**CENTRO UNIVERSITÁRIO EUROAMERICANO – UNIEURO PRÓ-REITORIA E PÓS-GRADUAÇÃO, PESQUISA E EXTENSÃO COORDENAÇÃO DE PÓS-GRADUAÇÃO LATO SENSU**

**CURSO DE MBA EMENGENHARIA DE SOFTWARE**

**MURILO SILVA ANDRADE SOUZA**

**NAYARA SOUZA DUARTE**

**ROSANA OLIVEIRA RAMOS DA COSTA**

QUALIDADE DE SOFTWARE COM O RUP

**Brasília, Abril / 2014**



**MURILO SILVA ANDRADE SOUZA**

**NAYARA SOUZA DUARTE**

**ROSANA OLIVEIRA RAMOS DA COSTA**

**QUALIDADE DE SOFTWARE COM O RUP**

**Trabalho de Conclusão de Curso – Monografia, apresentada como pré-requisito parcial para conclusão do curso de MBA em Engenharia de Software do Centro Universitário EUROAMERICANO – UNIEURO.**

**Orientadora: Prof.ªMSc. Edna Dias Canedo**

**Brasília**

**Abril, 2014**

Ficha catalográfica elaborada pela BibliotecáriaFulana de tal, CRB1/0000

M672Miranda, José Luis Carneiro de.

Como escrever umartigo científico/ José Luís Carneiro de

Miranda; Brasília : UNIEURO, 2007. vii, 27p. : il.

Monografia(Pós-graduação)–CursodeEspecializaçãoemGestão de Segurança em Redes de Computadores. Centro Universitário Euroamericano.

1.Palavrachave. 2.Palavrachave3.Palavrachave4. Palavrachave.5.Palavrachave.I. Silva,RosanaPioda, (Orientadora) II. Título.

CDU: 000000000000

**MURILO SILVA ANDRADE SOUZA**

**NAYARA SOUZA DUARTE**

**ROSANA OLIVEIRA RAMOS DA COSTA**

**QUALIDADE DE SOFTWARE COM O RUP**

Esta monografia foi julgada adequada à obtenção do grau de Especialista em Engenharia de Softwareeaprovadaemsuaformafinalpelocurso deMBAemEngenhariade Software do CentroUniversitário EUROAMERICANO– UNIEURO.

Data de aprovação:

Banca Examinadora

Profª. Msc. Edna Dias Canedo - Orientadora

Centro Universitário UNIEURO

ProfºMsc. Cleber Machado Ortiz

Centro Universitário UNIEURO

ProfºMsc. Leôncio Regal Dutra

Centro Universitário UNIEURO

**RESUMO**

A qualidade está cada vez sendo mais difundida dentro das grandes empresas, com propósitos de melhorar sua gestão e as negociações com seus clientes. Em questão, estaremos abordando como atingir qualidade no desenvolvimento de software utilizando como processo de engenharia o RUP. Confiabilidade, segurança, robustez e práticas deste processo de engenharia serão exemplificadas para fácil entendimento.

Palavras-Chaves: Qualidade de Software, RUP,Processo Unificado Rational

**ABSTRACT**

The quality is increasingly being more widespread within large companies, for purposes of improving its management and negotiations with customers. Concerned, we will be addressing how to achieve quality in software development using engineering process as RUP. Reliability, security, robustness and practices of this engineering process will be exemplified for easy understanding.

Key Words: Software Quality, RUP, Rational Unified Process..

**SUMÁRIO**

1 INTRODUÇÃO 2

1.1 O que é qualidade? 2

1.2 Desenvolvimento de Software 2

1.3 Aspectos importantes das definições de qualidade de software 4

1.4 Qualidade de software com RUP 5

1.5 Os Princípios do RUP 6

**Índice de Tabelas**

**Nenhuma entrada de índice de ilustrações foi encontrada.**

**Índice de Figuras**

Figura 1.1 – Princípios do RUP 7

# INTRODUÇÃO

## O que é qualidade?

Qualidade é um termo que pode ter diferentes interpretações. Pode ser definida pela perspectiva do produtor e pela perspectiva do cliente. Pela perspectiva do produtor, qualidade significa conformidade com as especificações (ou a ausência ou a variação dessas especificações). Um fabricante de relógios de pulso, por exemplo, poderia incluir uma especificação de confiabilidade que requeira que 99,995% dos relógios não atrasarão ou adiantarão mais de um segundo por mês. Testes simples permitirão ao fabricante medir precisamente essas especificações.(LAUDON,2001).

