz uma distinção entre os dois tipos de produtos de *software*: o *software* executável, versão final disponibilizada ao cliente, e os produtos que apóiam o processo de desenvolvimento, e que dele resultam como os planos, as especificações e outros artefatos (denominados também de *byproducts*), que estão associados ao processo (KROLL & KRUCHTEN, 2003).

4.2.3 Qualidade de processo de *software*

A qualidade de processo de *software* abrange a definição do processo, a garantia de qualidade do processo e a melhoria do processo. Com relação à definição e melhoria do processo, estes aspectos são tratados com base no produto RUP. Como o UP consiste de um *framework* de processos, incluindo ainda um conjunto de práticas de engenharia de *software*, o processo de *software* seguido pelo projeto é um subconjunto dos processos e práticas disponíveis no *framework*. Para estabelecer o processo a ser utilizado no projeto, o produto RUP é adaptado pelo engenheiro de processos, pelo gerente do projeto ou até mesmo por um mentor, que é alguém com experiência em implementações do Processo Unificado (KROLL & KRUCHTEN, 2003).

A definição do processo a ser utilizado no projeto é feita na instanciação da disciplina Ambiente, que é utilizada para configurar o processo e o conjunto de ferramentas adequado para dar suporte às necessidades dos projetos. O RUP produto provê três níveis de adaptação de processos: o nível organizacional (o próprio *framework*), o nível de projeto, e o processo definido em termos do domínio de aplicação.

O SEPA (*Software Engineering Process Authority*) é identificado como sendo o responsável por auxiliar a organização na identificação dos processos dos projetos, avaliando continuamente o processo para melhorá-lo. Não são especificadas, entretanto, como seriam realizadas as avaliações de melhoria do processo, nem se utilizariam técnicas baseadas em *assessment*, por exemplo.

Já em relação à garantia de qualidade, apesar de Kroll & Kruchten (2003) afirmarem que a qualidade do processo no UP é implementada por meio de revisões, de acordo com Reitzig (2003), as revisões do UP focam o produto de *software* e não o processo.

Royce (1998) afirma que as revisões no RUP possuem um importante papel para o desenvolvimento técnico à medida que profissionais com pouca experiência têm seu trabalho revisado por profissionais seniores na condução de um processo de revisão.

Segundo o RUP (2003), as revisões constituiriam as atividades de qualidade que, categorizadas segundo o padrão IEEE 1028 (IEEE 1028, 1997), são apenas de dois tipos: *revisões técnicas* e *revisões gerenciais*. As revisões técnicas, realizadas sobre os artefatos (como requisitos, modelos de casos de uso e código), são apresentadas na Tabela 4.4.

As revisões técnicas especificam um conjunto mínimo de artefatos do UP que deveriam ser submetidos a uma revisão formal: modelos de casos de uso de negócio, modelos de análise de negócio, requisitos, arquitetura, projeto (*design*) e código.

Tabela 4.4 – Revisões Técnicas no Processo Unificado

Revisão	Objeto da Revisão	Disciplina: *macro-atividades*	Executor	Ocorrência
Revisão do modelo de caso de uso de negócio	Verifica formalmente os resultados da modelagem dos casos de uso de negócio, de acordo com a visão dos *stakeholders* sobre o negócio.	**Modelagem de Negócios**: *refinar definições do processo de negócios.*	Revisor Técnico: deve conhecer o negócio e a tecnologia na qual o negócio será automatizado.	A cada nova versão do Modelo de casos de uso de negócio.
Revisão do modelo de análise de negócio	Verifica formalmente os resultados da modelagem da análise de negócio, de acordo com a visão dos *stakeholders* sobre o negócio.	**Modelagem de Negócios**: *desenvolver um modelo de domínio; refinar papéis e responsabilidades.*	Revisor Técnico: deve conhecer o negócio e a tecnologia na qual o negócio será automatizado.	Quando houver necessidade de checar a visão dos *stakeholders* sobre o negócio.
Revisão dos requisitos	Verifica formalmente os requisitos, conforme a visão do cliente sobre o sistema. Revisa as solicitações de mudança com impacto nos requisitos, o modelo de casos de uso, e as especificações de caso de uso.	**Requisitos**: *gerenciar mudanças nos requisitos*.	Revisor Técnico (revisor de requisitos)	Pelo menos uma vez por iteração, em cada fase, quando há refinamento dos casos de uso.
Revisão da arquitetura	As revisões de arquitetura possuem vários objetivos: identificar e mitigar riscos em cronogramas e custos; identificar falhas de projeto arquitetural; detectar erros entre a arquitetura e os requisitos; avaliar aspectos de arquitetura; e identificar oportunidades de reuso.	**Análise e Projeto**: *refinar arquitetura.*	Revisor Técnico (revisor de arquitetura)	Fim da fase de Iniciação, fim de fase da Elaboração, durante as fases de Construção e Transição, e no final desta última.

Revisão	Objeto da Revisão	Disciplina: *macro-atividades*	Executor	Ocorrência
Revisão do projeto (*design*)	Verifica adequação do projeto (*design*) aos requisitos, assegurando que o modelo de projeto seja consistente com as diretrizes gerais e que estas atinjam seus objetivos.	**Análise e Projeto**: *projetar o banco de dados; analisar comportamento; projetar componentes*.	Revisor Técnico	Uma revisão por iteração, nas fases de Elaboração e Construção, e nas demais fases quando o modelo for refinado.
Revisão do código	Verificar a implementação. É o único tipo de revisão no qual o RUP sugere o uso das técnicas de revisão, como inspeção e *walkthrough*.	**Implementação**: *implementar componentes*.	Revisor Técnico	A cada iteração.

O segundo tipo de revisão do UP, Revisão Gerencial, pode ser conduzido tanto pelo Gerente de Projetos, para avaliar o *status* das atividades de projeto pelo menos uma vez a cada iteração, como pelo PRA (*Project Review Authority*), para revisar a conformidade do projeto com obrigações contratuais e com padrões organizacionais (RUP, 2003). Informações sobre as Revisões Gerenciais são especificadas na Tabela 4.5 abaixo:

Tabela 4.5 – Revisões Gerenciais no Processo Unificado

Revisão	Objeto da Revisão	Disciplina: *macro-atividades*	Executor	Ocorrência
Revisão de aprovação do projeto	Determinar, do ponto de vista do negócio, se o projeto vale o investimento, avaliando riscos e custos.	**Gerenciamento de Projetos**: *conceber novo projeto.*	Revisor Gerencial: deve ter experiência no negócio e no domínio da aplicação. Envolve gerência *sênior*.	Uma vez por projeto
Revisão do planejamento do projeto	Aprovar ou revisar e aprovar mudanças no Plano de Desenvolvimento de *Software*.	**Gerenciamento de Projetos**: *planejar projeto*.	Revisor Gerencial: deve ser experiente em estimativas de projetos nos domínios técnicos e negocial para julgar as assunções feitas pelo gerente de projetos.	Uma vez no início do projeto e sempre que o Plano for atualizado
Revisão do plano da iteração	Aprovar o Plano de Trabalho proposto para a iteração.	**Gerenciamento de Projetos**: *planejar para a próxima iteração.*	Revisor Gerencial. Inclui a participação do cliente e de outros *stakeholders*. O Revisor Gerencial deve ser hábil em estimativas, planejamento e gerência de riscos, além de ter uma boa compreensão sobre o domínio da aplicação.	Uma vez por iteração. E sugerido o uso de *Walkthrough* ou revisão sobre o artefato internamente ao projeto antes de submetê-lo à Revisão pelos *stakeholders*.
Revisão dos critérios de avaliação da iteração	Aprovar critérios que serão utilizados para determinar se o trabalho concluído na iteração atingiu os objetivos propostos.	**Gerenciamento de Projetos:** *gerenciar iteração.*	Revisor Gerencial; requer experiência no domínio de aplicação e com a percepção do que está sendo considerado ou relegado durante a iteração.	A cada iteração, em cada fase.
Revisão de aceitação da iteração	Aceitar formalmente o trabalho completado na iteração.	**Gerenciamento de Projetos**: *gerenciar iteração.*	Revisor Gerencial, com a equipe do projeto e o cliente.	Uma vez por iteração.

Revisão	Objeto da Revisão	Disciplina: *macro-atividades*	Executor	Ocorrência
Revisão de marco de ciclo de vida	Revisar o status do projeto no final da fase, indicando o prosseguimento do projeto para a próxima fase.	**Gerenciamento de Projetos**: *fechar fase*.	Revisor Gerencial.	Uma vez, ao final de cada fase.
Revisão de aceitação do projeto	Revisão e aceitação formal pelo cliente dos produtos de trabalho do projeto.	**Gerenciamento de Projetos**: *fechar projeto*.	Revisor Gerencial, com a equipe do projeto, o cliente, e o PRA.	Ao final do projeto, quando for requerida uma aceitação formal.
Revisão do projeto pelo PRA	Revisar o progresso feito no projeto com o PRA, avaliando riscos e problemas do projeto, através do artefato Avaliação de Status.	**Gerenciamento de Projetos**: *monitorar e controlar projeto*.	Revisor Gerencial: necessita conhecer bem o gerenciamento de projetos, compreender largamente políticas e práticas da organização, além de ser capaz de realizar julgamentos sobre o desempenho financeiro dos projetos em relação a obrigações contratuais.	Em cada iteração em cada fase.

Não são previstas no UP revisões que enfoquem a garantia de qualidade, avaliando independentemente o processo de *software* seguido pelo projeto.

Apesar de considerar a importância da qualidade do processo e das atividades de revisão, Kroll & Kruchten (2003) dizem que, para o RUP, "*at the end of the day, what really counts is how good your code is, not how good your byproducts of software development are*". A afirmativa contrasta com a palavra dos *experts* em qualidade, cuja experiência demonstra que a qualidade deve ser vista como algo global, onde todos os envolvidos no processo são co-responsáveis em relação a ela; estes contribuem para a existência da qualidade em qualquer ação realizada, seja ela concretizada por meio do produto de *software*, dos artefatos que dão suporte ao processo, ou das ações desenvolvidas com qualidade ao longo do processo (SCHULMEYER & MCMANUS, 1999; UNHELKAR, 2003).

Os principais artefatos relacionados à qualidade de *software* no RUP (2003) são:

- Plano de Garantia de Qualidade: elaborado por projeto e estabelecido pelo gerente do projeto, para registrar as ações de qualidade previstas e realizadas, especificando como assegurar a qualidade do produto, dos artefatos e do processo.
- Caso de Desenvolvimento: elaborado por projeto, contém a definição do processo a ser seguido pelo projeto, a partir das adaptações do UP.

- Registro de Revisão: captura os resultados da atividade de revisão no qual um ou mais artefatos dos projetos são revisados. Para o RUP (2003), “é mais importante realizar a revisão do que documentá-la”, sugerindo que um *e-mail* sirva como registro da revisão.

O RUP (2003) define *gerenciamento da qualidade* como a implementação, medição e avaliação da qualidade do processo e do produto, afirmando que consiste em atividades de análise e avaliação de artefatos em cada uma das disciplinas, com o propósito de identificar medidas e indicadores válidos para qualidade. O gerente de projeto é o papel responsável pelo gerenciamento da qualidade.

A disciplina de Testes tem o papel de avaliar a qualidade do produto e a disciplina de Gerência de Projetos apresenta uma visão geral dos esforços para gerenciar qualidade, identificando revisões e auditorias requeridas para avaliar a aderência, a implementação e o progresso do processo de desenvolvimento. Isto contrasta com a afirmação de Reitzig (2003) de que as Revisões no UP são focadas no produto e não no processo. O Processo Unificado também não indica como seriam realizadas as auditorias de processo.

No UP, o gerenciamento de qualidade é realizado apenas através das revisões dos artefatos em suas disciplinas. Entretanto, o gerenciamento da qualidade envolve também questões relevantes, como a definição de processos e seus requisitos, métricas de processos, entradas, saídas e canais de *feedback*, que atuem para melhorar a qualidade do produto definido, e especificar o processo para atingir a qualidade desejada para o produto, garantindo uma melhor qualidade em cada projeto (ARTHUR, 1993; SWEBOK, 2004).

4.3. A Abordagem de Qualidade de *Software* do Processo Unificado em Relação a Modelos e Padrões de Qualidade de *Software*

Para obter uma visão mais específica da abordagem de qualidade do UP faz-se necessário compará-la com outras abordagens de qualidade de *software*. Este trabalho analisa o Processo Unificado em relação a duas abordagens que tratam de qualidade de *software*.

A primeira abordagem é a da ISO/IEC 12207 (1995, 2002), que trata dos processos do ciclo de vida de *software*, e é largamente utilizada como guia na definição de processos de *software*. Reinehr *et al.* (2003) analisaram como o RUP (versão anterior a 2003) pode ser utilizado para implementar a norma ISO/IEC 12207 (1995, 2002).

A segunda abordagem é a do CMMI (*Capability Maturity Model Integration*) (CMMI, 2002a; CMMI, 2002b), que foi proposto pelo SEI (*Software Engineering Institute*) da Universidade *Carnegie Mellon*, e tem sido amplamente utilizado por organizações que desenvolvem *software* visando a exportação para o mercado americano e europeu.

4.3.1. A qualidade de *software* do UP em relação à ISO/IEC 12207

Para este trabalho, foram selecionados os seguintes processos da ISO/IEC 12207 relacionados à qualidade de processo de *software*:

- *processos de suporte*: garantia de qualidade, verificação, validação, revisão conjunta, auditoria e resolução de problemas.
- *processos organizacionais*: gerenciamento, focando as atividades de revisão e avaliação, gestão da qualidade, e métricas.

A Tabela 4.6 apresenta um mapeamento entre terminologias da ISO/IEC 12207 (1995, 2002) e do RUP (2003). A ISO/IEC 12207 possui uma abordagem mais genérica, especificando processos. Ela não especifica técnicas, métodos, documentos ou *templates*, podendo ser utilizada para qualquer ciclo de vida. Já o UP é direcionado para o ciclo de vida iterativo incremental, estabelecendo atividades, atores, guias, artefatos e *templates*, que auxiliam a implementação do processo.

Tabela 4.6 – Mapeamento de terminologias da ISO/IEC 12207 e do UP

Terminologia	ISO/IEC 12207 (2002)	RUP (2003)
Ciclo de Vida	Descreve a estrutura e a arquitetura dos processos. Não recomenda o tipo de ciclo de vida a ser utilizado.	É utilizado para descrever o Processo Unificado, dentro de uma abordagem iterativa incremental.
Atividades	Conjunto de tarefas.	Corresponde às atividades do detalhamento da disciplina (macro-atividade).
Tarefas	Conjunto de passos elementares a serem executados.	Corresponde aos passos das atividades.
Processo	Abrange um conjunto de atividades.	Similar ao conceito de disciplina (*workflow*) do UP
Saídas	Resultado de uma atividade. Podem ser *entregáveis* ou *não-entregáveis*.	Resultado de uma atividade, correspondendo a um artefato do UP.

Terminologia	ISO/IEC 12207 (2002)	RUP (2003)
Tipos de processos	Distingue dois tipos de processos: processos de suporte, e processos organizacionais.	Não faz distinção entre processos de suporte e processos organizacionais. Trata dos processos organizacionais estabelecidos na norma como processos de suporte.
Infra-estrutura de processo	Possui um processo de infra-estrutura	Presente na disciplina de ambiente.

A seguir, nas Tabelas 4.7 e 4.8, a versão do RUP (2003) é analisada em relação à ISO/IEC 12207 (1995, 2002).

Tabela 4.7 – Mapeamento entre os processos de suporte do padrão ISO/IEC 12207 (1995, 2002) e o UP (2003)

ISO/IEC 12207 **Processo:** atividade	**Processo Unificado** **Disciplina:** *macro-atividade* **(**atividade)	**Análise em relação ao UP**
Garantia de Qualidade: Implementação do Processo.	**Gerenciamento de Projetos**: Planejar projeto (Desenvolver Plano de Garantia de Qualidade).	**Parcialmente coberto.** No RUP (2003), o gerente de projetos é responsável por criar o Plano de Garantia de Qualidade. A norma estabelece que as atividades de garantia de qualidade sejam executadas por indivíduos independentes em relação ao que está sendo auditado. O Plano de Garantia de Qualidade possui uma seção para registro das atividades de revisões e auditorias a serem executadas no projeto como atividades de garantia de qualidade, mas não estabelece que elas devem ser executadas por um papel independente da estrutura dos projetos. Não são especificadas atividades para realizar revisões e auditorias.
Garantia de Qualidade: Garantia do Produto	**Modelagem de Negócios**: Refinar definição do processo de negócio (Revisar modelo de Caso de Uso de Negócio). **Requisitos**: Gerenciar mudanças nos Requisitos (Revisar Requisitos). **Análise e Projeto**: Refinar Arquitetura (Revisar Arquitetura); Projetar Componentes (Revisar projeto). **Testes**: Validar estabilidade do *build* (Executar testes da suíte). **Implementação**: Implementar componentes (Revisar código). **Gerenciamento de Projetos**: Planejar projeto (Desenvolver Plano de Aceitação do Produto) **Implantação**: Desenvolver material de suporte (Desenvolver material de suporte); Desenvolver material de suporte (Desenvolver material de treinamento).	**Parcialmente coberto.** A atividade Gerenciar Teste de Aceitação contém a responsabilidade de assegurar que o produto satisfaz os critérios pretendidos que foram estabelecidos no artefato Plano de Aceitação de Produto. No RUP deve ser verificada a consistência entre os conjuntos de artefatos de diversas disciplinas, mas não há informações de como garantir que os produtos de *software* e sua documentação estejam aderentes aos planos e ao contrato.
Garantia de Qualidade: Garantia do Processo	**Não coberto.**	**Não coberto** Não há atividades específicas no RUP para verificar a situação corrente do processo.

ISO/IEC 12207 **Processo:** atividade	**Processo Unificado** **Disciplina:** *macro-atividade* **(**atividade)	**Análise em relação ao UP**
Garantia de Qualidade: Sistema de gestão da qualidade (SGQ)	**Não coberto**	**Não coberto** Esta atividade na ISO/IEC 12207 refere-se ao conceito da ISO 9001 (2000) de Sistemas de Gestão da Qualidade (SGQ). Não há atividades específicas no RUP para tratar o SGQ.
Verificação: Implementação do processo	As atividades de revisão do RUP estão descritas nas Tabelas 4.4 e 4.5.	**Completamente coberto.** As atividades de revisão técnica e revisão gerencial do RUP podem ser vistas como atividades que provêem verificação.
Verificação: Verificação	As atividades de revisão do RUP estão descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** Algumas revisões do UP verificam requisitos do sistema, requisitos de planejamento, código. A Revisão do PRA (*Project Review Authority*) avalia os contratos, mas não há revisão de verificação do processo no RUP.
Validação: Implementação do processo	**Gerenciamento de Projetos**: Planejar projeto (Desenvolver Plano de Aceitação do Produto)	**Completamente coberto.** A norma estabelece que as atividades de validação possam ser executadas com vários níveis de independência. O processo de validação no RUP utiliza revisões. O Plano de Aceitação de Produto descreve como o cliente avaliará os artefatos que receberá do projeto para determinar se atendem a um conjunto pré-definido de critérios de aceitação.
Validação: Validação	**Testes** (toda a disciplina) **Gerenciamento de projetos**: Fechamento do projeto (Revisão de Aceitação do Projeto)	**Completamente coberto.** A norma foca a validação por meio dos Testes. A disciplina de Testes, combinada com a atividade de fechamento do projeto, atende completamente à necessidade da norma.
Revisão Conjunta: Implementação do processo	As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Completamente coberto.** Esta atividade provê os procedimentos de revisão para cada fase e iteração. O padrão não prescreve o envolvimento do cliente como requisito mandatório.
Revisão Conjunta: Revisões de Gerenciamento de Projeto	As atividades de revisão do RUP descritas na Tabela 4.5.	**Completamente coberto.** No RUP são realizadas revisões gerenciais (Tabela 4.5), que podem ser feitas tanto pelo gerente de projetos, como pelo PRA, ou ser delegada pelo PRA a algum outro papel do RUP.
Revisão Conjunta: Revisões Técnicas	As atividades de revisão do RUP estão descritas na Tabela 4.4.	**Completamente coberto.** No RUP são realizadas revisões técnicas (Tabela 4.4), que estão relacionadas com a revisão dos artefatos, para validar produtos de *software*.
Auditoria: Implementação do processo	**Não coberto**	**Não coberto** Não há atividades específicas no RUP.
Auditoria: Auditoria	**Não coberto**	**Não coberto** Não há atividades específicas no RUP.
Resolução de Problemas: Implementação do Processo	**Gerenciamento de Projeto**: Planejar Projeto (Desenvolver Plano de Resolução de Problemas)	**Completamente coberto.** O Plano de Resolução de Problemas provê um procedimento definido para gerenciar e resolver problemas identificados durante o projeto, para a tomada de ações corretivas.

ISO/IEC 12207 Processo: atividade	**Processo Unificado Disciplina:** *macro-atividade* **(**atividade)	**Análise em relação ao UP**
Resolução de Problemas: Resolução de Problemas	**Gerenciamento de Configuração e Mudanças**: Gerenciar solicitações de mudança (Submeter pedidos de mudança); Gerenciar solicitações de mudança (Atualizar solicitações de mudança); Gerenciar solicitações de mudança (Revisar solicitações de mudança). **Gerenciamento de Projetos**: Monitorar e controlar projeto (Lidar com problemas e exceções).	**Completamente coberto.** Problemas detectados são submetidos para qualquer papel, originados da equipe de desenvolvimento. O proprietário da solicitação ou o CCB (*Change Control Board*) podem atualizá-lo. O CCB revisa as solicitações de mudança. E ações corretivas são tomadas.

A Tabela 4.8 apresenta o mapeamento do RUP (2003) em relação aos processos organizacionais da ISO/IEC 12207 relacionados à qualidade do processo.

Tabela 4.8 – Mapeamento entre os processos organizacionais da ISO/IEC 12207 (1995, 2002) e o UP (2003)

ISO/IEC 12207 Processo: atividade	**Processo Unificado Disciplina:** *macro-atividade* **(**atividade)	**Análise em relação ao UP**
Gerenciamento: Revisão e Avaliação	**Gerenciamento de Projetos:** Gerenciar iteração (Avaliar iteração); Gerenciar iteração (Revisar critérios de aceitação da iteração); Fechamento da Fase (Revisão do marco de ciclo de vida); Fechamento do Projeto (Revisão do marco de ciclo de vida).	**Completamente coberto.** Cada iteração e fase do RUP são finalizadas com uma revisão para aprovar a iteração ou fase. Em paralelo, o RUP possui vários tipos de revisões gerenciais (Tabela 4.5), que podem ser realizadas tanto pelo Gerente de Projetos, como pelo PRA, ou delegada pelo PRA a algum outro papel do RUP.
Gerenciamento: Gerenciamento de Qualidade	**Gerenciamento de Projetos:** Planejar Projeto (Desenvolver Plano de Garantia de Qualidade de Software); Planejar Projeto (Definir Processos de monitoração e controle).	**Parcialmente coberto.** A qualidade do produto é coberta pelas práticas do RUP. Como o foco do RUP é o processo de desenvolvimento, ele não provê um processo organizacional de gerenciamento da qualidade, não atendendo completamente ao que é solicitado na norma. Pode-se acrescentar, ainda, que gerenciar qualidade, de acordo com a ISO/IEC 12207, significa satisfazer os clientes, monitorando a qualidade de produtos e serviços, em nível organizacional e de projeto, para assegurar que a qualidade atenda os requisitos do cliente. Para gerenciar a qualidade, deve-se estabelecer objetivos de qualidade, implantar um SGQ, realizar atividades de garantia de qualidade (QA) e controle de qualidade (CQ) e monitorar o atingimento dos objetivos de qualidade, tomando as devidas ações corretivas. O RUP, na disciplina de Testes trata as atividades de controle de qualidade, mas não trata a garantia de qualidade, nem como essas atividades estariam relacionadas dentro de uma estratégia de qualidade, que, aliadas a métricas e objetivos de

ISO/IEC 12207 Processo: atividade	**Processo Unificado Disciplina:** *macro-atividade* **(**atividade)	**Análise em relação ao UP**
		qualidade, pudessem ser especificadas como Gerenciamento da Qualidade.
Gerenciamento: Métricas	**Gerenciamento de Projetos:** Planejar Projeto (Desenvolver Plano de Métricas); Planejar Projeto (Definir Processos de monitoração e controle); Monitorar e controlar projeto (Monitorar status do projeto).	**Parcialmente coberto.** A disciplina de Gerenciamento de Projetos provê uma atividade específica para definir o Plano de Métricas e também processos para monitorar e controlar o uso de métricas. Como o RUP foca o processo de desenvolvimento, e não provê um processo organizacional, o aspecto de métricas requerido pela norma não é completamente atendido. O Plano de Métricas continua contemplando as métricas no escopo do projeto, sendo na Fase de Iniciação, durante as atividades de planejamento. É estabelecido que as métricas devem ser coletadas e registradas.

4.3.2. A abordagem de qualidade de *software* do UP em relação ao modelo CMMI

São analisadas, a seguir, disciplinas, atividades e conceitos do UP em relação às áreas de processo do CMMI (CMMI, 2002a; CMMI, 2002b), em termos de abordagem relacionada com a qualidade de *software*. Por ser um modelo de maturidade e capacidade de processo de *software*, o CMMI está estruturado de acordo com uma visão de processos que considera processos de *software* implementados tanto em nível de projeto como organizacional. As áreas de processo selecionadas para este mapeamento estão na Tabela 4.9.

Tabela 4.9 – Áreas de Processo CMMI selecionadas para mapeamento do UP

Área de Processo do CMMI	**Objetivo da Área de Processo no CMMI**	**Importância da área de processo para a qualidade de *software***
Verificação	Objetiva garantir que produtos de trabalho selecionados atendam aos requisitos especificados	Representa no CMMI um dos processos que compõem a visão de gestão da qualidade da ISO/IEC 12207: verificação
Validação	Objetiva demonstrar que o produto é adequado para o uso pretendido, executado no seu ambiente de destino.	Representa no CMMI um dos processos que compõem a visão de gestão da qualidade da ISO/IEC 12207: validação
Garantia da Qualidade do Processo e do Produto	Objetiva prover a técnicos e gerentes com informações objetivas sobre os processos e seus produtos de trabalho associados.	Representa no CMMI um dos processos que compõem a visão de gestão da qualidade da ISO/IEC 12207: garantia de qualidade
Medição e Análise	Objetiva desenvolver e manter uma capacidade de medição utilizada para dar suporte às necessidades de informação da gerência.	Representa no CMMI um dos processos relacionados com qualidade de software, já que não é possível falar em qualidade de *software* sem tratar do uso de medições. (Schulmeyer & McManus, 1999)

Apesar das duas áreas de processo do CMMI nível 4 tratarem da gestão da qualidade, essas áreas não entraram no escopo deste mapeamento em função da gestão de qualidade realizada no nível 4 do CMMI ter como base o uso de processos estatísticos para controle de processo de *software*, que estão fora do escopo deste trabalho.

As áreas de processo do CMMI possuem objetivos genéricos e objetivos específicos que devem ser atendidos, constituindo componentes requeridos pelo modelo. Esses objetivos podem ser atendidos por meio de práticas genéricas ou específicas, que são componentes esperados pelo modelo. Dessa forma, os processos e as atividades do UP serão mapeados em relação ao cumprimento das práticas esperadas pelo CMMI.

Os quadros a seguir apresentam os mapeamentos realizados entre as quatro áreas de processo da Tabela 4.9 e o UP.

Validação			
Objetivo específico	**Prática específica**	**RUP Disciplina (**Atividade**)**	**Análise**
Preparar a Validação	Selecionar produtos de trabalho para validação	**Testes**	**Completamente coberto.**
	Estabelecer o ambiente de validação	**Testes**	**Completamente coberto.**
	Estabelecer procedimentos e critérios de validação	**Testes** As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** Os *checklists* utilizados nas revisões provêem critérios a serem utilizados. Não são especificados procedimentos para manutenção e treinamento.
Validar produtos ou componentes	Executar validação	**Testes** As atividades de revisão do RUP descritas nas tabelas 4.4 e 4.5.	**Completamente coberto.**
	Analisar resultados da validação	**Testes** As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Completamente coberto.**

Quadro 4.1 – Área de Processo Validação em relação ao UP

Medição e Análise			
Objetivo específico	**Prática específica**	**RUP Disciplina (**Atividade**)**	**Análise**
Alinhar métricas e atividades de análise	Estabelecer objetivos de medição	**Gerenciamento de Projetos**: Planejar Projeto (Desenvolver Plano de Métricas).	**Completamente coberto.** O RUP estabelece que sejam especificados os objetivos do projeto e que métricas sejam definidas para verificar o atendimento desses objetivos
	Especificar métricas	**Gerenciamento de Projetos**: Planejar Projeto (Desenvolver Plano de Métricas).	**Completamente coberto.** As métricas são registradas no Plano de Métricas do projeto, que é tido como um artefato opcional.
	Especificar coleta de dados e procedimentos de armazenamento	**Gerenciamento de Projetos**: Planejar Projeto (Desenvolver Plano de Métricas); Monitorar e Controlar Projeto (Monitorar Status do Projeto).	**Completamente coberto.** O RUP indica que os procedimentos de coleta devem ser registrados no Plano de Métricas. As métricas devem ser armazenadas no artefato Métricas do projeto, que consiste de um repositório contendo as métricas de projeto. A necessidade de se ter um procedimento de coleta e de análise de métricas também é especificada no item Conceitos sobre Métricas do RUP.
	Especificar procedimentos de análise	**Gerenciamento de Projetos**: Monitorar e Controlar Projeto (Monitorar Status do Projeto).	**Completamente coberto.** A especificação de procedimentos de análise deve ser feita no artefato Plano de Métricas. O procedimento de análise de métricas é tratado na atividade Monitorar Status do Projeto e no item Conceitos sobre Métricas do RUP.
Prover resultados de medição	Coletar dados de métricas	**Gerenciamento de Projetos**: Monitorar e Controlar Projeto (Monitorar Status do Projeto).	**Completamente coberto.** O RUP indica que deve ser realizada uma vez a cada iteração.
	Analisar dados de métricas	**Gerenciamento de Projetos**: Monitorar e Controlar Projeto (Monitorar Status do Projeto).	**Completamente coberto.**
	Armazenar dados e resultados	**Gerenciamento de Projetos**: Planejar Projeto (Desenvolver Plano de Métricas).	**Completamente coberto.** Os dados sobre métricas devem ser armazenados no repositório de métricas do projeto.
	Comunicar resultados	**Gerenciamento de Projetos**: Monitorar e Controlar Projeto (Reportar Status).	**Completamente coberto.**

Quadro 4.2 – Área de Processo Medição e Análise em relação ao UP

Verificação			
Objetivo específico	**Prática específica**	**RUP Disciplina (**Atividade**)**	**Análise**
Preparar a Verificação	Selecionar produtos de trabalho para verificação	**Testes** As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** As Revisões e a disciplina de Testes no RUP indicam os produtos de trabalho a serem considerados nessas atividades. Os métodos de verificação são providos na disciplina de Testes, mas nem todas as atividades de Revisão especificam os métodos a serem seguidos na revisão. Os *guidelines* e *checklists* provêem requisitos que devem ser atendidos por cada produto de trabalho.
	Estabelecer o ambiente de verificação	**Testes** As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** Embora coberto para a disciplina de Testes, não especifica, como deve ser o ambiente de verificação durante as revisões, por exemplo.
	Estabelecer procedimentos e critérios de verificação	As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** Os *checklists* utilizados nas revisões provêem critérios a serem utilizados. As *guidelines* para Revisões no RUP podem ser vistas como os procedimentos para realizar verificação, apesar de serem aplicáveis a qualquer tipo de revisão, sem considerar as características especificas de cada tipo de revisão.
Executar Revisões por Pares	Preparar revisões por pares	**Implementação**: Implementar componentes (Revisar Código).	**Parcialmente coberto.** Não são especificados como deverão ser coletados dados durante as revisões, nem critérios de entrada e saída das revisões, nem quais seriam os critérios a serem utilizados para realização de reuniões posteriores. Apenas no caso de Revisão de Código é indicada a técnica de revisão por pares.
	Conduzir Revisões por pares	**Implementação**: Implementar componentes (Revisar Código).	**Parcialmente coberto.** Devido à ausência de critérios para indicar necessidade de realização de revisões posteriores e de critérios de saída das revisões.
	Analisar dados sobre as revisões por pares	**Não coberto**	**Não coberto** Não há atividade específica no RUP
Verificar produtos de trabalho selecionados	Executar verificação	**Testes** As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Completamente coberto.**
	Analisar os resultados da verificação e identificar ações corretivas	**Não coberto**	**Não coberto** Não há atividade específica no RUP

Quadro 4.3 – Área de Processo Verificação em relação ao UP

Garantia de Qualidade do Processo e do Produto			
Objetivo específico	**Prática específica**	**RUP Disciplina (**Atividade**)**	**Análise**
Avaliar processos e produtos de trabalho objetivamente	Avaliar processos objetivamente	**Não coberto**	**Não coberto** Não há atividade específica no RUP
	Avaliar objetivamente produtos de trabalho e serviços	**Não coberto**	**Não coberto** Não há atividade específica no RUP
Prover *insight* objetivo	Comunicar e garantir resolução de não conformidades	**Gerenciamento de Projetos**: Planejar Projeto (Desenvolver Plano de Resolução de Problemas); Monitorar e Controlar Projeto (Lidar com Exceções e Problemas).	**Parcialmente coberto.** Problemas identificados durante as revisões são relatados para resolução. Entretanto não é especificado que são realizadas análises sobre não conformidades, que elas são rastreadas até resolução e que periodicamente são revistas com os *stakeholders*.
	Estabelecer registros	As atividades de revisão do RUP descritas nas Tabelas 4.4 e 4.5.	**Parcialmente coberto.** O RUP especifica que os resultados das Revisões devem ser registrados no artefato Registro das Revisões que, no entanto, pode ser utilizado com diferentes níveis de formalidade de acordo com o projeto.

Quadro 4.4 – Área de Processo Garantia de Qualidade do Processo e do Produto em relação ao UP

Observa-se que o UP abrange as práticas relacionadas principalmente com a qualidade do produto. Práticas relacionadas com a qualidade do processo no UP são deficientes em relação ao que é requerido pelos modelos e padrões de qualidade de *software*.

4.4. Deficiências na abordagem de qualidade de *software* do UP

Alguns pesquisadores apontam algumas deficiências no RUP na abordagem de qualidade de *software*. Algumas delas são apresentadas a seguir.

Manzoni & Price (2003) analisaram o Processo Unificado em relação ao modelo SW-CMM. Eles identificaram que o UP não estabelece como tratar os problemas identificados nas revisões, nem como as ações corretivas são realizadas. Segundo o mesmo estudo, o UP também não aborda como os desvios são identificados, documentados e acompanhados até sua conclusão. O UP também não trata como os resultados das revisões são comunicados aos demais envolvidos com o projeto de *software* e aos envolvidos com o processo de *software*, como o grupo de Engenharia de *Software*. Também não seria provido o acompanhamento dos custos e das atividades de qualidade de *software* a serem definidos no Plano de Garantia de Qualidade do projeto.

Segundo Reinehr *et al.* (2003), o Processo Unificado não apresenta atividades que implementem um processo de garantia de qualidade do processo, e não requerem independência para realização de atividades de revisão de *software*.

Smith (2003) analisou as características do UP em relação ao padrão IEEE 1074 (IEEE 1074, 1997) que trata do Ciclo de Vida de *Software*. De acordo com este trabalho, o UP possui várias carências relacionadas à qualidade do processo:

- ausência de formalização das avaliações de qualidade, especialmente em termos dos tipos de revisões realizadas;
- ausência de aprovação de desvios identificados em relação ao processo padrão (considerado como sendo constituído por todos os processos que compõem o *framework* do RUP);
- uso de avaliações com foco na melhoria do processo de *software* utilizado.

Esta análise sugere que os aspectos de qualidade abordados na disciplina de testes, que são bem cobertos, sejam aliados à abordagem das avaliações de revisão ao longo do processo, que são tratadas informalmente no RUP, e constituam uma nova disciplina de avaliação (SMITH, 2003).

4.5. Propostas para a extensão do Processo Unificado

Alguns autores fizeram propostas de extensão e melhoria do Processo Unificado. Algumas melhorias estão relacionadas com técnicas; outras propõem novas disciplinas, ou até mesmo novas fases para o Processo Unificado.

Massoni *et al.* (2002) estendem o UP com um método que suporta a implementação progressiva de aplicações persistentes, concorrentes e distribuídas, influenciando tanto a estrutura dinâmica como a estrutura estática do Processo Unificado.

Sousa & Furtado (2003) expandem o RUP propondo um processo de desenvolvimento de *software* para sistemas interativos. O novo processo, denominado RUP*i*, inclui conceitos de interação humano-computador (IHC) garantindo usabilidade, acessibilidade e aceitabilidade com foco nos usuários finais, no contexto de uso do sistema.

Phillips & Kemp (2002) também estendem o RUP dentro do contexto de projeto e usabilidade das interfaces de usuário final, propondo dois artefatos de suporte às duas principais atividades que envolvem interface no RUP: a modelagem e a prototipação.

O *Enterprise Unified Process*, (EUP, *Enterprise-RUP* ou ainda E-RUP) propõe uma extensão para o Processo Unificado considerando as necessidades das "organizações de Tecnologia da Informação do mundo real" (AMBLER, 2002). As extensões seriam classificadas em duas categorias. A primeira tornaria o RUP completo em relação ao suporte ao ciclo de vida do *software*, com a inclusão de duas novas fases abrangendo a operação do sistema (*Production*) e a descontinuação do mesmo (*Retirement*) e de uma disciplina adicional de suporte denominada Operação e Suporte (*Operations & Support*). A segunda extensão tornaria o UP completo em termos de suporte ao ciclo de vida de Tecnologia da Informação, acrescentando um conjunto de disciplinas agrupadas sob o título de *Enterprise Management*. Essa segunda proposta insere, no contexto do UP, a necessidade das disciplinas tratarem também de aspectos organizacionais, e não serem apenas voltadas para o nível do projeto de *software*.

4.6. Conclusão

Neste capítulo, foram apresentados diversos aspectos envolvendo a qualidade de *software* no RUP (2003), incluindo conceitos, processos e atividades, a compatibilidade do processo a padrões de qualidade de *software*, deficiências apontadas por outros autores à abordagem de qualidade do UP e ainda propostas estendendo-o e melhorando-o.

Neste contexto, o próximo capítulo apresenta a proposta de uma nova disciplina para o Processo Unificado: o Gerenciamento da Qualidade, cujo objetivo é estabelecer o fluxo de ações que conduzam à qualidade do processo de *software*, suprindo algumas das deficiências encontradas no UP em relação à garantia da qualidade de *software* e ao gerenciamento de qualidade.

Capítulo 3

GARANTIA DE QUALIDADE DE *SOFTWARE*

Este capítulo aborda a Garantia da Qualidade de Software (GQS), também denominada Software Quality Assurance (SQA), sua importância e suas abordagens.

Para melhorar a qualidade do *software* são utilizadas estratégias de qualidade. De acordo com Belchior (1997), estratégias de qualidade são selecionadas com base em alguns critérios: a busca em melhorar processos, já que alguns erros no produto são conseqüências do processo utilizado; o esforço para realizar a identificação e correção de defeitos; e por fim a identificação da causa dos erros que produziram defeitos.

Estratégias de Qualidade são identificadas também como esforços de Garantia de Qualidade de *Software* (GQS) ou *Software Quality Assurance* (SQA). A garantia de qualidade do *software* está diretamente relacionada às características de qualidade do processo de desenvolvimento e de seus produtos intermediários, bem como aos esforços de melhoria no processo de *software* das organizações. A garantia de qualidade de *software*, doravante denominada apenas SQA, é a área da engenharia de *software* que abrange procedimentos, métodos, ferramentas, técnicas e estratégias para garantir a adequação de processos de desenvolvimento de *software* e produtos de *software* aos padrões de desenvolvimento de *software* estabelecidos para uma organização (PRESSMAN, 2000).

O padrão IEEE 610.12 (1990) também estabelece dois conceitos para SQA. O primeiro afirma que SQA consiste em um "planejado e sistemático modelo de todas as ações necessárias para prover adequada conformidade de um item ou produto a requisitos técnicos estabelecidos". O segundo é de que SQA é um "conjunto de atividades projetadas para avaliar o processo pelo qual os produtos são desenvolvidos ou manufaturados".

A garantia de qualidade de *software* é uma atividade presente ao longo de todo o ciclo de vida de *software*, a fim de garantir que o projeto, o desenvolvimento e a disponibilização de uma aplicação aconteçam de forma bem sucedida (FELDMAN, 2005).

Embora haja um consenso sobre a necessidade de qualidade no desenvolvimento de *software*, na prática, SQA é tratada como algo que se deixa para depois, sem prioridade e com recursos insuficientes. SQA é considerada um fator crítico de sucesso para a melhoria dos processos e

contribui diretamente para a melhoria do nível de maturidade das organizações (CRAFT, 2001). Outro aspecto interessante sobre SQA é a sua forma de atuação, podendo ser desempenhado tanto por meio de um papel de "polícia de processo" (BAKER, 2001), como por meio de um papel de *coaching* e *mentoring* para projetos, em uma relação de parceria entre SQA e a equipe do projeto (PIRES *et al*., 2004).

3.1. Características de SQA

Garantir a qualidade de *software* envolve não apenas a definição do processo de *software* da organização, mas também a implementação de um processo de garantia da qualidade de *software* adequado às necessidades da organização, com a realização da garantia da qualidade do processo e da garantia da qualidade do produto (ROCHA *et al*., 2001). O produto de *software* deve ser entendido não apenas como o produto de *software* executável resultante do processo de desenvolvimento, mais ainda como sendo representado por todos os artefatos finais ou intermediários (procedimentos, documentos, modelos, código) produzidos ou utilizados ao longo do processo (ISO, 1995; JACOBSON *et al*., 1998).

A garantia da qualidade do processo de *software* visa prover evidências sobre a capacidade do processo em produzir determinado produto, identificando falhas nesse processo e buscando resolvê-las antes que impactem no produto. A garantia da qualidade do produto de *software*, por sua vez, visa garantir que o *software* produzido esteja em conformidade com requisitos funcionais e de desempenho especificados; atenda aos padrões de desenvolvimento documentados; seja o mais isento possível de erros; e atenda às características implícitas esperadas pelo usuário (PRESSMAN, 2000; SOMMERVILLE, 2003; SWEBOK, 2004). A estruturação da função de qualidade de *software*, utilizando SQA, permite acompanhar se o processo de *software* definido está sendo realmente utilizado e também avaliar quão aderentes estão as práticas dos projetos ao processo estabelecido.

As pessoas mais estreitamente ligadas a prover apoio à execução das atividades de garantia de qualidade constituem a equipe de SQA. Para atuar como SQA são necessários dois tipos de conhecimento: o conhecimento técnico de engenharia de *software* e o conhecimento organizacional. O conhecimento técnico abrange os seguintes itens: engenharia de *software*; qualidade de *software*; processo de desenvolvimento: ciclo de vida, etapas e fases; metodologias e técnicas; testes e critérios de qualidade para processo e produto. O conhecimento organizacional abrange as informações acerca da estrutura hierárquica da empresa, os níveis de gerenciamento, a distribuição

de responsabilidades entre os diversos níveis hierárquicos, o conhecimento da cultura da organização, suas políticas e seus padrões institucionais (SCHULMEYER & MCMANUS, 1999).

A discussão em torno do caráter de disciplina ou da definição da função de qualidade na organização é tratada por Baker (2001). O autor conclui que é difícil caracterizar SQA como uma disciplina (do ponto de vista da formalização do processo de SQA), devido às diferentes práticas das instituições em relação à garantia de qualidade, e que nem sempre SQA é um grupo. Dependendo da política estabelecida pela organização, os responsáveis pela execução de SQA podem trabalhar de forma exclusiva ou em dedicação de tempo parcial (SCHULMEYER & MCMANUS, 1999), desde que os critérios de independência e objetividade em relação aos projetos avaliados não sejam comprometidos.

Padrões de qualidade são muito importantes para a garantia de qualidade. Eles são definidos para exprimir as melhores práticas da organização e também para auxiliar os projetos a trabalharem com um nível de qualidade adequado às suas necessidades (SOMMERVILLE, 2003). Os padrões podem ser categorizados como padrões de processo (procedimentos) e padrões de produto (padrões, propriamente ditos). A categorização facilita a atualização, melhoria, criação de novos padrões ou exclusão de padrões vigentes não adequados à realidade da organização. Procedimentos podem ser considerados como padrões de processo, enquanto que os padrões de produto abrangem padrões de documentos, de codificação e de como estruturar a documentação dos processos (SOMMERVILLE, 2003). Os procedimentos especificam a forma como deverão ser executados processos e atividades (ISO, 2000). Isso significa que os procedimentos possuem um papel fundamental na especificação de um processo de *software* com qualidade, definindo os processos a serem seguidos durante o desenvolvimento e assegurando que os padrões de produtos sejam seguidos.

Métricas auxiliam as atividades de garantia de qualidade, já que só é possível melhorar aquilo que se consegue medir. O uso das métricas reduz a subjetividade na avaliação da qualidade de *software*, provendo uma base quantitativa sobre a qual tomar decisões (SCHULMEYER & MCMANUS, 1999). Métricas podem ser aplicadas tanto ao processo como ao produto. Medições históricas facilitam o processo de estimativas e de gerenciamento do processo de *software*.

As características descritas estão presentes em modelos, técnicas e padrões desenvolvidos para trabalhar SQA, que são tratados na seção a seguir.

3.2. Técnicas, Modelos e Padrões de SQA

Enfocando a Garantia da Qualidade de *Software*, existem técnicas, padrões e modelos de qualidade que foram propostos para realizar SQA do processo e/ou SQA do produto. Serão descritas a seguir abordagens relevantes relacionadas à SQA, presentes na literatura de engenharia de *software*.

3.2.1. Técnicas de SQA

Técnicas são procedimentos gerenciais e técnicos que auxiliam na avaliação e na melhoria do processo de desenvolvimento de *software* (IEEE 610.12, 1990). As técnicas de SQA são um dos tipos de estratégias utilizados para garantir a qualidade de *software*, seja do processo, seja do produto. As técnicas de SQA avaliam processos, encontram defeitos no produto diretamente, ou indicam a necessidade de um exame mais detalhado de produtos.