Philip Crosby [CRO79], em seu destacado livro sobre qualidade, discute essa situação:

O problema da administração da qualidade não é aquilo que as pessoas não sabem sobre ela. O problema é aquilo que elas sabem...

Nesse sentido, a qualidade tem muito em comum com o sexo. Todos se interessam por ele (sob certas condições, é claro). Todos acham que o entendem. (Ainda que não queiram explicá-lo.) Todos acham que a execução é apenas uma questão de seguir as inclinações naturais. (Afinal de contas, já nos saímos bastante bem!) E, é claro, a maioria das pessoas acha que os problemas nessas áreas são causados por outras pessoas. (Bastava que elas tivessem tempo para fazer as coisas corretamente.) – Engenharia de Software – Pressman, Roger S.pág. 723.

Qualidade de uma forma geral é a soma das características e critérios de um produto ou serviço que executam suas práticas para satisfazer às partes envolvidas.

Segundo a definição de Pressman, qualidade de software é a conformidade a requisitos funcionais e de desempenho que serão claramente declarados, e expostos a padrões de desenvolvimento claramente documentados, e a características implícitas que são esperadas de todo software desenvolvido por profissionais, ou seja, um software que atenda as necessidades do cliente com a capacidade de concentrar os requisitos funcionais e eficiência de forma harmônica.

## Desenvolvimento de Software

Focando no desenvolvimento de software, Ian Sommerville define um processo de software como um conjunto de atividades que leva à produção de um produto de software (SOMMERVILLE, 2007). Roger S. Pressman define processo de software como um arcabouço para as tarefas que são necessárias para construir software de alta qualidade (PRESSMAN, 2006). Wilson de Paula Filho faz uma analogia interessante, para ele processo é uma receita a ser seguida (PAULA FILHO, 2009).

Processos de softwares são complexos e como todos os processos intelectuais e criativos dependem de julgamento humano. A existência de um processo de software não garante que o software será entregue no prazo, de que ele irá satisfazer as necessidades do cliente, ou exibirá os atributos arquiteturais que manterão as características de qualidade em longo prazo. Um processo deve ser acoplado a uma sólida prática de engenharia de software e deve ser avaliado para garantir que satisfaça a um conjunto de critérios básicos de processo que demonstram ser essenciais para uma engenharia de software bem sucedida (PRESSMAN, 2006).

A engenharia de software surgiu num contexto onde a crise de software se apresentava e, decorrente disso, sistemática de trabalho mais consistentes e formais, inspiradas na engenharia, foram concebidas para solucionar os problemas que tendiam ser, cada dia, maiores e mais complexos e que acompanhavam a comunidade de desenvolvedores ao longo dos anos, de forma crônica (Pressman, 1995).

A partir deste cenário, surgiu a necessidade de tornar o desenvolvimento de Software como um processo planejado e padronizado, para que as necessidades fossem atendidas e os gastos controlados.

Entretanto, a Crise do Software ainda perdura, onde, mesmo depois de anos, ainda existem características da época da crise, como projetos atrasados, erros de estimativa de custos e de tempo, que tornam o processo, ainda que sistematizado, passível de muitos erros.

A maior prioridade é satisfazer o cliente, a partir de entregas de produtos de software de efetivo valor em tempo hábil e continuamente; · Acatar as necessidades de mudanças em qualquer estágio do processo de desenvolvimento, pois o software deve prover efetiva vantagem competitiva ao cliente.

A engenharia de software surgiu com o intuito de solucionar problemas que, pela sua magnitude, foram denominados de crise de software. Independentemente de ser uma crise ou problema crônico, o fato é que as dificuldades persistem (Pressman, 2002).

Para chegar a tal ponto, os processos foram desenvolvidos, porém nunca um deles trará satisfação total somada com qualidade inquestionável, seja em qualquer parte do projeto ou iteração de um processo.