Técnicas de SQA são categorizadas como: estáticas, quando não envolvem execução de código; e como dinâmicas, quando sua aplicação é baseada na execução de código. Técnicas Estáticas realizam o exame da documentação do projeto e avaliam informações sobre o produto de *software* sem executá-lo. Elas se classificam em (SWEBOK, 2001):

- Técnicas *People-Intensive*: dependem fundamentalmente de pessoas, requerendo, no mínimo, duas pessoas para conduzi-la. São exemplos as Revisões de *Software*, como *Walkthrough* e Inspeção.
- Técnicas Analíticas: possuem propósitos variados e nem todas podem ser aplicadas a todo o projeto. São aplicadas por um individuo, de forma manual ou automatizada, a diferentes partes de um produto ou a diferentes etapas de um projeto. São exemplos deste tipo de técnica: *Assessment* (avaliação), Análise de Fluxo de Controle e Métodos Formais.

As Técnicas Dinâmicas, por sua vez, executam o produto de *software* para avaliá-lo. São aplicadas ao longo do processo de desenvolvimento e da manutenção de *software*. São exemplos de Técnicas Dinâmicas: Testes, Simulação, Checagem de Modelos, e Execução Simbólica.

A equipe de SQA é responsável pela seleção, dentre as diversas técnicas existentes, daquelas que serão utilizadas no processo de SQA. A seleção de técnicas de SQA é baseada em dois fatores principais: a análise do custo-efetividade de cada técnica e a análise do custo de severidade das falhas identificáveis pela técnica (XIE & NOTKIN, 2002).

Para a seleção das técnicas, com relação ao custo-efetividade, procura-se avaliar:

- A redução de custos enquanto se mantém uma efetividade razoável na aplicação da técnica. Há técnicas de SQA que requerem um maior investimento quando do início de sua aplicação. Com o passar do tempo, o custo diminui e a efetividade é mantida em patamares razoáveis. Um exemplo deste caso é o uso de métodos formais em SQA.
- O aumento da efetividade, enquanto o custo é mantido de forma razoável. A efetividade depende de domínios de aplicação ou de aplicações específicas, assim como dos tipos de falhas existentes. A Inspeção é um exemplo de técnica onde o custo de aplicação da técnica se mantém, enquanto há aumento na efetividade da técnica.

Com relação ao custo de severidade das falhas, são avaliados: os tipos de falhas que devem ser priorizados e o grau de severidade dessas falhas; e quais as técnicas de SQA que poderão detectar essas falhas. Ao final, comparam-se os custos de utilização de determinadas técnicas em relação ao custo de severidade da existência de falhas, para que sejam selecionadas as técnicas mais efetivas em virtude dos objetivos pretendidos, de minorar a existência e o efeito das falhas.

Existem duas formas de melhorar a análise custo/efetividade para a seleção e a aplicação de técnicas de SQA: aumentando a interação entre diferentes técnicas de SQA e realizando a integração das técnicas de SQA (XIE & NOTKIN, 2002). A interação entre as diversas técnicas de SQA é dificultada, porque não há uma visão de que elas possam ser usadas de forma cooperativa. Técnicas de SQA são utilizadas comumente como métodos separados que competem entre si. A integração das técnicas de SQA, por sua vez, já vem sendo realizada por meio da combinação de técnicas. Essa combinação consiste de uma técnica de SQA utilizar produtos/artefatos intermediários ou finais de uma outra técnica de SQA. Quando uma técnica utiliza produtos intermediários de outra, a integração entre as técnicas é dita de baixo acoplamento. Quando utiliza produtos ou artefatos finais, é chamada integração de alto acoplamento (XIE & NOTKIN, 2002).

As principais técnicas de SQA são:

- Revisões de *Software*: avaliam o processo de desenvolvimento, seu gerenciamento, e o produto de *software*. São classificadas em Auditorias, Inspeções, *Walkthrough*s, Revisões Técnicas e Revisões Gerenciais (IEEE 1028, 1997).

- Avaliação (*Assessment*): é uma forma de examinar um processo, determinando sua capacidade por meio da comparação desse processo em relação a um padrão de melhores práticas.
- Testes: constituem um dos elementos críticos da garantia de qualidade de *software* e representam a última revisão das especificações de projeto e de código do produto de *software* (PRESSMAN, 2000). No entanto, realizar testes não significa por si só possuir SQA (FELDMAN, 2005).
- Técnicas de Análise de Processo: visam prover análises do processo, buscando a prevenção de defeitos e a identificação da estabilidade do processo. Incluem ainda os Métodos Formais aplicados à SQA, que utilizam a formalização matemática para demonstrar a qualidade existente no *software*. São exemplos dessas técnicas: Prevenção de Defeitos e Análise de Causas, Análise de Predição de Confiabilidade, e Análise de Controle Estatístico do Processo.

Estas técnicas são detalhadas a seguir.

3.2.1.1. Revisões de *Software*

As Revisões de *Software* são procedimentos para avaliar o *software*, seu processo de desenvolvimento, assim como o processo de gerenciamento desse desenvolvimento. Auxiliam a avaliação do processo realizado em relação ao planejado, identificam abordagens técnicas e de gerenciamento importantes para o processo de *software*, e possibilitam a identificação de erros nos produtos de *software*.

É importante destacar o papel da gestão administrativa para a realização das Revisões. Ela deve assegurar tempo, recursos, orçamento, treinamento e suporte apropriado para a realização das Revisões; deve garantir que as Revisões planejadas sejam realizadas; e deve atuar adequadamente sobre as recomendações oriundas das Revisões. A necessidade da realização de Revisões deve ser estabelecida pelos documentos de planejamento de projetos. O autor do produto, a equipe de qualidade, a gestão do projeto ou outro interessado poderá solicitar a realização de Revisões de *Software*.

O padrão IEEE 1028 (1997) categoriza as revisões de *software* em cinco tipos: Auditorias, Inspeções, *Walkthrough*, Revisões Técnicas e Revisões Gerenciais. As principais características de cada um dos tipos de revisão, segundo esse padrão, são especificadas a seguir. Aspectos mais detalhados estão localizados no Apêndice A.

- Auditorias são revisões de *software* que avaliam o processo de *software* e o produto de *software* em relação à conformidade com padrões, planos, diretrizes, legislação e procedimentos regulamentados, sendo caracterizadas pela independência e objetividade.
- As Inspeções são revisões de *software* de caráter formal, com procedimentos e papéis bem definidos para realizar a avaliação de um produto de *software*. Dentre as formalizações utilizadas, uma inspeção deve fazer uso de um *checklist*, e deve ser realizada com a participação de especialistas, treinados no processo de Inspeção, e de um moderador, buscando identificar defeitos (ou anomalias) no produto de *software*.
- Os *Walkthrough*s são revisões de *software* de caráter menos formal que as Inspeções, pois possuem o propósito educativo em relação ao seu público-alvo sobre a necessidade de realizar avaliações nos produtos de *software*.
- Revisões Técnicas de *software* são realizadas para avaliar um produto em relação à sua conformidade com padrões e especificações, e em relação ao uso pretendido do *software*. As avaliações, efetuadas por pessoas qualificadas para determinar a qualidade do produto, podem fazer recomendações ou avaliar alternativas.
- Revisões Gerenciais são reuniões realizadas com a participação de pessoas que possuem responsabilidade gerencial direta sobre um *software*, para monitorar o progresso do trabalho, confirmar os requisitos e o tratamento a ser dado a eles ao longo do projeto, e avaliar a efetividade de abordagens de gerenciamento utilizadas pelo projeto, identificando adequações e inadequações de procedimentos gerenciais.

3.2.1.2. Avaliação (*Assessment*)

Avaliação (*Assessment*) nada mais é do que determinar a capacidade de um processo comparando suas práticas em relação a um padrão de melhores práticas. É uma forma de mensurar um processo, identificando suas forças e fraquezas.

Medir um processo significa obter informações quantitativas sobre ele, coletando, analisando e interpretando dados sobre o processo. Mede-se um processo comumente depois de realizada sua implementação ou após ele ter sofrido alguma mudança. Um processo pode ser mensurado com *base em metodologias*, definindo-se métricas para o processo a partir de metas estabelecidas; ou com *base em paradigmas*, que mensuram o processo a partir de evidências quantitativas dos esforços de melhoria do processo (paradigma analítico), ou a partir da identificação de práticas e ferramentas utilizadas em organizações consideradas em alto nível de excelência ou elevado grau de

maturidade na capacidade do seu processo (paradigma de *benchmarking*). Um *Assessment* é um exemplo de paradigma de *benchmarking* (SWEBOK, 2001).

Organizações podem utilizar um *Assessment* com vários objetivos. Um *Assessment* é uma avaliação realizada por uma equipe treinada de profissionais de *software* com três objetivos: determinar o estado corrente de um processo de *software* de uma organização; determinar as mais altas prioridades das características relacionadas a processos de *software* de organizações; e obter suporte organizacional para melhoria dos processos. Alguns objetivos adicionais podem ser citados como: a identificação do Retorno sobre o Investimento (ROI), a partir da implementação de um novo processo; a identificação de barreiras para a implantação de novos processos, e a necessidade de certificação em um modelo ou padrão de qualidade (IEEE 610.12, 1990; PAULK *et al*., 1993a; STSC-TR, 2000; SWEBOK, 2001).

Existem alguns modelos para realizar processos de *Assessment*, estruturados de acordo com dois tipos de arquiteturas de processos (contínua e por estágios), e que se apresentam como um *framework* de processos. A arquitetura de *Assessment* a ser escolhida é aquela que atende às necessidades e objetivos da organização.

A abordagem contínua é recomendada para as organizações que possuem um conhecimento prévio sobre seus processos de *software*, podendo assim escolher o foco e a seqüência do processo de melhoria. Nessa abordagem, a organização escolhe um dos processos do *framework* por vez, para investir seus esforços. O processo escolhido deve ser visualizado como um subprocesso dentro de um processo em um nível mais macro. São exemplos desse tipo de arquitetura o padrão ISO/IEC 15504 (2003), e o CMMI (*Capability Maturity Model Integration*) (CMMI, 2002a; CMMI, 2002b) em sua abordagem contínua.

A abordagem por estágios é recomendada quando a organização decide trabalhar um conjunto de processos relacionados por vez, começando pelas práticas de gerenciamento, e progressivamente passando de um nível a outro. Neste caso, os modelos de representação por estágio provêem uma solução “pronta” para organizações que buscam a melhoria de processos, onde um nível serve de base para o próximo nível. O CMMI é um exemplo deste tipo de arquitetura.

Os métodos utilizados para fazer um *Assessment* dependem do modelo padrão adotado. O método SCAMPI - *Standard CMMI Appraisal Method for Process Improvement* (Método de Avaliação do padrão CMMI para melhoria de processos) - é o método de *Assessment* utilizado pelo

modelo CMMI. Esse método possui alguns princípios, determinados a partir de experiências com métodos utilizados anteriormente. São eles: patrocínio da alta administração da organização, foco nos objetivos de negócio da organização, confidencialidade para informações obtidas de entrevistados, uso de um método de avaliação documentado, uso de um modelo de referências de processo como base, uso de uma abordagem colaborativa para a equipe, e foco nas ações para melhoria do processo. O SCAMPI pode ser utilizado tanto para identificar melhorias no processo como para avaliar fornecedores de serviços, atendendo aos requisitos de *appraisals* (avaliações) exigidos pelo CMMI. O padrão internacional ISO/IEC 15504 (2003) por si só é considerado como um *framework* para avaliação de processos.

É importante salientar que um *Assessment* possui maior probabilidade de dar certo se houver um comprometimento, tanto do corpo gerencial quanto do corpo técnico da organização, com o esforço de melhoria do processo de *software*.

3.2.1.3. Testes

Testes representam uma das atividades mais importantes para SQA. Sua finalidade básica é identificar a existência de erros nos produtos gerados durante o ciclo de vida de *software*. Testes realizam a análise dinâmica dos produtos de *software*. De acordo com Pressman (2000), testes avaliam a qualidade do produto e servem para melhorá-lo, por meio da identificação de defeitos.

A atividade de testes pode chegar a custos bastante elevados. Cerca de 40% do esforço total de um projeto pode vir a ser representada pelos testes (PRESSMAN, 2000). Em sistemas críticos, que trazem riscos para a vida humana, os custos com testes são extremamente altos. Testes devem descobrir sistematicamente diferentes classes de erros, e devem fazê-lo no menor tempo e com o menor esforço possível. Alcançar estes objetivos diminui sobremaneira os custos envolvidos com a realização de testes.

As principais etapas para a realização de testes são: (*i*) planejar e construir casos de teste para um determinado programa; (*ii*) executar o programa com os casos de testes projetados; (*iii*) analisar o comportamento do programa para determinar se ele produz os resultados esperados para cada caso de testes especificado; (*iv*) o experimento é repetido até que o engenheiro de testes confie que o programa executa conforme esperado. É recomendável que o planejamento para realização dos testes comece nas etapas iniciais do ciclo de vida, já na análise de requisitos, sendo os

procedimentos e planos de testes refinados ao longo do processo de desenvolvimento (SWEBOK, 2001).

Os testes constituem uma verificação dinâmica do comportamento de um programa, com base em um conjunto finito de casos de testes, selecionados dentre inúmeros tipos de execuções possíveis, esperando por determinados tipos de resultados. Desta forma, eles devem ter objetivos, abrangência e graus de precisão muito bem especificados.

Segundo Myers (1979), os principais objetivos dos testes são os seguintes:

- executar um programa buscando identificar erros;
- especificar bons casos de testes, que são aqueles com alta probabilidade de revelar erros não descobertos;
- realizar testes bem sucedidos, onde são revelados erros ainda não descobertos.

Em relação à abrangência, testes podem avaliar um sistema completo, uma parte de um sistema (relacionado a algum fator, como uso ou estrutura), ou um único módulo. Testes não podem demonstrar ausências de erros, mas evidenciam apenas se determinados defeitos de *software* estão presentes.

Em geral, testes podem ser executados em três níveis:

- *Testes de Unidade*: verifica o funcionamento de um módulo do *software*, avaliando-o para identificar erros de lógica ou de implementação. O padrão IEEE 1008 estabelece os conceitos para o teste de unidade e descreve uma abordagem sistemática para esse tipo de teste (IEEE 1008, 1987).
- *Testes de Integração*: verifica a interação entre os componentes do sistema, já testados isoladamente, identificando erros relacionados às interfaces entre esses componentes.
- *Testes de Sistema*: realiza vários testes sobre o sistema, para verificar a adequada integração entre os módulos e o correto funcionamento das funcionalidades do *software*.

Ultimamente, testes vêm sendo tratados como uma abordagem de controle de qualidade, distinta da garantia de qualidade, mas complementar à atuação de SQA. Testes visam identificar defeitos no produto de *software* executável, a fim de que seja verificada a lógica, a execução e a performance do produto de *software* (UNHELKAR, 2003).

O IEEE (2005) define controle de qualidade como sendo “um método iterativo para comparar o produto em relação aos requisitos e tomar as ações se houver uma diferença” e exemplifica que a fase de testes do processo de *software* é parte do processo de controle de qualidade.

3.2.1.4. Técnicas de Análise

As *técnicas de análise* são aplicadas a processos e a produtos com variados objetivos: prevenir a ocorrência de defeitos em *software*, analisar as causas dos defeitos identificados, predizer a confiabilidade do *software*, ou até mesmo prover sistemáticas para controle de processos. Seguem algumas das principais técnicas de análise utilizadas com ênfase em SQA.

- **Prevenção de Defeitos com Análise de Causa Raiz**

Esta técnica tem por objetivo identificar as fraquezas do processo que possibilitaram a ocorrência de defeitos nos produtos, visando prevenir a re-ocorrência de defeitos similares. Pode ser utilizada nos vários processos que fazem parte do ciclo de vida de *software*, utilizada em combinação com a Análise da Causa Raiz do defeito.

A técnica de prevenção de defeitos consiste na reunião de equipes, que podem ser compostas por analistas, desenvolvedores e outros envolvidos com o processo de *software*, com o objetivo de identificar a causa raiz do defeito inserido no produto de *software*. Uma vez identificada a possível causa, ela é classificada. A classificação é composta por duas categorias: sistêmica ou com possibilidade de repetição. A equipe então realiza um *brainstorming* para identificar soluções que minimizem o risco de ocorrência de defeitos similares nas mesmas circunstâncias de ocorrência. As idéias para melhoria do processo são documentadas e enviadas à equipe de gerenciamento do processo.

- **Análise de Predição de Confiabilidade**

A medição da confiabilidade do *software* pode ser feita com base em dados históricos e de desenvolvimento. Eles provêem uma tomada de decisão com maior assertividade sobre o produto de *software* em avaliação. O indicador de tempo médio entre falhas (*Mean Time Between Failure* – MTBF) tem sido utilizado para medir a confiabilidade de *software*. Em Musa *et al.* (1987) podem ser encontrados alguns modelos de confiabilidade de *software* considerando os fatores formais que afetam a confiabilidade, como introdução e remoção de falhas, e o ambiente onde elas ocorrem.

- **Análise de Controle Estatístico do Processo**

Esta técnica utiliza métodos estatísticos para controlar o processo e assegurar a qualidade do processo e do produto. Os métodos estatísticos utilizados são exemplificados por gráficos de controle de *Shewhart*, Análise de Pareto, Histogramas e Diagramas Scatter (SWEBOK, 2004). Os resultados apresentados nesses gráficos são analisados para verificar a estabilidade do processo, identificando defeitos no processo e/ou potencial aumento no número de defeitos do produto. Florac & Carleton (1999) abordaram a utilização de controle estatístico de processos para a engenharia de *software*, com ênfase na melhoria dos processos de *software*.

3.2.2. Modelos e Padrões de SQA

As freqüentes discussões sobre a importância da qualidade sobre o processo de *software* levaram a indústria a se aliar à academia para o desenvolvimento de padrões e modelos, que pudessem trazer a qualidade de *software* desejada para as necessidades do mercado de *software* (PAULK *et. al*., 1993a; PAULK *et. al*., 1993b; ROCHA *et. al*., 2001). Alguns desses modelos trazem em seu escopo propostas para a garantia de qualidade.

3.2.2.1. ISO/IEC 12207

A norma ISO/IEC 12207 (1995) apresenta os processos de *software* de acordo com três tipos: processos fundamentais, processos de suporte e processos organizacionais, conforme apresentado no capítulo 2. Dentre o conjunto de processos de suporte, há cinco processos de ciclo de vida de *software* relacionados com a qualidade:

- processo de Garantia de Qualidade;
- processo de Verificação;
- processo de Validação;
- processo de Revisão Conjunta;
- processo de Auditoria.

Utilizados para gerenciar a qualidade ao longo do ciclo de vida de *software*, esses processos não têm suas atividades apresentadas pela norma. Os quatro últimos processos podem ser empregados separadamente e ainda serem utilizados como técnicas do processo de Garantia de Qualidade (ISO, 1995; ISO, 2002).

3.2.2.2. ISO/IEC 15504

A ISO/IEC 15504 (2003) considera a garantia de qualidade como um processo de suporte cujo objetivo é prover a garantia de que processos e produtos de trabalho de um processo ou projeto estejam em conformidade com seus requisitos especificados e aderentes aos planos estabelecidos. Os resultados de uma implementação bem sucedida desse processo incluem:

- uma estratégia para desenvolver, implementar e manter as atividades de garantia de qualidade;
- ter evidências de que as atividades de garantia de qualidade são realizadas;
- identificar problemas e não conformidades com requisitos contratuais;
- verificar objetivamente a aderência de produtos, processos e atividades aos padrões aplicáveis, requisitos e procedimentos de *software*.

O processo de Garantia de Qualidade, segundo a ISO/IEC 15504 (2003), está relacionado com os demais processos de suporte estabelecidos na norma, como Verificação, Validação, Revisão Conjunta, Auditorias e Resolução de Problemas. As práticas base para esse processo são: (*i*) desenvolver uma estratégia de garantia de qualidade; (*ii*) estabelecer padrões de qualidade; (*iii*) definir registros de qualidade; (*iv*) garantir a qualidade das atividades do processo; (*v*) garantir qualidade dos produtos de trabalho (artefatos); (*vi*) reportar os resultados da qualidade; e (*vii*) lidar com os desvios identificados. Os registros de qualidade são constituídos pelas evidências de que o processo de garantia de qualidade foi realizado.

3.2.2.3. CMMI

O CMMI trata a garantia de qualidade como sendo uma área de processo (*Process Area* - PA) do nível 2 do modelo. Categorizada como processo de suporte, a Garantia de Qualidade de Processo e de Produto (*Process and Product Quality Assurance* - PPQA) possui como propósito prover visibilidade para o *staff* técnico e para o gerenciamento de informações objetivas sobre processos e produtos de trabalho associados (CMMI, 2002a; CMMI, 2002b).

No CMMI, o escopo da atividade de garantia de qualidade pode abranger:

- avaliações de processos, produtos e serviços;
- atividades não vinculadas a projetos, como treinamento; nesse caso, a palavra *projeto* deve ser interpretada apropriadamente.

Como todas as áreas de processo do modelo CMMI-SW, PPQA também possui objetivos genéricos que lidam com a institucionalização de um processo gerenciado e de um processo definido, respectivamente, para os níveis 2 e 3 do modelo. Para institucionalizar um processo gerenciado, é necessário: estabelecer uma política organizacional para planejar e executar a garantia de qualidade de processo e de produto; planejar o processo de garantia de qualidade; prover recursos para execução das atividades de garantia de qualidade; definir responsabilidades e autoridade para executar o processo; treinar pessoas; manter os produtos de trabalho do processo de garantia de qualidade sob gerência de configuração; identificar e envolver *stakeholder*s relevantes no processo de garantia de qualidade; monitorar e controlar o processo em relação aos planos de qualidade; avaliar objetivamente a aderência do processo de garantia de qualidade; e revisar o status das atividades de garantia de qualidade com a gerência de alto nível. Para institucionalizar um processo definido é necessário estabelecer um processo definido e coletar informações de melhoria sobre esse processo.

Os dois objetivos específicos de PPQA são: avaliar objetivamente processos e produtos de trabalho e prover critérios objetivos, envolvendo:

- avaliação objetiva dos processos executados, dos produtos de trabalho e dos serviços em relação às descrições de processo, padrões e procedimentos aplicáveis;
- identificação e documentação de não conformidades, rastreando e comunicando-as objetivamente de forma a garantir que sejam tratadas e resolvidas;
- Prover *feedback* à equipe do projeto e gerentes sobre resultados das atividades de garantia de qualidade.

PPQA busca realizar avaliações objetivas de processos, produtos e serviços de forma que o avaliador seja independente da estrutura avaliada ou do objeto da avaliação e que possua critérios objetivos para realizar a avaliação de qualidade. Os critérios de avaliação abrangem descrições de processo, procedimentos e padrões. Os resultados da avaliação são representados por Relatórios de Avaliação, Relatórios de não Conformidades, e Ações Corretivas.

As etapas do processo de PPQA são as seguintes (CMMI, 2002a):

- **estabelecer planos, processos, padrões e procedimentos**: isto deve acontecer durante o planejamento do projeto, para que sejam adequados às necessidades do projeto e políticas organizacionais;

- **designar processos, produtos de trabalho e serviços que serão avaliados**: as avaliações podem ser feitas por amostragem e devem utilizar critérios objetivos;
- **realizar avaliações de garantia de qualidade**: com base nos planos, processos, padrões e procedimentos definidos para o projeto;
- **tratar não conformidades**: identificadas dentro do projeto, resolvendo nesse âmbito se possível;
- **escalar, para resolução pelo nível adequado de gerenciamento**: não conformidades não tratadas no âmbito do projeto;
- **reportar as atividades de garantia de qualidade e seus resultados**: para todos os envolvidos com o objeto da avaliação (equipes de projeto, gerentes, etc.)

As atividades de garantia de qualidade (QA) acontecem em relação ao processo de *software* da seguinte forma: embutidas no processo de *software* ou realizadas como um processo à parte. Em ambos os casos, algumas ações devem ser tomadas para que o processo seja efetivo. Para o caso das atividades de QA serem embutidas no processo, deve ser tratada a questão da objetividade da avaliação; os avaliadores devem ser treinados em garantia de qualidade; os realizadores de atividades de QA sobre produto de trabalho não devem estar diretamente envolvidos no processo de desenvolvimento ou manutenção desse produto; e deve ser instituído um canal de relato das atividades de QA para o nível gerencial adequado de forma a escalar as não conformidades.

Para o caso das atividades de QA serem realizadas à parte do processo de desenvolvimento, a avaliação deve ser feita por equipe independente dos projetos, dedicada à QA; deve haver treinamento em garantia de qualidade; deve ser instituído um canal de relato das atividades de QA para o nível gerencial adequado de forma a escalar as não conformidades para resolução.

3.2.2.4. Padrões IEEE e SQA

A visão de SQA do IEEE transcrita a seguir baseia-se nos conceitos sobre qualidade de *software*, abordados pelos diversos padrões IEEE relacionados com a qualidade de *software*, segundo o SWEBOK (2004). SQA é definido pelo padrão IEEE 610.12 (IEEE 610.12, 1990) como: “um padrão planejado e sistemático de todas as ações necessárias para prover a garantia adequada de que o produto de trabalho de *software* esteja em conformidade com os requisitos técnicos estabelecidos; um conjunto de atividades projetadas para avaliar o processo pelo qual os produtos de trabalho de *software* foram desenvolvidos e/ou mantidos”.

Existe, nos diversos padrões IEEE, a preocupação com a garantia de qualidade do processo e com a garantia de qualidade do produto. Para realizar SQA do processo é necessário avaliá-lo, utilizando métodos de avaliação previstos pelo IEEE no padrão 1028 (IEEE 1028, 1997), que são as Revisões de *Software*. Para realizar SQA do produto, a ênfase é dada a determinados requisitos para esses produtos (requisitos específicos para *software* executável e requisitos de formato e conteúdo para artefatos do processo).

A partir da necessidade de estabelecer requisitos para produtos de *software* e procedimentos para processos, o IEEE estabeleceu diversos padrões para o Ciclo de Vida de *Software*, os quais servem de guia para a elaboração e a execução de processos de *software* de qualidade, além de estabelecer como devem ser produtos de *software* de qualidade. A Tabela 3.1 apresenta alguns dos principais padrões IEEE relacionados à SQA.

Tabela 3.1 – Padrões IEEE relacionados à SQA

Número do Padrão	Nome do Padrão	Descrição
IEEE Std 1012-1998 e IEEE Std 1012a-1998 (IEEE 1012, 1998 e IEEE 1012a, 1998)	IEEE *Standard of Software Verification and Validation Plans e Supplement to IEEE Standard for Software Verification and Validation*	Requisitos para formato e conteúdo dos Planos de Verificação e Validação.
IEEE Std 1028-1997 (IEEE 1028, 1997)	IEEE *Standard for Software Reviews*	Tipos de Revisão de *Software* e seus procedimentos associados.
IEEE Std 730-2002 (IEEE 730, 2002)	IEEE *Standard for Software Quality Assurance Plans*	Requisitos de formato e conteúdo para Plano de SQA, em conformidade com a ISO/IEC 12207-1995.
IEEE Std 1061-1998 (IEEE 1061, 1998)	IEEE *Standard for a Software Quality Metrics Methodology*	Estabelece requisitos de qualidade e identifica e valida métricas de qualidade de processo e produto.
IEEE Std 1044-1993 (IEEE 1044, 1993)	IEEE *Standard Classification for Software Anomalies*	Abordagem para classificar anomalias encontradas no *software* e em sua documentação.

Os padrões acima são brevemente descritos a seguir:

- IEEE *Standard of Software Verification and Validation Plans*: o processo de Verificação aborda a utilização do processo correto para produzir o *software*, enquanto que o processo de Validação avalia se o produto de *software* é correto para o que se propõe. Esse padrão provê os requisitos mínimos e uniformes para o formato e o conteúdo dos Planos de Verificação e Validação. Seu suplemento mapeia os requisitos do padrão aos requisitos de outros padrões IEEE (IEEE 1012 e IEEE 1012a, 1998).
- IEEE *Standard for Software Reviews*: define os tipos de revisão de *software* e os procedimentos requeridos para execução de cada tipo de revisão. Não aborda a etapa de

determinar a necessidade de revisão. As revisões devem ser realizadas de forma sistemática e são aplicáveis aos diversos processos do ciclo de vida de *software*: aquisição, fornecimento, desenvolvimento, operação e manutenção (IEEE 1028, 1997).

- IEEE *Standard for Software Quality Assurance Plans*: provê requisitos para a preparação, o conteúdo e a aprovação do Plano de Garantia de Qualidade, ou Plano de SQA, especificando seu formato e conteúdo, em conformidade com os requisitos da ISO/IEC 12207 (IEEE 730, 2002). Define a terminologia utilizada e a estrutura e conteúdo de cada seção do Plano de SQA, assim como a aplicabilidade do padrão. O anexo deste padrão mostra um mapeamento entre ele e a norma ISO/IEC 12207, a partir dos requisitos relacionados à garantia de qualidade de *software*.
- IEEE *Standard for a Software Quality Metrics Methodology*: aborda uma metodologia para estabelecer requisitos de qualidade e identificar, implementar, analisar e validar a definição de métricas de qualidade para o processo e o produto de *software* (IEEE 1061, 1998). Define qualidade de *software* como o grau no qual o *software* deve atender a uma determinada combinação de atributos, estabelecendo que métricas de *software* devem ser coletadas ao longo de todo o ciclo de vida, buscando minimizar a subjetividade envolvida.

3.3. Processos de SQA

Unhelkar (2003) apresenta o processo de garantia de qualidade como sendo realizado a partir de padrões, *templates*, artefatos, atividades e tarefas que enfocam a qualidade. Ele reivindica o uso de modelagem do que denomina de: *espaço problema* (onde são analisados os problemas e especificados os requisitos de solução); *espaço solução* (que envolve a descrição da solução); e *background* (ou arquitetura de apoio e gerenciamento do trabalho do projeto). A proposta do autor é voltada para projetos baseados em UML (*Unified Modeling Language*), onde a modelagem utiliza diagramas UML para melhor compreender e especificar todas as variáveis envolvidas em cada um desses espaços.

As atividades de SQA de Unhelkar (2003) visam checar atividades, tarefas, artefatos e modelos UML, além das conformidades aos padrões de qualidade, identificando técnicas adequadas de SQA e utilizando *checklist*s para garantir a qualidade. Erros devem ser identificados e reportados e avaliações devem ser realizadas após os erros reportados terem sido corrigidos.

Marckzak *et al.* (2003) utiliza um modelo de organização da função de qualidade de *software*, considerando aspectos práticos da implantação da garantia de qualidade em uma organização de desenvolvimento de *software*, no contexto de implantação do CMM-SW. A proposta refere-se a um conjunto de atividades realizadas pela equipe de SQA para apoiar os projetos de *software*. São elas:

- Apoiar o uso e compreensão dos processos, procedimentos e *templates*;
- Apoiar o planejamento do projeto;
- Planejar as atividades de qualidade para o projeto;
- Revisar as atividades do projeto;
- Auditar os artefatos do projeto;
- Documentar e reportar os desvios encontrados;
- Acompanhar a resolução dos desvios encontrados;
- Coletar as métricas de qualidade definidas para o projeto;
- Revisar as atividades de qualidade do projeto com os gerentes (de projeto e organizacional);
- Revisar as atividades de qualidade do projeto com o SQA do cliente;
- Revisar as atividades e artefatos que fazem parte do processo de SQA com um especialista.

Cada uma dessas atividades utiliza artefatos de entrada e produzem artefatos de saída, e são baseadas na área chave de processo SQA do modelo CMM-SW, no nível 2.

Drabick (2000) propõe um modelo de processo para SQA/SQE (*Software Quality Engineering*) genérico, modelado graficamente utilizando técnicas de entrada-processo-saída (*input-process-output* ou IPO). O processo seria composto por oito sub-processos: revisão dos planos nos níveis de projeto e de programa; desenvolvimento do Plano de SQA; coordenação de métricas; coordenação do programa de riscos; execução de auditorias; coordenação de reuniões de revisão; facilitação na melhoria do processo e monitoração do programa de testes. Além de aspectos clássicos de SQA, o processo abrange ainda o programa de riscos, o programa de testes e a melhoria de processo e seria aplicável a projetos de desenvolvimento grandes ou médios.

3.4. Conclusão

Este capítulo apresentou um panorama geral sobre SQA e sua importância para o processo de desenvolvimento de *software*, destacando modelos de atuação, técnicas empregadas, perfil adequado para realizar atividades de SQA e a importância dos padrões e métricas para a garantia de qualidade.

O próximo capítulo apresenta o Processo Unificado da *Rational*/IBM (RUP) e a visão de como os conceitos de qualidade abordados nos capítulos 2 e 3 deste trabalho são tratados no *Rational Unified Process*.

Capítulo 2

QUALIDADE DO PROCESSO DE *SOFTWARE* & GERENCIAMENTO DA QUALIDADE

Este capítulo apresenta conceitos de qualidade de software e a importância da qualidade do processo de software para o processo de desenvolvimento. Além disso, trata do processo de qualidade de software e das principais abordagens sobre o Gerenciamento da Qualidade.

Software é um fator diferencial para a sociedade, sejam pelas informações que fornece, sejam pelas oportunidades que proporciona, fornecendo suporte aos negócios e projetos da sociedade. Devido a esta importância, não houve apenas o crescimento da demanda por *software*, mas também por *software* de qualidade. Nota-se, entretanto, ainda uma certa permissividade com a qualidade por parte dos envolvidos com o processo de desenvolvimento de *software* e com o gerenciamento desse processo.

2.1 Qualidade de *Software*

Preocupada em prover soluções práticas para o desenvolvimento e a manutenção de sistemas de *software*, a engenharia de *software* emergiu como ciência visando, a partir do estabelecimento de princípios de engenharia, minimizar custos e entregar um produto final de qualidade. O esforço despendido pela engenharia de *software* (por meio de técnicas, procedimentos, métodos e ferramentas) possui um único objetivo: produzir *software* com qualidade (PRESSMAN, 2000).

Mas o que é Qualidade? Os precursores na definição desse conceito foram Deming, Crosby e Juran. Para Crosby (1979), qualidade é a conformidade aos requisitos. Já para Deming (1982), qualidade é definida pelo cliente, enquanto que Juran (1988) identifica qualidade como sendo adequação ao uso. Recentemente, alguns autores também conceituaram a qualidade. Basili *et al*. (1991) vêem a qualidade como uma característica intrínseca e multifacetada de um produto. Kitchenham *et al*. (1996) afirmam que qualidade é um conceito complexo, porque possui significados diversos para diferentes pessoas, dependendo do

contexto. Já Belchior (1997) declarou que "... não se pode pensar em qualidade como sinônimo de perfeição. Trata-se de algo factível, relativo, substancialmente dinâmico e evolutivo, amoldando-se à granularidade dos objetivos a serem atingidos".

O conceito de qualidade foi incorporado à área de *software*. Para o IEEE (*International Electronical and Electric Engineering*), qualidade de *software* é "o grau no qual o *software* atende a uma determinada combinação de atributos" (IEEE 610.12, 1990). A ISO (*International Standardization Organization*) conceitua qualidade de *software* como sendo "a totalidade de características que influenciam a habilidade de satisfazer necessidades explícitas ou implícitas para o desenvolvimento de um *software*" (ISO, 1997). Pressman (2000) trata a qualidade de *software* como sendo a conformidade com requisitos de performance e requisitos funcionais claramente especificados, com padrões de desenvolvimento documentados e com características implícitas esperadas pelo usuário.

Para o contexto deste trabalho, o conceito de Qualidade de *Software* adotado é mais próximo do enunciado pela ISO, onde a qualidade de *software* é expressa pelo grau no qual o *software* atende requisitos implícitos e explícitos do cliente.

O esforço de melhoria da qualidade de *software* numa organização depende da definição das características de qualidade pertinentes e na decisão sobre como a qualidade será mensurada. É responsabilidade do engenheiro de *software* definir os requisitos para a qualidade do *software*, preocupando-se com a definição do propósito do *software* (porque a necessidade do cliente deve vir em primeiro lugar), e com as funcionalidades que o *software* deve implementar (SWEBOK, 2001). Neste processo de elicitação de requisitos da qualidade, deve-se buscar identificar fatores explícitos e implícitos, que influenciam e determinam a qualidade do *software* de acordo com a perspectiva do usuário. Depois de definido o significado de qualidade de *software* para a organização, esta deverá determinar como deverá ser realizado seu processo de qualidade de *software*.

A qualidade de *software* é responsabilidade de todos os envolvidos direta e indiretamente com o processo de *software* da organização, e possui impacto frontal na qualidade do produto gerado (IEEE, 2005). Cada profissional contribui para a qualidade existente no processo e para a qualidade do produto final. A qualidade de um produto de *software* é diretamente proporcional à qualidade profissional das pessoas que intervêm no seu processo de desenvolvimento (CONRADI & FUGGETTA, 2002).

Os gerentes devem estar cônscios da importância da qualidade para se comprometerem com suas atividades e seus resultados. Os atores do processo, como gerentes de projeto, analistas, projetistas e desenvolvedores, devem contribuir para a qualidade em cada ação realizada. Também é importante o papel do cliente e/ou usuário, definindo os requisitos de qualidade para o *software*, e validando suas especificações em relação ao produto, o que ocorre com mais freqüência em processos de ciclo de vida iterativo (KROLL & KRUCHTEN, 2003; UNHELKAR, 2003).

A qualidade de *software* é importante para o processo de desenvolvimento, traduzindo-se na preocupação com a qualidade do processo de *software*. A obtenção de qualidade resulta do esforço de todos os envolvidos com o processo de *software* e requer comprometimento gerencial para que os objetivos de qualidade sejam atingidos (SCHULMEYER & MCMANUS, 1999).

2.2 Qualidade do Processo de *Software*

A qualidade do processo de *software* relaciona-se fortemente à qualidade do produto de *software* (FUGGETTA, 2000; SOMMERVILLE, 2003). A preocupação com a qualidade do processo de *software* visa verificar se o processo de *software* é capaz de produzir determinado produto, e identificar as falhas do processo antes que impactem sobre o produto.

As atividades de qualidade do processo de *software* compreendem a definição do processo, a verificação da utilização do processo definido, e a preocupação com a melhoria do processo utilizado. Primeiramente, o processo de *software* da organização deve ser definido, atendendo aos objetivos da organização. No CMMI, por exemplo, este trabalho pode ser conduzido pela equipe de processo de engenharia de *software* (*Software Engineering Process Group* - SEPG), com a participação da equipe de garantia de qualidade (*Quality Assurance* – QA) (CMMI, 2002a; CMMI, 2002b). Enquanto o SEPG é responsável pela definição do processo, o SQA verifica a aderência das práticas de engenharia de *software* dos projetos ao processo definido.

São utilizadas técnicas de garantia de qualidade de *software* (*Software Quality Assurance* - SQA) para verificar se o processo definido está sendo seguido, para encontrar defeitos diretamente no produto, ou para indicar a necessidade de um exame mais detalhado no produto. As principais técnicas utilizadas são as Revisões de *Software*, classificadas como Auditorias, Inspeções, *Walkthrough*s, Revisões Técnicas e Revisões Gerenciais (IEEE 1028,

1997). As Revisões de *Software* avaliam o processo de desenvolvimento, seu gerenciamento, e o produto de *software* (SWEBOK, 2004; IEEE 1028, 1997). Além das Revisões, são também realizados Testes, que constituem as ações denominadas de Controle de Qualidade (UNHELKAR, 2003) e representam a última revisão das especificações de projeto e código do produto de *software* (PRESSMAN, 2000). Quando o foco é a melhoria do processo estabelecido, são utilizadas as Avaliações (*Assessment*s), que examinam um processo para determinar a sua capacidade, comparando-o a um padrão de melhores práticas (PAULK *et al*., 1993a; PAULK *et al*., 1993b; DUNAWAY & MASTERS, 1996; SWEBOK, 2004).

Alguns modelos e normas internacionais tratam a qualidade do processo de *software* e trazem, em seu escopo, a figura da função de qualidade. O CMMI é um modelo aplicado para determinar a capacidade e maturidade dos processos de uma organização que desenvolve *software* (CMMI, 2002a; CMMI, 2002b). A norma ISO 9001 (2000) trata da definição, implementação e manutenção do Sistema de Gestão da Qualidade para uma organização (ISO, 2000). A ISO/IEC 15504 constitui-se de um *framework* para modelos de avaliação de processos de *software*, podendo ser utilizada também em estratégias para melhoria de processos (ISO, 2003).

2.3. Processo de Qualidade de *Software*

Desenvolver *software* é uma tarefa complexa, expressa por meio do estabelecimento de um processo de *software*. Segundo Humphrey (1995), um processo de *software* é uma seqüência de estágios para desenvolver *software*, utilizando métodos e ferramentas e incluindo pessoas nas tarefas de construir o sistema. Um processo de *software* visa dotar uma organização, de forma customizada às suas características, com um passo a passo para realizar suas tarefas de desenvolvimento ou manutenção de *software*, especificando, no mínimo, papéis, atividades, tarefas e artefatos.

Unhelkar (2003) afirma que mesmo uma organização estabelecendo e seguindo um processo de *software*, não significa necessariamente que ela tenha um processo com qualidade. O autor sustenta a necessidade de se possuir um esforço de qualidade, representado por um processo de qualidade, que seja distinto do processo de *software* da organização. Apesar de serem dois processos distintos, processo de *software* e processo de qualidade atuam em conjunto para proverem a organização com a estrutura necessária para desenvolver e manter *software* de qualidade.

Não basta adotar um processo de *software*, para que se tenha um processo de qualidade. Um processo, embora definido e executado, precisa ser constantemente melhorado a fim de amoldar-se às necessidades apresentadas pela organização ao longo do tempo (SWEBOK, 2004). Apesar de haver crescido o número de técnicas disponíveis para desenvolver sistemas, e de haver diversos tipos de processos de *software* disponíveis, as unidades responsáveis por desenvolver *software* continuam a fazê-lo de forma arcaica, estourando prazos e custos e sacrificando o processo estabelecido. Se o processo de *software* não é seguido, não é possível verificar se ele é adequado às necessidades da organização, nem como ele pode ser melhorado para atendê-la. A qualidade do processo de *software* resulta do esforço de todos os envolvidos nesse processo. O processo de qualidade auxilia a execução do processo de *software*, orientando sobre como melhor utilizá-lo, verificando seus pontos de dificuldade e auxiliando todos os atores do processo a efetivamente produzir *software* com qualidade.

A importância do processo de qualidade é traduzida pela afirmação de Unhelkar (2003): “um processo de qualidade não somente ajuda a produzir *software* que satisfaça as necessidades do usuário final; ele também ajuda a melhorar a produtividade, reduzindo os custos com erros”. Usualmente, a preocupação maior tem sido com o produto de *software* que está sendo construído, especialmente no final do processo de desenvolvimento, quando são realizados os testes de *software* para identificar defeitos. Sabe-se, entretanto, que o custo de corrigir um problema identificado ao final do processo de desenvolvimento é cerca de 20 a 100 vezes mais caro do que se ele houvesse sido identificado nas primeiras etapas do processo de *software* (THE CHAOS STUDY, 1994; UNHELKAR, 2003).

Um processo de qualidade de *software* atuando ao longo de todo o processo de desenvolvimento pode auxiliar a:

- identificar e corrigir problemas mais cedo, reduzindo os custos de correção de erros;
- desenvolver um produto que atenda as necessidades dos usuários;
- utilizar boas práticas do processo de *software* de forma corporativa;
- melhorar continuamente o processo de desenvolvimento de *software*.

A estruturação de um processo de qualidade passa pela importante etapa da disseminação de uma cultura de qualidade (SOMMERVILLE, 2003; SCHULMEYER & MCMANUS, 1999). Os conceitos relacionados com a qualidade de *software* também devem ser disseminados. A seguir descreveremos os principais conceitos utilizados neste trabalho:

- *Padrão*: representa um requisito mandatário, empregado para prescrever uma abordagem uniforme e disciplinada para o desenvolvimento de *software* em termos de práticas e convenções (IEEE 610-12, 1990).
- *Métrica*: propriedade utilizada como unidade de medição. Por exemplo, número de linhas de código.
- *Objetivos da Qualidade*: proporcionam um foco na direção da organização, em conjunto com as Políticas de Qualidade, para determinar os resultados desejados e auxiliar a organização no uso de seus recursos para alcançar esses resultados. (ISO, 2000).
- *Não Conformidade*: constitui o reconhecimento da falta de aderência de determinado item sob garantia de qualidade em relação aos padrões de qualidade (CMMI, 2002a; CMMI, 2002b).
- *Defeito*: não atendimento a um requisito relacionado a um uso pretendido ou especificado (ISO, 2000). O termo defeito pode ser utilizado como sinônimo para falha ou anomalia no *software*.
- *Desvio*: qualquer não conformidade que não possa ser resolvida no projeto de *software*, ou porque os padrões estão inadequados às necessidades do projeto ou porque são assumidas como uma exceção pelo nível de gerenciamento. Devem ser registradas e analisadas como indicadores de qualidade (CMMI, 2002a; CMMI, 2002b).
- *Ação preventiva*: ação para eliminar a causa de uma não conformidade em potencial ou outra situação potencialmente indesejável (ISO, 2000).
- *Ação corretiva*: ação para eliminar a causa de uma não conformidade identificada ou outra situação indesejável (ISO, 2000).

Unhelkar (2003) propôs um processo de qualidade de *software* para projetos baseados em UML (*Unified Modeling Language*). O autor entende o processo de qualidade de *software* somo sendo uma união entre o processo de desenvolvimento de *software* e o processo de qualidade, conforme a Figura 2.1.

Segundo Unhelkar (2003), a qualidade de *software* deve ser organizada e planejada similarmente à organização de gerenciamento de projetos. Utilizando os conceitos da gestão de projetos, a gestão da qualidade seria tratada como uma função distinta, com suas atividades, tarefas, papéis, e artefatos.