## Aspectos importantes das definições de qualidade de software

Os requisitos de software são a base a partir da qual a qualidade é medida. A falta de conformidade com os requisitos significa falta de qualidade;

Um requisito de software descreve o que é requerido para que o sistema cumpra o seu objetivo.

Sommerville (2003), Engenharia de Requisitos e o processo de descobrir, analisar,documentar e verificar as funções e restrições do sistema.

Para que os requisitos do software sejam atingidos é necessário, definir o que o sistema deve fazer, e não como ele deve ser implementado, são organizados e acordo com os diferentes subsistemasque constituem o sistema.

Para estabelecer o que o sistema deve fazer é necessário que dentro do processo de desenvolvimento de sistemas, a atividade engenharia de requisitosproduz um documento que retrata de forma geralo que o sistema deve fazer. Segundo Sommerville(2003) compreender a natureza dos problemas pode ser muito difícil, especialmente se o sistema for novo.

Consequentemente é difícil estabelecer com exatidão o que o sistema vai fazer. As descrições das funções e das restrições são os requisitos para o sistema; e o processo de descobrir, analisar, documentar e verificar essas funções e restrições é chamado de engenharia de requisitos. Pressman(2006)

Outro aspecto de definições que devem ser verificados é a definição de um conjunto de critérios de desenvolvimento que orientam a maneira segundo a qual o software passa pelo trabalho de engenharia. Caso os critérios não sejam seguidos, o resultado quase que seguramente será a falta de qualidade.

Para que um software seja desenvolvido de forma consistente, é preciso aliar boas práticas da engenharia de software com um robusto e eficiente processo de desenvolvimento. Diferentes tipos de sistemas necessitam de diferentes processos de desenvolvimento. Por exemplo, um software de tempo real de uma aeronave deve ser completamente especificado antes do inicio do desenvolvimento, enquanto que um sistema de comércio eletrônico a especificação e o desenvolvimento do software podem ser conduzidos paralelamente. O uso de um processo de software inadequado pode reduzir a qualidade ou a utilidade do produto de software a ser desenvolvido e/ou aumentar os custos de desenvolvimento. Este fato leva as organizações que produzem software a usar processos de desenvolvimento que sejam eficientes e que atendam plenamente suas necessidades (SOMMERVILE, 2007).

Existe um conjunto de requisitos implícitos que freqüentemente não são mencionados na especificação, são os requisitos não funcionais que Pressman define como: Fatores de qualidade de software que podem ser medidos de forma indireta, ou como:  Características implícitas que são esperadas de todo software profissionalmente desenvolvido.Por exemplo, o desejo de uma boa Integridade no acesso ao Sistema.

## Qualidade de software com RUP

A comunidade de desenvolvimento de software tem enfrentado grandes dificuldades para entregar software de qualidade, no prazo contratado, com todos os requisitos contemplados e de acordo com o orçamento aprovado. A busca de novas abordagens tem caracterizado a engenharia de software que busca a melhora dos processos e produtos de software que tem no modelo clássico, uma das abordagens mais difundidas na década de 70. No final da década de 90, surgem os métodos denominados ágeis, que tiveram grande impulso na sua difusão a artir de 2001, com a publicação do manifesto ágil. Esses métodos caracterizam-se por valores, princípios e práticas que se orientam por uma perspectiva diferente da abordagem tradicional. Enquanto a abordagem tradicional dá ênfase para o processo e documentação rigorosa, a abordagem ágil foca pessoas, iterações curtas com entrega de produtos e leveza do processo.

O RUP (RationalUnifiedProcess)é um processo de desenvolvimento de software que possui um conjunto completo de atividades que define quem faz o que, quando e como.

Ele usa uma abordagem de orientaçãoa objetos em sua concepção e é projetado e documentado utilizando a notação UML para ilustrar os processos em ação.

Suas características principais: Iterativo e Incremental, que corresponde um meio de dividir grandes projetos em projetos menores ele foi elaborado para superar as complexidadesdeterminada pelo modelo cascata, que é utilizado com muito sucesso em projetos pequenos, onde o problema é bem conhecido e controlável.

O sistema é dividido em pequenas partes que são inteiradas. Essa iteração segue o formato sequencial tradicional, a cada interação é identificada as necessidades, analisada