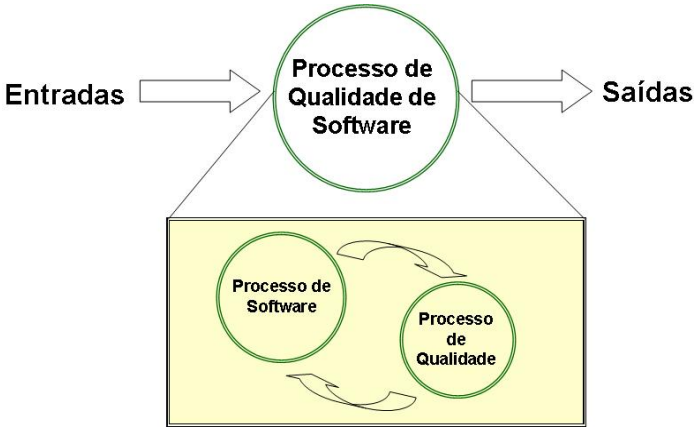

Figura 2.1 – Processo de Qualidade de *Software* (UNHELKAR, 2003)

O processo de qualidade de *software* é formado por três componentes:

- *Gerenciamento da Qualidade*: objetiva direcionar aspectos sociais (pessoas e perfis) e aspectos metodológicos (atividades, prazos, recursos) do projeto enfocando a qualidade. Deve identificar os padrões de qualidade aplicáveis ao projeto, obter e estabelecer as expectativas dos usuários e obter os recursos (capital humano) adequados para trabalhar com qualidade. Dentre as atividades estão o planejamento da qualidade, o estabelecimento de padrões e o planejamento dos testes, além da criação de um ambiente de qualidade (definição da estrutura organizacional, da equipe de qualidade, e da forma de relacionamento da equipe de qualidade com os membros da equipe dos projetos).
- *Garantia da Qualidade*: objetiva garantir a qualidade do processo e dos modelos (UML) utilizados no projeto, em relação aos padrões de qualidade estabelecidos.
- *Controle da Qualidade*: verifica e valida o resultado dos esforços de seguir o processo de *software* e de utilizar os modelos UML – o produto de *software*. Abrange os testes de *software*, visando checar a lógica, a execução e o desempenho do produto de *software*.

Outros aspectos importantes do processo de qualidade abordados por Unhelkar (2003) dizem respeito à independência da *função de qualidade* para realização das atividades de qualidade. Segundo ele, o Gerente de Qualidade deve possuir acesso independente ao Comitê de Decisões da Organização e aos Gerentes responsáveis pela tomada de decisão que influenciam os projetos, provendo a eles informações sobre a qualidade dos projetos. A

independência da função de qualidade somente é conseguida se ela estiver associada e for exercida por alguém que esteja fora da estrutura do projeto.

2.4. Gerenciamento da Qualidade

O *gerenciamento da qualidade* pode ser definido como o conjunto de ações que visam coordenar os esforços de qualidade, para atender as necessidades dos clientes em termos de produtos e serviços de qualidade (ISO, 2000). O gerenciamento da qualidade coordena as atividades dos processos de *garantia de qualidade* e *controle de qualidade*.

Segundo o IEEE (2005), há uma distinção entre os conceitos de garantia de qualidade e controle de qualidade. O controle de qualidade pode ser definido como sendo “um método iterativo para comparar o produto em relação aos requisitos e tomar as ações se houver uma diferença”. A garantia de qualidade consiste em “prover evidências e confidencialidade da habilidade do processo de *software* em produzir um produto que atenda seus requisitos especificados”, prestando suporte ao controle de qualidade. Exemplificando, a fase de Testes do processo de *software* faz parte do processo de controle de qualidade, enquanto que a garantia de qualidade atua no processo de testes, verificando a aderência das práticas de testes ao processo de testes documentado.

Sommerville (2003) acredita que o gerenciamento da qualidade “envolva a definição de procedimentos e padrões que devam ser utilizados durante o desenvolvimento de *software* e a verificação de que eles estão sendo seguidos por todos os envolvidos com o processo de desenvolvimento”. O próprio autor afirma ainda a necessidade de:

- ser criada uma cultura de qualidade na organização;
- ter os procedimentos (passo a passo do processo) e os padrões (requisitos de qualidade para o produto) como base para o gerenciamento da qualidade;
- encorajar as equipes de projeto a desenvolverem melhores práticas para os aspectos de qualidade que não podem ser especificados como padrões.

Sommerville (2003) apresenta o gerenciamento da qualidade como sendo composto por:

- *Garantia de Qualidade*: estabelecimento de uma estrutura de padrões e procedimentos que direciona a produção do *software* com qualidade.

- *Planejamento da Qualidade*: seleção dos padrões e procedimentos adequados e sua adaptação para o projeto, baseado na estrutura de padrões e procedimentos definida para a organização.
- *Controle de Qualidade*: definição e aprovação de processos que assegurem que os padrões e procedimentos sejam seguidos pelas equipes dos projetos.

Para o autor, o gerenciamento da qualidade fornece uma verificação independente sobre o processo de desenvolvimento de *software*, ou seja, a equipe de qualidade, sendo independente do processo de desenvolvimento de *software*, identifica e relata problemas e dificuldades para a gerência sênior da organização por meio de uma visão objetiva do processo.

A norma ISO/IEC 12207 (1995, 2002), que trata dos processos de ciclo de vida de *software*, incluindo os processos relacionados com qualidade, estabelece uma estrutura comum de processos para ser utilizada como referência em negócios relacionados a produtos e serviços de *software*. Os processos de Ciclo de Vida dessa norma estão agrupados em classes de processos:

- *Fundamentais*: constituídos por processos que atendem as partes fundamentais (pessoa ou organização) durante o ciclo de vida de *software*.
- *Apoio*: constituídos por processos que auxiliam a execução de um outro processo, como se dele fizessem parte, mas com objetivo distinto, contribuindo para que o projeto tenha qualidade e seja bem sucedido.
- *Organizacionais*: constituídos por processos empregados para estabelecer e implementar uma estrutura de ciclo de vida e de pessoas, possibilitando a melhoria contínua da estrutura e dos processos.

Na categoria de processos organizacionais, há o processo de gerenciamento da qualidade, cujo propósito é atingir a satisfação do cliente, monitorando a qualidade dos produtos e serviços, nos níveis organizacional e de projeto. A ISO/IEC 12207 (2002) não especifica como implementar ou executar as atividades e tarefas, devendo ser adaptada de acordo com a organização e projetos específicos.

O padrão ISO/IEC 15504 (2003) é utilizado para avaliar processos, assessoria de treinamentos e competências, e para determinar a capacidade e a melhoria de processo. Ele organiza e classifica as práticas de engenharia de *software* nas dimensões de: processo, por meio de suas categorias; e de capacidade, por meio dos níveis do modelo. Esse padrão

apresenta o processo de gerenciamento da qualidade em termos de: objetivos, resultados esperados de sua implementação e práticas-base para execução.

O processo de gerenciamento da qualidade é classificado pela ISO/IEC 15504 como um processo gerencial e visa monitorar a qualidade de produtos e/ou serviços do projeto, e assegurar que eles satisfaçam os clientes. Esse processo envolve estabelecer foco na monitoração da qualidade do processo e do produto, tanto no nível do projeto como no nível organizacional. Neste caso, os resultados de uma implementação bem sucedida do processo de gerenciamento da qualidade envolvem:

- objetivos de qualidade definidos com base nos requisitos implícitos e explícitos do cliente e estabelecidos de acordo com os vários *checkpoints* ao longo do ciclo de vida do projeto;
- desenvolvimento de uma estratégia geral de qualidade para atingir as metas definidas;
- identificação das atividades de controle de qualidade e de garantia de qualidade que serão executadas;
- monitoração do desempenho atual em relação às metas de qualidade;
- tomada de ações apropriadas, quando as metas de qualidade não forem atingidas.

As práticas base para execução do processo de gerenciamento de qualidade, segundo a ISO/IEC 15504 (2003) são: estabelecer objetivos de qualidade, definir a estratégia geral, identificar atividades de qualidade, executar atividades de qualidade, avaliar a qualidade (checar se os objetivos foram atingidos), e tomar ações corretivas.

A ISO 9001 (ISO, 2000) apresenta a qualidade como sendo o grau no qual um conjunto de características inerentes satisfaz a requisitos. O conceito de qualidade está ligado à satisfação do cliente, que corresponde à percepção de que o cliente possui do grau em que seus requisitos foram atendidos. Segundo esta norma, a gestão da qualidade é o conjunto de atividades coordenadas para dirigir e controlar uma organização no que diz respeito à qualidade. A gestão da qualidade é composta por quatro partes:

- *Planejamento da Qualidade*: estabelece os objetivos da qualidade, e especifica os recursos e processos operacionais necessários para atender a esses objetivos.
- *Controle da Qualidade*: visa o atendimento dos requisitos da qualidade.

- *Garantia da Qualidade*: objetiva prover confiança de que os requisitos da qualidade serão atendidos.
- *Melhoria da Qualidade*: tem como foco o aumento da capacidade de atender os requisitos da qualidade.

A garantia da qualidade e a melhoria da qualidade estão ligadas à eficácia do processo: atividades planejadas são realizadas e os resultados planejados alcançados. A melhoria da qualidade também está ligada à eficiência do processo, sendo uma relação entre os resultados alcançados e os recursos utilizados (ISO, 2000).

A Tabela 2.1 mostra um resumo dos conceitos relacionados aos componentes do processo de *software*, de acordo com os padrões ISO/IEC 12207 (2002), ISO/IEC 15504 (2003), ISO 9001 (2000) e IEEE (2005) sobre qualidade de *software*.

Tabela 2.1 – Componentes do processo de qualidade de *software*

	ISO/IEC 12207 (2002)	**ISO/IEC 15504 (2003)**	**ISO 9001 (2000)**	**IEEE (2005)**
Gerencia-mento da Qualidade	• Satisfação do cliente; • Monitorar qualidade (nível organizacional e de projeto); • Processo gerencial.	• Processo gerencial; • Definição de objetivos de qualidade; • Ações corretivas visando alcançar objetivos de qualidade	• Definir objetivos de qualidade; • Monitorar o atingimento dos objetivos; • Qualidade como compromisso gerencial; • Melhoria do processo.	Não encontrado
Garantia de Qualidade	• Processo de suporte ao processo de desenvolvimento.	• Garantia da qualidade do processo e do produto; • Evidências objetivas; • Ações corretivas.	• Confidencialidade do processo para produzir produto atendendo requisitos.	• "Prover evidência e confidencialidade da habilidade do processo de *software* em produzir um produto que atenda seus requisitos especificados"
Controle de Qualidade	• Atividade do processo fundamental de desenvolvimento de *software* (testes).	Não especificado	• Produto atendendo requisitos.	• Comparar o produto em relação aos requisitos e tomar as ações corretivas.

2.5. Conclusão

O gerenciamento da qualidade é visto como processo gerencial da qualidade, com abrangência e atuação nos níveis organizacional e de projeto. Traduz-se como sendo o conjunto de ações para promover e estabelecer a qualidade no processo de desenvolvimento de *software*, considerando os objetivos de qualidade da organização e sua definição de qualidade de *software*.

O gerenciamento da qualidade gerencia a execução das ações de qualidade executadas por meio dos processos de garantia de qualidade e controle de qualidade. Como o escopo deste trabalho abrange a qualidade de processo de *software*, não serão detalhados aspectos sobre controle de qualidade, que visam a qualidade do produto.

O próximo capítulo tratará da garantia de qualidade de *software* e de sua importância para a qualidade de processo de *software*.

REFERÊNCIAS BIBLIOGRÁFICAS

Ambler, 2002	Ambler, Scott. *The Enterprise Unified Process*. 2002. Disponível em http://www.enterpriseunifiedprocess.info/ Acessado em julho de 2004.
April *et al.*, 1998	April, Alan. Abran Alain & Merlo, Etore. *Process Assurance Audits: Lessons Learned*. 1998. Paper disponível na Biblioteca Digital da ACM – *Association Computational Machinery*.
Arthur, 1993	Arthur, L.J. *Improving Software Quality: an insider's guide to TQM*. John Wiley & Sons, 1993.
Bach, 1997	Bach, James. *Good Enough Quality: beyond the buzzword*. IEEE Computer, August, 1997.
Baker, 2001	Baker, E. R. *Which Way, SQA?* IEEE *Software* January/February 2001.
Basili *et al.*, 1991	Basili, V. R., Musa, J. D. *The future engineering of software: A management perspective*. IEEE Computer, September. 1991.
Belchior, 1997	BELCHIOR, A. D. *Um modelo fuzzy para avaliação da qualidade de software*. Tese de Doutorado, UFRJ-COPPE, Maio, Rio de Janeiro, 1997.
Benbasat *et al.*, 1987	Benbasat, Izak; Goldstein, David & Mead, Melissa. *The Case Research Strategy in Studies of Information Systems*. MIS Quaterly / September 1987.
Bonoma, 1985	Bonoma, T. V. *Case Study in Marketing: opportunities, problems and a process*. Journal of a Marketing Research. Vol 22. May, 1985.
Booch, 1995	Booch, Grady. *Object Solutions - Managing the Object-Oriented Project.* Addison Wesley Longman, 1995.
Booch *et al.*, 1998	Booch, Grady. Jacobson, Ivar and Rumbaugh, James. *Unified Modeling Language* 1.3, White paper. *Rational Software* Corp., 1998.
CMMI, 2002[a]	CMMI Product Team. *CMMI for Systems Engineering/Software Engineering*. Version 1.1 Staged Representation (CMU/SEI-2002-TR-029, ESC-TR-2002-029). Pittsburgh, PA: *Software* Engineering Institute, Carnegie Mellon University, August, 2002.
CMMI, 2002b	CMMI Product Team. *CMMI for Systems Engineering/Software Engineering*. Version 1.1 Continuous Representation (CMU/SEI-2002-TR-028, ESC-TR-2002-028). Pittsburgh, PA: *Software* Engineering Institute, Carnegie Mellon University, August, 2002.
Conradi & Fuggetta, 2002	Conradi, R. and Fuggetta, A. *Improving Software Process Improvement.* IEEE *Software* July/August 2002.

Craft, 2001	Craft, Terrence W. *SQA in Review*. In: SEPG Conference, 2001.
Crosby, 1979	Crosby, P. *Quality is Free*. New York, New York: McGraw-Hill, 1979.
Drabick, 2000	Drabick, Roger. *A Process Model of Software Quality Assurance/Software Quality Engineering*. SQP, Vol 2, No. 4, ASQ, 2000.
Deming, 1982	Deming, W. *Quality, productivity and competitive position*. MIT Press, USA, 1982.
Duarte & Falbo, 2000	Duarte, C. and Falbo, R. A. *Uma ontologia de qualidade de software. Workshop de Qualidade de Software.* João Pessoa, p. 275-285, Outubro de 2000.
Dunaway & Masters, 1996	Dunaway, D. & Masters, S. *CMM-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description* (CMU/SEI-96-TR-007). Pittsburgh, PA: *Software* Engineering Institute, Carnegie Mellon University, April 1996.
Feldman, 2005	Feldman, Stuart. *Quality Assurance: much more than testing.* ACM Queue. February, 2005.
Florac & Carleton, 1999	Florac, W & Carleton, A. *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. Addison Wesley, 1999.
Fuggetta, 2000	Fuggeta, A. *Software Process: a roadmap*. In: The future of *Software* Engineering. A Fulkenstein (ed). 2000.
Grady, 1992	Grady, Robert. *Practical Software Metrics for Project Management and Process Improvement*. Prentice-Hall, 1992
Harter & Slaughter, 2000	Harter, Donald E. & Slaughter, Sandra A. *Process Maturity and Software Quality: a field study*. In: Proceedings of the 21th International Conference on Information Systems. p 407-411. Australia, 2000.
Humphrey, 1995	Humphrey, W. S. *A discipline for software engineering*. Reading, USA: Addison-Wesley Publishing Company, 1995. 816p.
IEEE 1008, 1987	IEEE Std 1008-1987 (R2003), *IEEE Standard for Software Unit Testing*: IEEE, 1987.
IEEE 610.12, 1990	IEEE Std 610.12-1990 (R2002), *IEEE Standard Glossary of Software Engineering Terminology*: IEEE, 1990.
IEEE 1044, 1993	IEEE Std 1044–1993 (R2002), *IEEE Standard Classification for Software Anomalies*: IEEE, 1993.
IEEE 1028, 1997	IEEE Std 1028–1997 (R2002), *IEEE Standard for Software Reviews*: IEEE, 1997.

IEEE 1074, 1997	IEEE Std 1074-1997, *IEEE Standard for Developing Software Life Cycle Process*: IEEE, 1997.
IEEE 1012, 1998	IEEE Std 1012–1998, *IEEE Standard of Software Verification and Validation Plans*: IEEE, 1998.
IEEE 1012a, 1998	IEEE Std 1012a–1998, *Supplement to IEEE Standard for Software Verification and Validation*: IEEE, 1998.
IEEE 1061, 1998	IEEE Std 1061–1998, *IEEE Standard for a Software Quality Metrics Methodology*: IEEE, 1998.
IEEE 730, 2002	IEEE Std 730-2002, *IEEE Standard for Software Quality Assurance Plans*: IEEE, 2002.
IEEE, 2005	*IEEE Computer Society resource on practical Software Engineering (SE) knowledge*. Disponível em: http://www.computer.org/portal/site/seportal/ . Acessado em janeiro de 2005.
ISO, 1995	ISO/IEC International Organization for Standardization/International Electrotechnical Comission 12207:1995 - Information Technology – *Software Life Cycle Processes* – Amendment I. ISO/IEC, 1995.
ISO, 1997	Associação Brasileira de Normas Técnicas (ABNT), NBR-ISO 8402. Gestão da qualidade e garantia da qualidade - Terminologia, Rio de Janeiro: 1997.
ISO, 2000	ISO 9001, *Sistemas de Gestão da Qualidade* – Requisitos. 2000.
ISO, 2002	ISO/IEC International Organization for Standardization/International Electrotechnical Comission 12207:2002–Amendment I: 2002– Information Technology – *Software Life Cycle Processes* – Amendment I. ISO/IEC, 2002.
ISO, 2003	ISO/IEC 15504, *Information Technology - Software Process Assessment*: ISO and IEC, 2003.
Jacobson *et al.*, 1997	Jacobson, I. Griss, M. and Jonsson, P. *Software Reuse—Architecture, Process and Organization for Business Success.* Harlow, England, AWL, 1997.
Jacobson *et al.*, 1998	Jacobson, Booch, & Rumbaugh. *The unified software development process*. Massachusetts, EUA: Addison-Wesley Publishing Company, 1998. 463p
Juran, 1988	Juran, J. M. *Juran on Planning for Quality*. New York, New York: MacMillan, 1988.
Kan, Basili & Shapiro, 1994	Kan, S. H., Basili, V. R. & Shapiro, L. N. *Software Quality: An Overview from the Perspective of Total Quality Management*. IBM Systems Journal 33(l), 419. 1994.

Kitchenham *et al.*, 1996	Kitchenham, B. *et al*. *Software quality: the elusive target*. IEEE *Software*, January, 1996.
Kroll & Kruchten, 2003	Kroll, P. and Kruchten, P. *The Rational Unified Process made easy: a practitioner's guide to the RUP*. Pearson Education, 2003.
Kruchten, 1999	Kruchten, Philippe. *Rational Unified Process - An Introduction*. Addison-Wesley, 1999.
Kruchten, 2001	Kruchten, Philippe. *Software Maintenance cycles with the RUP*. White Paper, v4, maio de 2001.
Manzoni & Price, 2003	Manzoni, L. V. and Price, R. T. *Identifying Extensions required by RUP (Rational Unified Process) to comply with CMM (Capability Maturity Model) Levels 2 and 3*. IEEE Transactions on *Software* Engineering, Vol. 29. No. 2, February 2003.
Marckzak *et al.*, 2003	Marckzak, S. *et al.*. *Uma proposta de organização e funcionamento da função de garantia de qualidade de software em um contexto de implantação do SW-CMM*. In: Anais do II Simpósio Brasileiro de Qualidade de *Software*. Fortaleza, Brasil. 2003.
Marconi & Lakatos, 2001	Marconi, Marina de A. & Lakatos, Eva Maria. *Metodologia do Trabalho Científico: procedimentos básicos, pesquisa bibliográfica, projeto e relatório, publicações e trabalhos científicos*. 6 ed. São Paulo: Atlas, 2001.
Massoni *et al.*, 2002.	Massoni, T., Sampaio A. & Borba, P. *A RUP-Based Software Process Supporting Progressive Implementation.* In: Issues and Trends of IT *Management* in Contemporary Organizations. Pp 480 a 484.
Minasi, 2000	Minasi, M. *Software Industry's Bugs Unacceptable*. Houston Chronicle, Houston, TX, January 30, 2000. p5.
Mohaghegi, 2002	Mohaghegi, P. *Software Engineering Processes RUP and XP*. Lecture Notes to IKT-2340 "Open Systems Seminar" at Hogskolen Agder, 2002. Disponível em http://fag.grm.hia.no/ikt2340/year2002/themes/process/notes/2-RUP-XP.pdf. Acessado em janeiro de 2004.
Musa *et al.*, 1987	Musa, J. D., A. Ianinno & K. Okumoto. *Engineering and Managing Software with Reliability Measures*. McGraw-Hill, 1987.
Myers, 1979	Myers, Glenford J. *The Art of Software Testing*, John Wiley & Sons, Inc., New York. 1979.
Paulk *et al.*, 1993a	Paulk, M. C. *et al.*. *Capability Maturity Model for Software*. Version 1.1 (CMU/SEI-93-TR-024, ADA 263403). Pittsburgh, PA: *Software* Engineering Institute, Carnegie Mellon University, 1993.
Paulk *et al.*, 1993b	Paulk, M. C. et. al. *Key Practices of the Capability Maturity Model*, Version 1.1. 1993.

Phillips & Kemp, 2002	Phillips C. and Kemp E.A. *In support of User Interface Design in the Rational Unified Process.* In: Proceedings of the Third Australasian User interface Conference, J. Grundy and P. Calder, Pag 21-27, Australian Computer Society, January, February, 2002.
Pires *et al.*, 2004	Pires, Carlos Giovano *et al.*. *O Processo de Melhoria de Software do Instituto Atlântico.* In: Anais do III Simpósio Brasileiro de Qualidade de *Software*. Brasília, 2004.
Prahalad & Krishnan, 1999	Prahalad, C. K., & Krishnan, M. S. *The New Meaning of Quality in the Information Age*. Harvard Business Review (77:5). September/October 1999, pp. 109-118.
Pressman, 2000	Pressman, Roger S. *Engenharia de Software: uma abordagem prática.* 5ª ed. São Paulo: McGrall-Hill, 2000.
Reinehr *et al.*, 2003	Reinehr, Sheila S., Balduino, Ricardo, Machado, Cristina & Pessoa, Marcelo S. *Implementing ISO/IEC 12207 Standard Using Rational Unified Process.* In: Proceedings of the International Conference on *Software* Engineering Research and Practice, SERP '03, June 23 - 26, 2003, Las Vegas, Nevada, USA, Volume 2. pags 667-680.
Reitzig, 2003	Reitzig, Rolf W. *Using Rational Software Solutions to Achieve CMMI Level 2*. Janeiro, 2003. Disponível em http://www.the*Rational*edge.com/content/jan_03/f_CMMI_rr.jsp. Acessado em junho de 2004.
Rocha *et al.*, 2001	Rocha, A. R. C. *et al.*. *Qualidade de software: Teoria e prática*. Prentice Hall, 2001.
Royce, 1998	Royce, Walker. *Software* Project *Management*: *A Unified Framework*. Addison Wesley Longman, 1998.
RUP, 2003	*Rational Software* Corporation, *Rational Unified Process*, Version 2003.06.00.65, CD-ROM, *Rational Software*, Cupertino, California, 2003.
Schulmeyer & McManus, 1999	Schulmeyer, G. and McManus, J. *Handbook of Software Quality Assurance.* New Jersey: Prentice Hall, 1999.
Smith, 2003	Smith, J. *Inconsistencies found in RUP Test Drive with IEEE 1074.* IBM by Chuck Walrad of Davenport Consulting, Inc., 2003. Disponível em: http://www.p1074-workgroup.org/ 3M_Compatibility_Team/references/Annex%20A%20table-RUP.doc. Acessado em dezembro de 2003.
Sommerville, 2003	Sommerville, I. *Software* Engineering. 6ª ed. Addison-Wesley: São Paulo, 2003.
Sousa & Furtado, 2003	Sousa & Furtado (2003) *RUPi – A Unified Process that Integrates Human-Computer Interaction and Software Engineering*. IN: 10th International Conference on Human - Computer Interaction. Creta, Grécia, junho de 2003.

STSC-TR, 2000	*Software Quality Assurance*: *a technical report outlining representative publications on the subject*. Revision 0.2. STSC e Computer Resources Support Improvement Program (CRSIP):April, 2000.
SWEBOK, 2001	*Guide to the Software Engineering Body of Knowledge*. IEEE, 2001
SWEBOK, 2004	*Guide to the Software Engineering Body of Knowledge*. IEEE, 2004 version.
The Chaos Study, 1994	*The Chaos Study*. The Standish Group International, Inc., Dennis, MA. 1994.
Ulferts, 2005	Ulferts, Karen. *Why isn´t there a RUP workflow for Software Quality Assurance?* WhitePaper in: The *Rational* Edge. Disponível em: www-106.ibm.com/developerworks/*Rational*/library/jun05/ulferts/index.html. Acessado em junho de 2005.
Unhelkar, 2003	Unhelkar, B. *Process Quality Assurance for UML-based projects*. In: The Addison-Wesley, 2003.
Xie & Notkin, 2002	XIE, Tao & NOTKIN, David. *Macro and Micro Perspectives on Strategic Software Quality Assurance in Resource Constrained Environments*. ACM, EDSER – 4, maio de 2002.
Yin, 2002	Yin, Robert K. *Case Study Research: design and methods*, 3/e. Thousand Oaks, CA: Sage Publications, 2002.

Apêndice A

REVISÕES DE *SOFTWARE*

As Revisões de *Software* abordadas neste trabalho estão baseadas no padrão IEEE 1028 (1997) - *IEEE Standard for Software Reviews* - que também especifica os procedimentos para realização das revisões. As revisões abrangem 5 tipos:

- Auditorias;
- Inspeções;
- *Walkthroughs*;
- Revisões Técnicas;
- Revisões Gerenciais.

As revisões, de acordo com o padrão IEEE 1028 (1997), devem ser realizadas de forma sistemática e são aplicáveis aos diversos processos do ciclo de vida de *software*: aquisição, fornecimento, desenvolvimento, operação, e manutenção. Sistematizar uma revisão significa dotá-la de procedimentos documentados para sua realização, de registro de seus resultados e da participação dos envolvidos. Aspectos detalhados sobre essas cinco revisões são apresentados a seguir:

Auditorias

Auditorias são revisões de *software* que avaliam o processo de *software* e o produto de *software* em relação à conformidade com padrões, planos, diretrizes, legislação e procedimentos regulamentados. Auditorias caracterizam-se pela independência e objetividade que devem acompanhar sua realização, ou seja, elas devem ser executadas por pessoas não relacionadas diretamente com o objeto da avaliação. Baseada em critérios objetivos, a auditoria busca identificar registros de que padrões e procedimentos estão sendo seguidos e se o processo utilizado é suficiente e efetivo para atender às necessidades de qualidade do *software*.

Auditorias podem ser realizadas por uma única pessoa ou por uma equipe de até cinco pessoas. O papel fundamental neste processo é o de *Auditor líder*, que é o responsável pela condução da Auditoria, pela tomada de decisões e pelas recomendações de ações corretivas relacionadas à Revisão. Outro papel importante é o

da *unidade auditada*, que deve prover as informações necessárias, conforme requerido pelos auditores, e implementar as recomendações e ações corretivas após a conclusão da Auditoria.

Usualmente, a auditoria é iniciada por uma reunião de visão geral para que auditores e a unidade auditada possam examinar e concordar sobre os critérios a serem abordados para a auditoria, como: (*i*) propósito e escopo da auditoria; (*ii*) informações sobre a unidade auditada; (*iii*) produtos a serem auditados; (*iv*) critérios de avaliação; (*v*) requisitos de recursos; (*vi*) cronograma; (*vii*) *checklists*; (*viii*) relatos de não conformidades; e (*ix*) atividades de acompanhamento requeridas.

Antes de a Auditoria começar, a Gestão é avisada e os Auditores são treinados sobre os objetos de sua atuação. São coletadas e analisadas evidências, sobre conformidades e não-conformidades, em relação aos critérios de auditoria. Um relatório contendo os resultados da Revisão é preparado e distribuído a todos os interessados. As ações corretivas são determinadas pelo auditor em conjunto com a unidade auditada, para remover e prevenir as não conformidades e para implementar as ações necessárias.

As evidências buscadas pela auditoria podem ser tanto de não conformidades como de conformidades, em relação aos critérios de auditoria, e podem ser obtidas por meio de entrevistas, exames de documentos e investigações adicionais, se necessário. Não conformidades devem abranger padrões e procedimentos não aplicados totalmente ou aplicados de forma incorreta, e deverão ser categorizados em relação aos efeitos causados sobre a qualidade do produto, o custo e o cronograma do projeto.

A auditoria é considerada completa quando o relatório de auditoria for entregue aos interessados e quando as ações de acompanhamento incluídas no escopo da Auditoria forem executadas, revisadas e aprovadas.

Inspeções

As inspeções são Revisões de *Software* de caráter formal, que possuem procedimentos e papéis bem definidos para realizar a avaliação de um produto de *software*. Fazem uso de *checklists* e devem ser realizadas com a participação de especialistas treinados no processo de inspeção e de um moderador. Quando aplicadas a determinados estágios do ciclo de vida, são conhecidas como inspeções por fase.

O principal objetivo de uma inspeção é a detecção de erros ou identificação de anomalias, isto é, quaisquer desvios de expectativas baseados em requisitos,

documentos de projeto, documentos de usuário, padrões ou na percepção e/ou experiência de alguém. Uma inspeção possui as seguintes características:

- verifica se o produto de *software* satisfaz suas especificações, seus atributos de qualidade especificados, e está em conformidade com padrões, normas, legislação e procedimentos;
- verifica se existem desvios de padrões e especificações;
- coleta dados sobre o processo de engenharia de *software*, em termos de anomalias e esforço, usando esses dados para melhorar o próprio processo de inspeção.

A inspeção é conduzida por um "facilitador" imparcial, devidamente treinado. Para cada anomalia identificada em uma inspeção, deve ser determinada uma ação remediadora ou investigativa. Um aspecto importante é que a resolução da anomalia não deve ser feita durante a inspeção, mas posteriormente à realização do encontro. A coleta de dados sobre o processo é altamente recomendável, porém não obrigatória.

Os principais papéis em uma inspeção são: o *Inspetor líder* que é responsável pelas tarefas administrativas da inspeção, sobretudo o planejamento e a preparação, e pela coleta de dados sobre o processo; o *Autor* que é responsável por fazer com que a revisão atenda aos critérios previamente estabelecidos; o *Inspetor* que identifica e descreve anomalias no produto de *software*, a partir de diferentes pontos de vista do produto.

Para realizar inspeções, devem ser utilizados documentos como: declaração de objetivos da inspeção, o produto a ser inspecionado, procedimento documentado para realizar a inspeção, formulários de relato de inspeção, lista de anomalias correntes, *checklists* de inspeção, além de padrões, legislação, planos e normas regulamentadoras.

A inspeção deve ser devidamente planejada e documentada, e pode ser realizada em diversas etapas ao longo do ciclo de vida de *software*: aquisição, fornecimento, desenvolvimento, operação, e manutenção de *software*, por solicitação do gerente de projetos, do gerente de qualidade, ou do autor do produto.

O processo de inspeção é composto pelos passos a seguir. O Autor disponibiliza previamente o material sobre a inspeção para o Inspetor Líder. Antes da reunião, os membros da equipe de Inspeção examinam o produto de *software* e toda a documentação pertinente. Anomalias já identificadas são documentadas e enviadas ao Inspetor Líder, que as classificará e encaminhará ao Autor do produto. Durante a

reunião de inspeção, o Autor apresenta o produto aos demais e o Inspetor Líder fala sobre o *checklist* de Inspeção, os papéis, e as anomalias típicas identificadas em produtos de *software* similares. O foco da reunião de Inspeção é a identificação de anomalias e não a sua resolução. O produto de *software* é revisado e as anomalias detectadas são documentadas. É avaliado se o encontro de Inspeção pode ser encerrado.

Um dos critérios para verificar o encerramento do encontro de Inspeção é a denominada disposição do produto de *software*, a qual possui três identificações possíveis em relação ao produto que está sendo avaliado:

- o produto é aceito como está ou com um mínimo retrabalho, desde que este não requeira verificação posterior;
- o produto é aceito com retrabalho de verificação, ou seja, apenas é aceito após o Inspetor líder ou algum membro da equipe de inspeção verificar o re-trabalho a ser realizado pelo Autor;
- o produto sofrerá uma nova Inspeção, onde serão, no mínimo, avaliadas: as partes do produto que foram modificadas desde a última inspeção que solucionam as anomalias, e os efeitos das mudanças realizadas.

A inspeção será considerada completa quando as etapas acima descritas forem concluídas e existir a documentação da inspeção contendo informações sobre o projeto inspecionado, o produto inspecionado, os membros da inspeção, a duração do encontro de inspeção e o tamanho do produto inspecionado, os objetivos da inspeção identificando se eles foram atingidos, a lista completa das anomalias identificadas e um resumo de anomalias por categoria, a disposição do produto de *software*, as estimativas do esforço de retrabalho, e a data de conclusão do retrabalho, além do tempo gasto na preparação dos inspetores.

Devem ser coletados dados sobre o processo de inspeção, que auxiliam na análise da qualidade do produto de *software* e da efetividade do processo de *software*, além da efetividade do próprio processo de inspeção. É fundamental a classificação das anomalias identificadas para a análise dos dados coletados. Outras informações podem ser coletadas, como a identificação do produto de *software*, do Inspetor líder, dos tempos de preparação e de inspeção, do volume do material inspecionado e da disposição do produto de *software*.

O padrão IEEE 1028 (1997) sugere uma categorização baseada em: classes de anomalias (identificando ausência, presença desnecessária, ambigüidade, inconsistência, não conformidade a padrões, dentre outras); e na classificação de anomalias de acordo com o potencial impacto sobre o produto de *software* (maior, se poderia resultar em falha de *software*, ou menor, se mesmo originada de desvios a partir de especificações relevantes, não causariam falhas no *software*). A classificação de anomalias também pode ser realizada com base no padrão IEEE 1044 (1993).

Outro aspecto importante das inspeções é a preocupação com a melhoria do próprio processo de inspeção e das atividades de *software* usadas para produzir os produtos de *software*. Aspectos da inspeção poderão ser analisados com foco na melhoria: os *checklist*s, os tempos de preparação e inspeção, o número de participantes. A análise visa determinar conexões entre estes aspectos e a quantidade e severidade de anomalias identificadas.

O processo de inspeção é considerado um dos mais efetivos na detecção de anomalias sobre produtos de *software*, melhorando também o processo de desenvolvimento desses produtos. Estatísticas de projetos com uso efetivo de Inspeção reportam uma taxa de 80% na detecção de defeitos.

Walkthrough

Os *walkthroughs* são considerados revisões de *software* de caráter menos formal que as inspeções, pois possuem como principal propósito educar seu público-alvo sobre a necessidade de realizar avaliações em produtos de *software*. São recomendados como um modo inicial de uma organização trabalhar a revisão de seus produtos de *software*.

Os principais objetivos dos *walkthrough*s são:

- detectar anomalias;
- melhorar o produto de *software*;
- considerar implementações alternativas;
- avaliar a conformidade a padrões e especificações.

Walkthroughs podem ser utilizados com variações das técnicas e estilos aplicados e provendo ou não treinamento aos participantes. Códigos-fonte, manuais de instalação, e documentos de projeto e de especificação de requisitos são exemplos de produtos de *software* que podem ser submetidos a um *walkthrough*. Essa técnica auxilia

na identificação de várias deficiências, como problemas de modularidade no projeto ou no código, e especificações que não podem ser testadas na prática.

Existem pelo menos dois papéis principais durante um *walkthrough*: o *Líder de walkthrough*, que conduz a reunião e distribui a documentação necessária, além de preparar a declaração dos objetivos do *walkthrough* e assegurar que os membros da equipe cheguem à conclusão sobre as decisões e ações identificadas em cada item sob discussão; e o *Autor*, que apresenta o produto de *software* na reunião.

Os critérios de entrada para o processo de *walkthrough* são: a declaração dos objetivos da reunião, o produto de *software* a ser examinado e os padrões pertinentes à etapa do ciclo de vida do produto de *software* (aquisição, desenvolvimento, manutenção etc.) em avaliação. Podem ser utilizados ainda planos e procedimentos e uma categorização das anomalias.

Um *walkthrough* somente deverá ser realizado se a declaração de seus objetivos tiver sido estabelecida pela equipe gerencial, que constitui o público-alvo do resultado da revisão, e se os critérios de entrada enumerados estiverem disponíveis.

A reunião de *walkthrough* acontece da seguinte forma: O Líder de *walkthrough* identifica a equipe, planeja o cronograma do encontro e seleciona o local da reunião, além de distribuir o material necessário aos participantes, permitindo tempo necessário para a preparação destes. Antes da Revisão, os membros da equipe examinam os critérios de entrada para a reunião de *walkthrough* e o produto de *software* distribuído previamente pelo Líder de *Walkthrough*. Eles identificam pontos a serem discutidos durante a reunião, aplicáveis a todo o produto ou que dizem respeito a apenas uma parte do produto. Os pontos identificados são documentados e enviados ao Líder do *walkthrough*, o qual classificará as anomalias para uso efetivo do tempo da reunião e as encaminhará ao Autor do produto de *software*. O Autor apresenta uma visão geral do produto de *software* para os membros da equipe, e passa-se a discutir os pontos do produto. Após a reunião, o Líder de *walkthrough* deve reportar os resultados da reunião, identificando anomalias, ações, decisões e outras informações pertinentes.

Para que um *walkthrough* seja considerado completo é necessário que todo o produto de *software* tenha sido examinado, as ações requeridas e as recomendações tenham sido registradas e a documentação de saída do *walkthrough* esteja completa, identificando: o produto examinado; os membros da equipe de *walkthrough;* os objetivos

do *walkthrough* e se eles foram atingidos; a lista de recomendações feitas para cada anomalia identificada; a lista de ações a serem concretizadas, com datas e responsáveis; recomendações sobre como superar deficiências e tratar anomalias não resolvidas; e propostas para melhorar a realização das próximas reuniões de *walkthrough*.

De modo similar às inspeções, dados devem ser coletados sobre o *walkthrough* e anomalias devem ser classificadas. Isto visa melhorar o próprio processo de *walkthrough* e as atividades utilizadas na geração dos produtos de *software*.

Revisões Técnicas

Revisões Técnicas de *software* são realizadas com o propósito de avaliar um produto em relação à sua conformidade com padrões e especificações, e em relação ao uso pretendido do *software*. A avaliação é efetuada por pessoas qualificadas para determinar a qualidade do produto, em termos das seguintes informações:

- o produto de *software* atende a suas especificações;
- o produto de *software* está em conformidade com normas, padrões, planos e procedimentos aplicáveis ao projeto que o desenvolve;
- mudanças no produto de *software* são adequadamente implementadas e afetam apenas as partes do *software* identificadas na solicitação de mudança do *software*.

As Revisões Técnicas podem fazer recomendações ou avaliar alternativas, o que pode ser feito durante uma ou mais reuniões. Uma avaliação não precisa avaliar todos os aspectos de um produto. Os produtos de *software* avaliados fazem parte do processo de *software* da organização, e são exemplos as Especificações de Requisitos de *Software*, os Documentos de Projeto (*design*) de *Software*, a documentação de Teste, dentre outros documentos.

Em uma Revisão Técnica, as responsabilidades são definidas por meio dos papéis, onde mais de um papel pode ser exercido por uma única pessoa. Há um *tomador de decisões*, que conduz a Revisão Técnica e determina se seus objetivos foram atingidos; um *Líder da revisão*, que verifica se a Revisão atingiu seus objetivos, além de emitir e distribuir os produtos do processo de Revisão Técnica; o *Staff técnico*, que avalia o produto de *software*; e o *Staff gerencial*, quando houver situações que requeiram a tomada de decisão gerencial. O cliente ou o usuário pode ser representado por uma pessoa.

São produtos de entrada para o processo de Revisão Técnica: a declaração dos objetivos da Revisão, o produto de *software* a ser examinado, o Plano de Gerenciamento de Projeto, a lista de anomalias (defeitos) e os procedimentos para realizar a revisão. Podem ser utilizados ainda: normas, padrões, procedimentos e legislação pertinente para a avaliação do produto, lista de anomalias categorizadas, incluindo ainda relatórios de revisão relevantes. Antecedendo a realização da Revisão Técnica, deve haver uma declaração de objetivos estabelecida e os itens de entrada da Revisão devem estar disponíveis.

A Revisão Técnica consiste dos passos a seguir. O Líder da revisão identifica a equipe de Revisão, com o devido suporte gerencial, e determina as responsabilidades de cada um dos membros da equipe. Ele planeja o cronograma e o local do encontro e distribui o material necessário aos participantes, dando tempo necessário para sua preparação. Ele também reúne os comentários dos demais membros e os encaminha ao *Autor* para discussão. Como etapa do Planejamento da Revisão, a equipe da Revisão Técnica deverá determinar se a avaliação de alternativas será realizada durante a reunião de Revisão ou se em outras reuniões realizadas à parte.

Uma pessoa tecnicamente qualificada apresenta uma visão geral sobre o produto sob avaliação para a equipe de Revisão. Antes da Revisão, os membros da equipe: examinam os critérios de entrada para a reunião e o produto de *software* sob avaliação; documentam os defeitos identificados, enviando-os ao Líder da revisão. Este classifica previamente as anomalias e as encaminha ao Autor do produto de *software*. Durante a reunião de Revisão, a agenda de avaliação do produto de *software* e as anomalias são determinadas. O produto de *software* é avaliado, verificando se está completo e em conformidade com normas, leis e padrões aplicáveis ao projeto. Verifica-se se as mudanças identificadas no produto de *software* foram adequadamente implementadas e se afetaram apenas partes específicas do *software*. Também se determina se o produto de *software* atende os requisitos de uso pretendido e se está pronto para a próxima atividade. São verificadas ainda: a existência de uma lista de anomalias de *hardware* ou de discrepâncias nas especificações; se anomalias foram identificadas e se a lista de ações, enfatizando os riscos, foi gerada. O encontro deve ser devidamente documentado.

Após a Revisão Técnica do produto de *software*, as anomalias encontradas no produto devem constar de uma lista de anomalias, e as recomendações para a gestão devem ser descritas. Uma vez que as anomalias encontradas sejam suficientemente críticas e numerosas, o Líder da Revisão pode recomendar a realização de uma nova

Revisão sobre o produto modificado, cobrindo áreas do produto que tenham sido modificadas para solucionar as anomalias encontradas e aquelas que sofreram os efeitos das mudanças.

Para que uma Revisão Técnica seja considerada completa é necessário que as etapas do processo descrito no item anterior tenham sido concluídas e que a documentação de saída esteja completa, contendo os seguintes itens: o projeto revisado; os membros da equipe de revisão; os objetivos da Revisão Técnica e o registro do atendimento desses objetivos; o produto de *software* revisado e as entradas específicas para a Revisão; a lista de anomalias do produto, tanto as solucionadas como as não solucionadas; a lista de anomalias de *hardware* e de sistema identificadas e não solucionadas, com os devidos itens de ação recomendados; a lista de itens sobre os quais a gerência pode atuar, o *status* dos itens de ação (abertos ou concluídos), com o responsável e data prevista, se aberto, e data de conclusão, quando cumprido; além de quaisquer recomendações feitas sobre como tratar itens e anomalias não solucionados.

Os principais objetivos de uma Revisão Técnica são avaliar o produto em relação a padrões e especificações, e assegurar a integridade do produto quando da realização de mudanças. A equipe de Revisão Técnica solicita a atuação de lideranças técnicas e gerenciais para solucionar problemas no *software* com base nas recomendações oriundas do processo de Revisão. A técnica não requer treinamento formal de facilitadores e a participação de usuários e/ou clientes é facultativa.

Revisões Gerenciais

Revisões Gerenciais, no contexto de *software*, são reuniões realizadas com a participação de pessoas que possuem responsabilidade gerencial direta sobre um *software*, com um amplo conjunto de finalidades:

- Monitorar o progresso do trabalho, determinando o *status* de planos e cronogramas;
- Confirmar os requisitos e o tratamento a ser dado a eles ao longo do projeto;
- Avaliar a efetividade de abordagens de gerenciamento utilizadas pelo projeto, identificando adequações e inadequações de procedimentos gerenciais.

As Revisões Gerenciais avaliam diversos produtos de *software* que fazem parte do processo de *software* da organização, como os Planos de Gestão do Projeto, Garantia da Qualidade, Relatórios de Acompanhamento, Planos de Manutenção, dentre outros.

As responsabilidades dentro de uma Revisão Gerencial são definidas com base em papéis bem específicos. Com participação obrigatória, enumeramos o *Staff* gerencial, que deve ter participação ativa no processo de Revisão, especialmente os responsáveis pelo sistema como um todo; e o *Staff* técnico que é responsável por prover a informação necessária para que o *Staff* Gerencial cumpra com suas responsabilidades. A participação de cliente é facultativa.

Os itens de entrada para o processo de Revisão Gerencial são: a declaração dos objetivos da Revisão; o produto de *software* a ser examinado; o Plano de Gestão de Projeto; Status do produto de *software* completo ou em desenvolvimento em relação ao Plano; lista de defeitos; e os procedimentos de Revisão Gerencial documentados. Podem ser utilizados ainda: normas, padrões, procedimentos, legislação pertinente para a avaliação do produto, além de relatórios de revisão relevantes.

A conclusão de um produto de *software* específico ou de uma atividade prevista nos planos pode dar início a uma Revisão Gerencial de *software*. Antes de realizar a Revisão Gerencial, a declaração de seus objetivos deve ter sido estabelecida pela equipe gerencial, e os itens de entrada da Revisão devem estar disponíveis.

Os procedimentos para realização de uma Revisão Gerencial são descritos a seguir. O Líder da revisão identifica a equipe de revisão, determinando as responsabilidades de cada um de seus membros. Alguém qualificado deve apresentar uma visão geral do encontro aos demais participantes, quando solicitado pelo Líder revisor, podendo ocorrer dentro da reunião de Revisão ou em outra reunião à parte. Antes da revisão, os membros da equipe examinam os critérios de entrada para a reunião, o produto de *software* sob avaliação, e verificam se os defeitos identificados foram documentados e enviados ao Líder da revisão. Os defeitos identificados são classificados previamente e encaminhados ao Autor do produto de *software*.

As reuniões devem atingir os seguintes pontos: os objetivos da Revisão devem ser verificados; o produto de *software* deve ser avaliado à luz dos objetivos da Revisão; devem ser revisados o *status* do projeto e de seus planos e cronogramas; defeitos identificados deverão ser revisados. Outros objetivos da reunião são: a geração de uma

lista de ações com ênfase nos riscos, avaliando aqueles que possam comprometer o sucesso do projeto; e a documentação do encontro.

Com base nas informações avaliadas em uma Revisão Gerencial, o processo de tomada de decisões é facilitado dentro da empresa, sobretudo no que diz respeito às decisões focadas em ações corretivas, mudanças na alocação de recursos e mudanças no escopo do projeto.

Projeto X
Plano de Garantia de Qualidade

Versão 1.4

Projeto X	Versão: 1.4
Plano de Garantia de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

Histórico de Revisão

Data	Versão	Descrição	Autor
dd/mm/aaaa	*1.0*	*Planejamento de Qualidade para a Fase de Iniciação*	*Gerente de Qualidade e Engenheiro de Qualidade*
dd/mm/aaaa	*1.2*	*Planejamento de Qualidade para a Fase de Elaboração*	*Gerente de Qualidade e Engenheiro de Qualidade*
dd/mm/aaaa	*1.3*	*Planejamento de Qualidade para a Fase de Construção*	*Gerente de Qualidade e Engenheiro de Qualidade*
dd/mm/aaaa	*1.4*	*Planejamento de Qualidade para a Fase de Transição*	*Gerente de Qualidade e Engenheiro de Qualidade*

Projeto X	Versão: 1.4
Plano de Garantia de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

Índice

Projeto X	Versão: 1.4
Plano de Garantia de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

Plano de Garantia de Qualidade

1. Introdução

1.1 Propósito

O propósito deste Plano de Garantia de Qualidade é estabelecer os objetivos de qualidade de software para o projeto e as ações de qualidade a serem realizadas para atingir esses objetivos.

1.2 Escopo

Este Plano de Garantia de Qualidade se aplica ao Projeto X. Serão avaliados todos os artefatos do processo de desenvolvimento e a aplicação gerada (executável) e todas as atividades de desenvolvimento planejadas de acordo com o Plano de Desenvolvimento de Software do Projeto X.

1.3 Definições, Acrônimos e Abreviaturas

GUI – Graphical User Interface ou Interface Gráfica de Usuário

SQA – Software Quality Assurance ou Garantia de Qualidade de Software

1.4 Documentos de Referência

Os documentos referenciados neste Plano de Garantia de Qualidade, relativos ao Projeto X, são:

- *Formulários de Modelagem de Negócios*
- *Visão*
- *Regras de Negócios*
- *Especificação de Requisitos de Software e Protótipo de GUI (Interface Gráfica de Usuário)*
- *Especificação de Casos de Uso*
- *Realização de Casos de Uso*
- *Documento de Arquitetura de Software*
- *Plano de Testes*
- *Plano de Desenvolvimento de Software e respectivos Planos de Iteração*
- *Manual do Usuário*
- *Plano de Implantação*

2. Objetivos de Qualidade

Os objetivos de qualidade para o Projeto X foram definidos visando atender os objetivos do projeto de identificação e mapeamento de requisitos com a unidade de negócios. Posteriormente os objetivos foram sendo refinados de acordo com cada Fase/Iteração. No Projeto X, cada Fase possui uma iteração. Os objetivos de qualidade são descritos para cada fase.

- *Objetivos de Qualidade para a Fase de Iniciação*
  - ✓ *Avaliar se o planejamento do projeto foi realizado adequadamente e as dificuldades apresentadas no planejamento, além de avaliar a definição do escopo do projeto (definição dos requisitos e casos de uso e sua validação com os usuários) e a correspondente documentação técnica.*
- *Objetivos de Qualidade para a Fase de Elaboração*
  - ✓ *Avaliação de Qualidade 1: Avaliar a qualidade das Especificações de Casos de Uso, o mapeamento entre as Especificações de Casos de Uso e os requisitos, e avaliar o mapeamento dos requisitos na estrutura de arquitetura definida para a aplicação.*

- ✓ *Avaliação de Qualidade 2: Avaliar a rastreabilidade entre a documentação de análise/design e a implementação do software e de seus componentes, além do planejamento e execução dos testes de validação da arquitetura.*

- *Objetivos de Qualidade para a Fase de Construção*
  - ✓ *Avaliar o impacto das mudanças solicitadas pelos stakeholders sobre os requisitos, e a rastreabilidade dessas mudanças sobre os casos de uso, realizações de casos de uso, projeto do banco de dados, componentes já desenvolvidos, e sobre o executável como um todo.*
- *Objetivos de Qualidade para a Fase de Transição*
  - ✓ *Avaliar se as providências para implantar o software foram tomadas e se os erros críticos identificados nos testes realizados na fase anterior foram corrigidos antes de o software entrar em produção.*

3. Gerenciamento

3.1 Organização

A estrutura da organização, que influencia e controla a qualidade de software em termos de unidades, papéis e responsabilidades, é descrita a seguir:

- *Há uma unidade composta pela equipe de desenvolvimento do software, que é responsável pelo gerenciamento dos projetos de desenvolvimento de software.*
- *Há uma unidade responsável pelo gerenciamento da qualidade que contém:*
  - ✓ *uma equipe de qualidade, que é independente da equipe de testes e independente em relação ao projeto de desenvolvimento de software;*
  - ✓ *uma equipe de testes que presta serviço de avaliação do produto para a unidade de desenvolvimento de sistema.*

3.2 Tarefas

As tarefas de Garantia de Qualidade que serão realizadas no projeto são descritas a seguir:

Etapa do Ciclo de Vida	Tarefas de Garantia de Qualidade	Marcos do Projeto	Relacionamento entre as atividades de qualidade e os marcos do projeto
Fase de Iniciação	✓Auditoria de Qualidade ✓Orientações de Qualidade	✓Planejamento da próxima iteração/fase ✓Validação de requisitos em nível macro com usuário	✓Problemas identificados no planejamento atual poderiam ser resolvidos antes que impactassem negativamente o planejamento do projeto como um todo e a entrega do software ao cliente. ✓A Identificação dos Casos de Uso e de seus riscos associados, com os usuários, proporciona uma melhor compreensão do sistema e um planejamento mais acurado.

Fase de Elaboração	✓Revisões Técnicas (Revisor Técnico de Requisitos) ✓Auditorias de Qualidade ✓Orientações de Qualidade	✓Especificações de Casos de Uso detalhadas antes de validar com o usuário ✓*Design* (Realizações de Casos de Uso) e Arquitetura	✓Identificação e correção de problemas nos requisitos e Casos de Uso aconteceriam antes de validação com usuários. ✓Rastreabilidade entre requisitos, Casos de Uso e elementos de arquitetura e design auxiliam no tratamento de mudanças nos requisitos.
	✓Auditorias de Qualidade ✓Orientações de Qualidade	✓Testes de validação da arquitetura	✓Rastreabilidade entre a documentação de análise/design e a implementação do software e de seus componentes também possibilitam ganho de tempo ao se lidar com mudanças e estimar prazos, enfocando a manutenibilidade. ✓Planejamento e execução dos testes de validação da arquitetura proporcionam segurança de que a aplicação tem robustez para atender os requisitos dos usuários.
Fase de Construção	✓Auditorias de Qualidade ✓Orientações de Qualidade	✓Impacto das mudanças solicitadas pelos usuários	✓O impacto das mudanças solicitadas pelos stakeholders sobre os requisitos, e sua rastreabilidade em relação aos casos de uso, realizações, projeto do banco de dados, componentes e executável, auxilia no atendimento do usuário com maior brevidade.
Fase de Transição	✓Auditorias de Qualidade ✓Orientações de Qualidade	✓Testes de Regressão ✓Implantação do Software	✓Correção de erros críticos identificados em testes anteriores precedendo a disponibilização do software em produção e a antecipação das providências de implantação melhoram a qualidade do produto e evitam problemas no momento da implantação.

3.3 Papéis e Responsabilidades

Os papéis e as responsabilidades em relação à realização das tarefas de Garantia de Qualidade para o Projeto X são:

- *gerente de qualidade: estabelecer objetivos de qualidade e definir ações de qualidade. Reportar resultados para a alta gerência.*
- *engenheiro de qualidade: auditar projeto de desenvolvimento e reportar não conformidades; identificar padrões de qualidade adequados para o projeto; orientar projetos.*
- *gerente do projeto: implementar ações corretivas e preventivas de sua alçada de resolução; identificar em conjunto com a equipe de qualidade os prazos adequados para resolução de não conformidades.*
- *gerente do desenvolvimento de sistemas: implementar ações corretivas e preventivas de sua alçada de resolução. Atuar adequadamente para que as ações de qualidade possam ser executadas.*
- *equipe do projeto: colaborar na definição dos objetivos de qualidade e na implementação das ações corretivas e preventivas para resolução de não conformidades.*

4. Documentação

A documentação do projeto é composta por:

- *Visão*
- *Regras de Negócios*
- *Especificação de Requisitos de Software*
- *Protótipo de GUI (Interface Gráfica de Usuário)*
- *Especificação de Casos de Uso*
- *Modelos UML do software*
- *Realização de Casos de Uso*
- *Documento de Arquitetura de Software*
- *Plano de Testes*
- *Sumário de Testes*
- *Plano de Desenvolvimento de Software e respectivos Planos de Iteração*
- *Lista de Riscos*
- *Manual do Usuário*
- *Plano de Implantação*

5. Padrões e Guias

Os padrões de qualidade utilizados no projeto e que devem ser utilizados como base para as avaliações de qualidade são:

- *Procedimento Organizacional de Documentação;*
- *Procedimento Organizacional de Gerência de Configuração;*
- *Procedimento Organizacional de Garantia de Qualidade;*
- *Padrão de Componentes e Framework de Desenvolvimento*

6. Métricas

As métricas para este projeto, coletadas em cada fase, são as seguintes:

- *Quantidade de avaliações de qualidade planejadas versus realizadas*
- *Quantidade de Não Conformidades identificadas*
- *Não Conformidades resolvidas no prazo*

7. Revisões de Software

As Revisões de Software que serão realizadas no Projeto X estão relacionadas a seguir:

Fase	Revisão	Responsável	Recursos	Artefatos Avaliados
Iniciação	*Auditoria*	*Equipe de Qualidade*	✓ *Plano de SQA* ✓ *Relatório de Avaliação de Qualidade* ✓ *Guia de Revisões de Software (Apêndice A)*	✓ *Visão* ✓ *Regras de Negócios* ✓ *Especificação de Requisitos de Software* ✓ *Protótipo de GUI (Interface Gráfica de Usuário)* ✓ *Especificação de Casos de Uso* ✓ *Plano de Testes* ✓ *Plano de Desenvolvimento* ✓ *Planos de Iteração* ✓ *Lista de Riscos*

Elaboração	*Revisão Técnica*	*Revisor Técnico*	✓ *Plano de SQA* ✓ *Relatório de Avaliação de Qualidade* ✓ *Guia de Revisões de Software (Apêndice A)*	✓ *Visão* ✓ *Regras de Negócios* ✓ *Especificação de Requisitos de Software* ✓ *Especificação de Casos de Uso* ✓ *Lista de Riscos*
	Auditoria	*Equipe de Qualidade*	✓ *Plano de SQA* ✓ *Relatório de Avaliação de Qualidade* ✓ *Guia de Revisões de Software (Apêndice A)*	✓ *Especificação de Requisitos de Software* ✓ *Protótipo de GUI (Interface Gráfica de Usuário)* ✓ *Especificação de Casos de Uso* ✓ *Modelos UML do software* ✓ *Realização de Casos de Uso* ✓ *Documento de Arquitetura de Software* ✓ *Plano de Testes* ✓ *Plano de Desenvolvimento* ✓ *Planos de Iteração* ✓ *Lista de Riscos*
Construção	*Auditoria*	*Equipe de Qualidade*	✓ *Plano de SQA* ✓ *Relatório de Avaliação de Qualidade* ✓ *Guia de Revisões de Software (Apêndice A)*	✓ *Plano de Desenvolvimento* ✓ *Planos de Iteração* ✓ *Lista de Riscos* ✓ *Especificação de Casos de Uso* ✓ *Modelos UML do software* ✓ *Realização de Casos de Uso* ✓ *Documento de Arquitetura de Software* ✓ *Plano de Testes* ✓ *Sumário de Testes* ✓ *Plano de Implantação*
Transição	*Auditoria*	*Equipe de Qualidade*	✓ *Plano de SQA* ✓ *Relatório de Avaliação de Qualidade*	✓ *Plano de Testes* ✓ *Sumário de Testes* ✓ *Plano de Implantação* ✓ *Lista de Riscos* ✓ *Documentação gerada para o software pronta para disponibilização ao cliente.*

8. Testes

As atividades de Controle de Qualidade são previstas no Plano de Testes do Projeto X, incluindo os testes de validação de arquitetura e os testes do executável.

9. Relato de Problemas e Ações Corretivas

Os problemas identificados durante as avaliações de qualidade serão reportados para a equipe do projeto e para o gerenciamento sênior do desenvolvimento de sistemas, assim como para o cliente ou quaisquer outros interessados, de acordo com a criticidade e o grau de envolvimento dos diversos atores com a não conformidade identificada.

O Engenheiro de Qualidade deve identificar não conformidades que ficaram de ser resolvidas pela equipe de projeto e não o foram no prazo estabelecido. O Engenheiro de Qualidade é responsável por escalar essas não conformidades abertas para o nível de gerenciamento imediatamente superior ao da gerência de projeto, e assim sucessivamente, até que a não conformidade seja resolvida, ou seja, identificada como um desvio.

A atuação do nível de gestão ao receber não conformidades em aberto deve ser compatível com a Política de Qualidade da organização, de forma que a resolução das não conformidades escaladas seja realizada com a maior brevidade possível.

Quando houver o aceite dos stakeholders relevantes para a não resolução de uma não conformidade, esse caso é considerado como uma situação de exceção (também chamado desvio). O desvio deve ser documentado, informando o contexto e a justificativa para que tenha sido considerado um desvio, e informando o responsável pela decisão.

10. Ferramentas, Técnicas e Metodologias

Será utilizada a Técnica de SQA de Auditoria (conforme padrão IEEE 1028-1997) para realização das avaliações de qualidade. Paralelamente serão realizadas ações de mentoring/orientação no projeto.

O Revisor Técnico de Requisitos utiliza a técnica de SQA de Revisão Técnica para realização da Revisão Técnica de Requisitos.

11. Gestão de Configuração

Os artefatos do projeto, incluindo os artefatos de qualidade estão armazenados no repositório de gestão de configuração do projeto, conforme Plano de Gerenciamento de Configuração.

12. Controle de Fornecimento

Não se aplica.

13. Registros de Qualidade

A coleta dos registros de qualidade poderá ser feita manual ou automaticamente, na medida em que as auditorias de qualidade sejam realizadas. Os registros de qualidade serão mantidos sob gestão de configuração no repositório de gestão de configuração organizacional e retidos por uma prazo mínimo de 3 anos.Caso a organização julgue necessário, esse prazo poderá ser estendido para atender objetivos de qualidade da organização.

14. Treinamento

O treinamento necessário para atender os objetivos de qualidade para este projeto diz respeito ao mapeamento de riscos operacionais de processos, em cujo contexto está inserido o desenvolvimento do software.

Projeto X	Versão: 1.4
Plano de Garantia de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

15. Gerenciamento de Riscos

Os principais riscos do projeto identificados pela equipe de qualidade foram inseridos no Plano de Gerenciamento de Riscos.

<Nome do Projeto>

Relatório de Avaliação de Qualidade Auditoria

Versão <x.x>

<Nome do Projeto>	Versão: <x.x>
Relatório de Avaliação de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Histórico de Revisão

Data	Versão	Descrição	Autor
<dd/mm/aaaa>	<x.x>	<detalhes>	<Autor>

<Nome do Projeto>	Versão: <x.x>
Relatório de Avaliação de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Índice

<Nome do Projeto>	Versão: <x.x>
Relatório de Avaliação de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Relatório de Avaliação de Qualidade

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Relatório de Avaliação de Qualidade.]

1.2 Escopo

[Especificar o escopo do relatório informando a qual projeto se aplica, quais os itens de software avaliados, qual a etapa do ciclo de vida abrangida (fase e iteração).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Estrutura de Garantia de Qualidade

2.1 Organização

[Descrever como está organizada a estrutura de qualidade dentro da organização e relacionada aos projetos sob avaliação. Identificar a equipe de Garantia de Qualidade alocada ao projeto]

2.2 Técnicas utilizadas

[Descrever as técnicas a serem utilizadas para Garantia de Qualidade do projeto. Neste template está sendo apresentada a técnica de Auditoria.]

2.3 Fase e Iteração

[Descrever a fase e a iteração na qual está sendo realizada a avaliação de Garantia de Qualidade do projeto.]

3. Avaliação de Qualidade

3.1 Auditoria de Qualidade

3.1.1 Processos de Software

Fase de Iniciação		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender o que construir (determinar o escopo do sistema, para quem o sistema servirá e qual é a melhor solução).	• se a aplicação a ser desenvolvida (*software*), no artefato Visão, foi definida em termos de: - benefícios e oportunidades; - problemas resolvidos pela aplicação; - usuários-alvo; - necessidades e características. • se há descrição de casos de uso e atores; • se há descrição dos requisitos não funcionais da aplicação; • se os protótipos de interface foram gerados, se previstos;	• Visão • Especificação de requisitos de *software* • Especificação suplementar de requisitos • Especificação de caso de uso
2. Identificar casos de uso críticos.	• se as funcionalidades-chave (casos de uso críticos) foram identificadas e priorizadas.	• Especificação de requisitos de *software*
3. Determinar pelo menos uma solução possível.	• se foi identificada pelo menos uma solução arquitetural; • se, para cada solução arquitetural proposta: - a estratégia de evolução do *software* foi estabelecida; - as tecnologias a serem utilizadas foram identificadas e os riscos associados a sua utilização; - decisões relativas à aquisição ou desenvolvimento interno foram consideradas, e os riscos associados a essas decisões foram especificados; - os componentes a serem utilizados (database, middleware) foram considerados em termos de aquisição, reuso, custos de aquisição e se os riscos associados a essas decisões foram especificados; • se o protótipo funcional (se existir) foi avaliado pela equipe de testes. • Se a estratégia de testes foi definida.	• Documento de arquitetura de *software* • Lista de riscos • Realizações de caso de uso • Plano de Testes (Estratégia de Testes) • Sumário de Avaliação de Testes
4. Estimar custos, cronograma e riscos associados ao projeto.	• se foram especificadas informações de: - estimativas; - recursos; - cronograma; - custo do projeto; - listas de riscos.	• Caso de Negócio • Lista de Riscos • Plano de Desenvolvimento de S*oftware*
5. Decidir que processo seguir e que ferramentas utilizar.	• se foram especificadas informações de: - processo definido para o projeto; - ferramentas a serem utilizadas; - *templates* e artefatos. • Se o ambiente (processo, ferramentas, *template*s) foi preparado para uso pelo projeto.	• Caso de Desenvolvimento • Plano de Desenvolvimento de S*oftware*

<Nome do Projeto>	Versão: <x.x>
Relatório de Avaliação de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Fase de Elaboração		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender requisitos detalhadamente.	• se os requisitos funcionais foram descritos e detalhados e representados em especificações de casos de uso; • se todos os casos de uso arquiteturalmente significativos foram detalhados; • se há evidências de que os casos de uso foram testados; • se o Glossário está sendo atualizado; • se mudanças nos requisitos são gerenciadas e rastreadas.	• Especificação de requisitos de *Software* • Especificação de caso de uso • Realização de caso de uso • Glossário • Registros de revisão
2. Projetar, implementar e validar a arquitetura, estabelecendo uma *baseline* de arquitetura.	• se o sistema foi definido em termos de blocos de construção, identificando: - subsistemas e componentes; - interfaces entre os componentes; - decisões e riscos de implementação dos componentes (construção, aquisição, ou reuso); - descrição da interação entre os componentes ao longo dos cenários. • se há evidências da realização dos casos de uso, com as devidas correspondências entre as classes de análise e de *design*; • se há evidências de que as classes foram empacotadas de forma apropriada. • se há evidências da descrição de modelos abrangendo concorrência, processos e *threads*. • Se há modelo de dados do sistema, e se foi identificada a estratégia de armazenamento e recuperação de dados. • se foram utilizados na solução os mecanismos arquiteturais padrões da organização adequados aos problemas. • se há evidências de que o protótipo foi implementado, contemplando cenários críticos. • se há evidências de que houve revisão do código implementado, e se houve teste de código, com implementação de classes de teste pelos desenvolvedores. • se há evidências de que o integrador combinou as partes adequadas (componentes e subsistemas) do sistema; • se há evidências de que foram realizados testes do protótipo (*build*) pela equipe de testes, segundo o Plano de testes.	• Documento de Arquitetura de *software* • Realizações de caso de uso • Especificações de requisitos de *software* • Especificação de caso de uso • Plano de testes • Relatório de testes • Modelo de dados • Modelos UML • Registros de revisão • Plano de integração • Sumário de avaliação de testes
3. Mitigar riscos e produzir estimativa acurada de custo e cronograma	• se os dados do plano de projeto e estimativas de custo foram atualizados.	• Plano de Desenvolvimento de *Software* • Lista de Riscos
4. Refinar Caso de Desenvolvimento e ambiente de desenvolvimento.	• se, no artefato Caso de Desenvolvimento, foram identificadas melhorias e adaptações no uso de processos e ferramentas; • se o ambiente de desenvolvimento foi ajustado para a iteração.	• Caso de Desenvolvimento

Fase de Construção		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Minimizar custos de desenvolvimento e atingir algum grau de paralelismo.	• se há evidências nos modelos UML do projeto de que as responsabilidades do sistema foram divididas em subsistemas bem definidos; • se os artefatos do projeto encontram-se sobre controle de versão e sob gestão de configuração; • se há evidências de que é realizado o planejamento da integração, identificando que determinada versão do *software* é composta por determinados componentes ou subsistemas definidos para o sistema. • se os mecanismos arquiteturais planejados foram considerados entre as opções de projeto; • se há evidências de que os desenvolvedores testam o código (teste de unidade).	• Modelos UML do projeto • Realização de Caso de Uso • Plano de Gestão de Configuração • Mecanismos Arquiteturais • Sumário de Avaliação de Testes
2. Desenvolver um produto completo que esteja pronto para ser entregue ao cliente.	• se há evidências de que todos os casos de uso remanescentes foram projetados, implementados e testados; • se houve revisão do *design* dos Casos de Uso remanescentes; • se todos os cenários dos Casos de Uso foram projetados, implementados e testados. • se as mudanças relacionadas a banco de dados abrangem apenas criação de colunas, de visões e de índices. • são realizados testes de integração e testes sistêmicos; • se há evidências de que são realizados testes de usuários; • se há evidências de que são preparados os artefatos para implantação do *software*.	• Realização de caso de uso • Modelos UML • Especificação de Requisitos de Software • Casos de uso • Registro de Revisão • Plano de Testes • Casos de Testes • Sumário de Avaliação de Testes • Solicitações de mudanças de Bancos de Dados • *Build* • Instruções de Instalação • Manuais de Usuários • Material de Treinamento

Fase de Transição		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Realizar testes beta.	• se os usuários realizaram testes de avaliação do *software* em sua versão beta; • se solicitações de mudança do usuário foram implementadas antes de implantar a versão final do *software*; • se solicitações de mudança para correção de defeitos foram implementadas; • se foram realizados testes de regressão pelo desenvolvedor e pela equipe de testes; • se a documentação do *software* a ser utilizada na implantação foi desenvolvida; • se os testes de sistema planejados foram realizados; • se os testes de aceitação foram realizados;	• Sumário de avaliação de testes • Relatórios de defeitos • Solicitações de mudanças • *Help On line* • Material de treinamento • Manual do usuário • Manual de instalação

	• se *patch releases* foram produzidos para corrigir defeitos significativos.	
2. Treinar usuários.	• se o material de treinamento foi concluído; • se a documentação do usuário foi concluída; • se os manuais operacionais foram preparados e concluídos; • se houve treinamento para os usuários.	• Registros de treinamento • Material de treinamento • Manual de instalação • Manuais do usuário
3. Preparar implantação e converter bancos de dados operacionais.	• se houve necessidade de migração de dados e se ela realmente aconteceu; • se houve necessidade de que o sistema anterior (se for o caso) e sistema novo executarem em paralelo, e se foram registradas análises relacionadas a isto; • se o hardware novo necessário foi adquirido e instalado.	• Documento de Arquitetura de *Software* • Plano de Implantação
4. Preparar empacotamento.	• Se foi definida a estrutura de empacotamento do *software* e se ela está completa.	• Empacotamento de *software* • Plano de Implantação
5. Avaliar as *baselines* de implantação.	• se foram realizados testes de aceitação do produto, considerando a completude das *baselines* de implantação.	• Sumário de Avaliação de Testes • Plano de Gerenciamento de Configuração
6. Registrar lições aprendidas.	• se a Revisão *Post Mortem* foi realizada; • se o artefato Caso de Desenvolvimento foi atualizado com base nas lições aprendidas do projeto.	• Registros de Revisão • Caso de Desenvolvimento

3.1.2 Artefatos

[Avaliar os processos de software nas quatro fases do Processo Unificado, a partir das orientações do subitem anterior.]

Artefato	NC (não conformidade)	Severidade	Ação Corretiva	Prazo	Responsável	Ação Preventiva	Data de Abertura	Data de Fechamento
1. Visão								
2. Caso de Negócios								
3. Glossário								
4. Especificação de Requisitos de Software								
5. Especificação Suplementar								
6. Especificação de Caso de Uso								
7. Lista de Riscos								
8. Plano de Desenvolvimento de Software								
9. Documento de Arquitetura de Software								
10. Realização de Caso de Uso								
11. Modelos UML								
12. Modelo de Dados								
13. Manual do Usuário								
14. Manual de Instalação								
15. Plano de Implantação								
16. Solicitação de Mudanças								
17. Plano de Testes								
18. Sumário de Avaliação de Teste								
19. Registros de Revisão								
20. Plano de Gestão de Configuração								
21. Help On line								
22. Material de Treinamento								
23. [novos artefatos.]								

4. Resultados da Avaliação de Qualidade

[Informar os principais resultados da avaliação e do projeto, a atuação da equipe de qualidade, necessidades do projeto em relação à capacitação.]

<Nome do Projeto>
Plano de Garantia de Qualidade

Versão <x.x>

<Nome do Projeto>	Versão: <x.x>
Plano de Garantia de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Histórico de Revisão

Data	Versão	Descrição	Autor
<dd/mm/aaaa>	<x.x>	<detalhes>	<nome>

<Nome do Projeto>	Versão: <x.x>
Plano de Garantia de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Índice

<Nome do Projeto>	Versão: <x.x>
Plano de Garantia de Qualidade	Data: <dd/mm/aaaa>
<Identificador do documento>	

Plano de Garantia de Qualidade

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informar sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Informar o propósito do Plano de Garantia de Qualidade.]

1.2 Escopo

[Especificar qual o escopo do plano, informando a qual projeto se aplica, quais os itens de software cobertos pelo Plano (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Objetivos de Qualidade

[Referenciar a seção da Especificação de Requisitos de Software que trata dos requisitos de qualidade.]

3. Gerenciamento

3.1 Organização

[Descrever a estrutura da organização, especificando como ela influencia a qualidade de software.]

3.2 Tarefas

[Apresentar as tarefas de Garantia de Qualidade que serão realizadas no projeto, sincronizando com os marcos e os objetivos do projeto.]

3.3 Papéis e Responsabilidades

[Identificar papéis e responsabilidades em relação à realização das tarefas de Garantia de Qualidade.]

4. Documentação

[Relacionar a documentação utilizada e produzida pelo projeto no processo de desenvolvimento. A documentação mínima requerida é composta pelos seguintes artefatos:

- *Visão*
- *Especificação de Requisitos de Software*
- *Especificação de Caso de Uso*
- *Realização de Caso de Uso*
- *Documento de Arquitetura de Software*

- *Lista de Riscos*
- *Plano de Desenvolvimento de Software*
- *Plano de Testes*
- *Plano de Gestão de Configuração]*

5. Padrões e Guias

[Referenciar padrões e guias (orientações) que são utilizados no projeto e que servirão como base para as avaliações de qualidade realizadas.]

6. Métricas

[Referenciar o Plano de Métricas (ou o Plano de Desenvolvimento de Software).]

7. Revisões de Software

[Especificar, para as atividades de qualidade, que revisões serão realizadas, qual o cronograma, os recursos e métodos utilizados na condução dessas atividades.]

8. Testes

[Referenciar o Plano de Testes.]

9. Relato de Problemas e Ações Corretivas

[Descrever as práticas para relatar, rastrear e resolver problemas ou tratar questões identificadas no processo e no produto de software.]

10. Ferramentas, Técnicas e Metodologias

[Identificar ferramentas, técnicas e métodos usados para suportar as atividades de Garantia de Qualidade.]

11. Gestão de Configuração

[Esta seção referencia o Plano de Gestão de Configuração, na medida em que deve indicar como os artefatos intermediários e que devem ser entregues ao cliente devem ser armazenados, mantidos e recuperados caso haja algum problema.]

12. Controle de Fornecimento

[Estabelecer os procedimentos para assegurar que o software desenvolvido pelos fornecedores atende os requisitos estabelecidos.]

13. Registros de Qualidade

[Identifica a documentação dos registros de qualidade que deve ser mantida ao longo do projeto, assim como e por quanto tempo cada artefato deve ser armazenado e retido para verificação.]

14. Treinamento

[especifica os treinamentos necessários para atender os objetivos de qualidade, tanto para a equipe de

qualidade, como para os stakeholders do projeto]

15. Gerenciamento de Riscos

[Esta seção referencia o Plano de Gerenciamento de Riscos, especificando como serão identificados, mitigados e contingenciados os riscos da fase do ciclo de desenvolvimento que é escopo do Plano de Garantia de Qualidade].

Projeto X
Relatório de Avaliação de Qualidade
Auditoria
Versão 1.0

Projeto X	Versão: 1.0
Relatório de Avaliação de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

Histórico de Revisão

Data	Versão	Descrição	Autor
Dd/mm/aaaa	1.0	*Criação de documento – registro sobre a 1ª. avaliação de qualidade do Projeto X na fase de Elaboração*	*Engenheiro de Qualidade*

Índice

Projeto X	Versão: 1.0
Relatório de Avaliação de Qualidade	Data: xx/xx/xxxx
<Identificador do documento>	

Relatório de Avaliação de Qualidade

1. Introdução

1.1 Propósito

Esse relatório tem por objetivo registrar os resultados da avaliação de qualidade 1, realizada no Projeto X, durante a Fase de Elaboração.

1.2 Escopo

Esse documento abrange informações sobre a avaliação de qualidade executada no Projeto X, durante a Fase de Elaboração. Corresponde à versão 1.2 do Plano de Garantia de Qualidade do Projeto X.

1.3 Definições, Acrônimos e Abreviaturas

GUI – Graphical User Interface ou Interface Gráfica de Usuário

SQA – Software Quality Assurance ou Garantia de Qualidade de Software

1.4 Documentos de Referência

Os documentos utilizados são referenciados a seguir:

- *Plano de Garantia de Qualidade do Projeto versão 2*
- *Especificação de Requisitos de Software*
- *Protótipo de GUI (Interface Gráfica de Usuário)*
- *Especificação de Casos de Uso*
- *Modelos UML do software e Realização de Casos de Uso*
- *Documento de Arquitetura de Software*
- *Plano de Testes*
- *Plano de Desenvolvimento de Software e respectivos Planos de Iteração*
- *Lista de Riscos*

2. Estrutura de Garantia de Qualidade

2.1 Organização

A estrutura de qualidade dentro da organização é descrita no Plano de Garantia de Qualidade. Há uma pessoa da equipe de qualidade alocada ao Projeto X, exercendo simultaneamente os papéis de Engenheiro de Qualidade e Gerente de Qualidade.

2.2 Técnicas utilizadas

A técnica utilizada para Garantia de Qualidade do projeto é a de Auditoria. A avaliação abrange os documentos referenciados no item 1.4 deste documento, além das atividades realizadas pelo projeto na Fase de Elaboração.

2.3 Fase e Iteração

A avaliação de Garantia de Qualidade do Projeto X descrita nesse relatório abrange a primeira etapa da Fase de Elaboração, iteração no. 2 do projeto.

3. Avaliação de Qualidade

3.1 Auditoria de Qualidade

3.1.1 Processos de Software

Fase de Iniciação		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender o que construir (determinar o escopo do sistema, para quem o sistema servirá e qual é a melhor solução).	• se a aplicação a ser desenvolvida (*software*), no artefato Visão, foi definida em termos de: - benefícios e oportunidades; - problemas resolvidos pela aplicação; - usuários-alvo; - necessidades e características. • se há descrição de casos de uso e atores; • se há descrição dos requisitos não funcionais da aplicação; • se os protótipos de interface foram gerados, se previstos;	• Visão • Especificação de requisitos de *software* • Especificação suplementar de requisitos • Especificação de caso de uso
2. Identificar casos de uso críticos.	• se as funcionalidades-chave (casos de uso críticos) foram identificadas e priorizadas.	• Especificação de requisitos de *software*
3. Determinar pelo menos uma solução possível.	• se foi identificada pelo menos uma solução arquitetural; • se, para cada solução arquitetural proposta: - a estratégia de evolução do *software* foi estabelecida; - as tecnologias a serem utilizadas foram identificadas e os riscos associados a sua utilização; - decisões relativas à aquisição ou desenvolvimento interno foram consideradas, e os riscos associados a essas decisões foram especificados; - os componentes a serem utilizados (database, middleware) foram considerados em termos de aquisição, reuso, custos de aquisição e se os riscos associados a essas decisões foram especificados; • se o protótipo funcional (se existir) foi avaliado pela equipe de testes. • Se a estratégia de testes foi definida.	• Documento de arquitetura de *software* • Lista de riscos • Realizações de caso de uso • Plano de Testes (Estratégia de Testes) • Sumário de Avaliação de Testes
4. Estimar custos, cronograma e riscos associados ao projeto.	• se foram especificadas informações de: - estimativas; - recursos; - cronograma; - custo do projeto; - listas de riscos.	• Caso de Negócio • Lista de Riscos • Plano de Desenvolvimento de S*oftware*
5. Decidir que processo seguir e que ferramentas utilizar.	• se foram especificadas informações de: - processo definido para o projeto; - ferramentas a serem utilizadas; - *templates* e artefatos. • Se o ambiente (processo, ferramentas, *template*s) foi preparado para uso pelo projeto.	• Caso de Desenvolvimento • Plano de Desenvolvimento de S*oftware*

Fase de Elaboração		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Compreender requisitos detalhadamente.	• se os requisitos funcionais foram descritos e detalhados e representados em especificações de casos de uso; • se todos os casos de uso arquiteturalmente significativos foram detalhados; • se há evidências de que os casos de uso foram testados; • se o Glossário está sendo atualizado; • se mudanças nos requisitos são gerenciadas e rastreadas.	• Especificação de requisitos de *Software* • Especificação de caso de uso • Realização de caso de uso • Glossário • Registros de revisão
2. Projetar, implementar e validar a arquitetura, estabelecendo uma *baseline* de arquitetura.	• se o sistema foi definido em termos de blocos de construção, identificando: - subsistemas e componentes; - interfaces entre os componentes; - decisões e riscos de implementação dos componentes (construção, aquisição, ou reuso); - descrição da interação entre os componentes ao longo dos cenários. • se há evidências da realização dos casos de uso, com as devidas correspondências entre as classes de análise e de *design*; • se há evidências de que as classes foram empacotadas de forma apropriada. • se há evidências da descrição de modelos abrangendo concorrência, processos e *threads*. • Se há modelo de dados do sistema, e se foi identificada a estratégia de armazenamento e recuperação de dados. • se foram utilizados na solução os mecanismos arquiteturais padrões da organização adequados aos problemas. • se há evidências de que o protótipo foi implementado, contemplando cenários críticos. • se há evidências de que houve revisão do código implementado, e se houve teste de código, com implementação de classes de teste pelos desenvolvedores. • se há evidências de que o integrador combinou as partes adequadas (componentes e subsistemas) do sistema; • se há evidências de que foram realizados testes do protótipo (*build*) pela equipe de testes, segundo o Plano de testes.	• Documento de Arquitetura de *software* • Realizações de caso de uso • Especificações de requisitos de *software* • Especificação de caso de uso • Plano de testes • Relatório de testes • Modelo de dados • Modelos UML • Registros de revisão • Plano de integração • Sumário de avaliação de testes
3. Mitigar riscos e produzir estimativa acurada de custo e cronograma	• se os dados do plano de projeto e estimativas de custo foram atualizados.	• Plano de Desenvolvimento de *Software* • Lista de Riscos
4. Refinar Caso de Desenvolvimento e ambiente de desenvolvimento.	• se, no artefato Caso de Desenvolvimento, foram identificadas melhorias e adaptações no uso de processos e ferramentas; • se o ambiente de desenvolvimento foi ajustado para a iteração.	• Caso de Desenvolvimento

Fase de Construção		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Minimizar custos de desenvolvimento e atingir algum grau de paralelismo.	• se há evidências nos modelos UML do projeto de que as responsabilidades do sistema foram divididas em subsistemas bem definidos; • se os artefatos do projeto encontram-se sobre controle de versão e sob gestão de configuração; • se há evidências de que é realizado o planejamento da integração, identificando que determinada versão do *software* é composta por determinados componentes ou subsistemas definidos para o sistema. • se os mecanismos arquiteturais planejados foram considerados entre as opções de projeto; • se há evidências de que os desenvolvedores testam o código (teste de unidade).	• Modelos UML do projeto • Realização de Caso de Uso • Plano de Gestão de Configuração • Mecanismos Arquiteturais • Sumário de Avaliação de Testes
2. Desenvolver um produto completo que esteja pronto para ser entregue ao cliente.	• se há evidências de que todos os casos de uso remanescentes foram projetados, implementados e testados; • se houve revisão do *design* dos Casos de Uso remanescentes; • se todos os cenários dos Casos de Uso foram projetados, implementados e testados. • se as mudanças relacionadas a banco de dados abrangem apenas criação de colunas, de visões e de índices. • são realizados testes de integração e testes sistêmicos; • se há evidências de que são realizados testes de usuários; • se há evidências de que são preparados os artefatos para implantação do *software*.	• Realização de caso de uso • Modelos UML • Especificação de Requisitos de Software • Casos de uso • Registro de Revisão • Plano de Testes • Casos de Testes • Sumário de Avaliação de Testes • Solicitações de mudanças de Bancos de Dados • *Build* • Instruções de Instalação • Manuais de Usuários • Material de Treinamento

Fase de Transição		
Objetivos da Fase	**Escopo das atividades de Gerenciamento da Qualidade** (A avaliação de qualidade verifica:)	**Artefatos avaliados**
1. Realizar testes beta.	• se os usuários realizaram testes de avaliação do *software* em sua versão beta; • se solicitações de mudança do usuário foram implementadas antes de implantar a versão final do *software*; • se solicitações de mudança para correção de defeitos foram implementadas; • se foram realizados testes de regressão pelo desenvolvedor e pela equipe de testes; • se a documentação do *software* a ser utilizada na implantação foi desenvolvida; • se os testes de sistema planejados foram realizados; • se os testes de aceitação foram realizados;	• Sumário de avaliação de testes • Relatórios de defeitos • Solicitações de mudanças • *Help On line* • Material de treinamento • Manual do usuário • Manual de instalação

	• se *patch releases* foram produzidos para corrigir defeitos significativos.	
2. Treinar usuários.	• se o material de treinamento foi concluído; • se a documentação do usuário foi concluída; • se os manuais operacionais foram preparados e concluídos; • se houve treinamento para os usuários.	• Registros de treinamento • Material de treinamento • Manual de instalação • Manuais do usuário
3. Preparar implantação e converter bancos de dados operacionais.	• se houve necessidade de migração de dados e se ela realmente aconteceu; • se houve necessidade de que o sistema anterior (se for o caso) e sistema novo executarem em paralelo, e se foram registradas análises relacionadas a isto; • se o hardware novo necessário foi adquirido e instalado.	• Documento de Arquitetura de *Software* • Plano de Implantação
4. Preparar empacotamento.	• Se foi definida a estrutura de empacotamento do *software* e se ela está completa.	• Empacotamento de *software* • Plano de Implantação
5. Avaliar as *baselines* de implantação.	• se foram realizados testes de aceitação do produto, considerando a completude das *baselines* de implantação.	• Sumário de Avaliação de Testes • Plano de Gerenciamento de Configuração
6. Registrar lições aprendidas.	• se a Revisão *Post Mortem* foi realizada; • se o artefato Caso de Desenvolvimento foi atualizado com base nas lições aprendidas do projeto.	• Registros de Revisão • Caso de Desenvolvimento

3.1.2 Artefatos

Artefato	NC (não conformidade)	Severidade	Ação Corretiva	Prazo	Responsável	Ação Preventiva	Data de Abertura	Data de Fechamento
1. Visão	-	-	-	-	-	-	-	-
2. Caso de Negócios	Não se aplica	-	-	-	-	-	-	-
3. Glossário	*Documento incompleto, com termos não definidos. Não está sendo atualizado quando necessário.*	*média*	*Completar as definições dos termos no documento*	*16/04/2005*	*Analista*	-	*14/04/2005*	
4. Regras de Negócios	*Informações registradas em documento diferente do que deveria ser utilizado. Informações de glossário desatualizadas.*	*média*	*Separar informações sobre a execução da funcionalidade das informações das regras de negócio. Atualizar terminologia.*	*22/04/2005*	*Analista*	-	*14/04/2005*	
5. Especificação de Requisitos de Software	-	-	-	-	-	-	-	-
6. Especificação Suplementar	*Requisitos suplementares não especificados.*	*média*	*Especificar os requisitos suplementares*	*08/04/2005*	*Analista*	-	*06/04/2005*	*08/04/2005*
7. Especificação de Caso de Uso	*Dificuldades na especificação de Casos de Uso. Existência de informação sobre interface gráfica na especificação do caso de uso. Regras de Negócio descritas apenas na Especificação de Casos de Uso.*	*alta*	*Melhorar a Especificação de Caso de Uso, colocando informações de interface gráfica no protótipo de GUI e de regras de negócio no documento de regra de negócio.*	*20/04/2005*	*Analista*	-	*14/04/2005*	

8. Lista de Riscos	*Documento identifica apenas riscos associados a Casos de Uso, não identificando outros riscos do projeto.*	*Media*	*Identificar demais riscos do projeto e estabelecer estratégias de mitigação e de contingenciamento para todos os riscos identificados.*	*16/04/2005*	*Gerente de Projeto*	-	*14/04/2005*	
9. Plano de Desenvolvimento de Software	*Ausentes as estimativas.*	*Média*	*Fazer as estimativas para o projeto*	*16/04/2005*	*Gerente de Projeto*	-	*14/04/2005*	*Registrada como desvio com aceite dos stakeholders em 16/04/2005*
10. Documento de Arquitetura de Software	*1.Não foram descritas ou realizadas avaliações sobre possíveis sugestões de arquitetura.* *2. Documentação de arquitetura incompleta, com ausência de diagramas UML importantes. Apresentou os seguintes problemas: a visão lógica UML foi confundida com as regras de negócios; a visão de processos UML foi confundida com o documento Visão do projeto; a visão de implantação UML não foi especificada adequadamente.*	*Alta*	*Descrever a arquitetura de acordo com o modelo 4+1 da UML.* *Descrever as possíveis sugestões de arquitetura e registrar as avaliações realizadas.*	*20/04/2005*	*Arquiteto*	-	*14/04/2005*	
11. Realização de	-	-	-	-	-	-	-	-

Caso de Uso								
12. Modelos UML	*Ausência de diagramas de componentes e diagramas de implantação UML*	*média*	*Especificar os diagramas UML referenciados.*	*20/04/2005*	*Arquiteto*	-	*14/04/2005*	*20/04/2005*
13. Modelo de Dados	-	-	-	-	-	-	-	-
14. Manual do Usuário	-	-	-	-	-	-	-	-
15. Manual de Instalação	-	-	-	-	-	-	-	-
16. Plano de Implantação	-	-	-	-	-	-	-	-
17. Solicitação de Mudanças	-	-	-	-	-	-	-	-
18. Plano de Testes	-	-	-	-	-	-	-	-
19. Sumário de Avaliação de Teste	-	-	-	-	-	-	-	-
20. Registros de Revisão	-	-	-	-	-	-	-	-
21. Plano de Gestão de Configuração	-	-	-	-	-	-	-	-
22. Help On line	-	-	-	-	-	-	-	-
23. Material de Treinamento	-	-	-	-	-	-	-	-

Observação: o campo não conformidade é preenchido com a informação "não se aplica" quando o artefato não é considerado na avaliação de qualidade.

4. Resultados da Avaliação de Qualidade

Após realizada auditoria no projeto, os principais resultados identificados foram:

- *Requisitos não funcionais não foram especificados.*
- *Decisões sobre sugestões de arquitetura não foram documentadas.*
- *A arquitetura não foi descrita seguindo a visão 4+1 de modelos UML.*
- *Houve dúvidas, por parte da equipe do projeto, sobre a documentação a ser utilizada nessa fase, especialmente em relação ao nível de detalhamento das informações a ser feito em cada iteração. Não houve uma identificação clara do ponto em que terminava a atuação do analista e onde começava o papel do arquiteto na definição da estrutura do sistema.*
- *Houve dúvidas sobre planejamento iterativo do projeto e identificação de riscos para o planejamento das iterações.*
- *Foi registrado um desvio para o projeto, abrangendo as estimativas. Como os casos de uso não estão sendo bem especificados, o projeto se depara com a dificuldade adicional de conseguir estimar com base em casos de uso mal estruturados.*
- *Há necessidade de treinamento em modelos UML para o projeto, especialmente em relação à especificação de Casos de Uso*
- *Deve ser intensificado o acompanhamento do projeto em relação aos riscos, pois o mesmo apresenta dificuldades em identificar riscos e estabelecer as estratégias de mitigação e contingência.*

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo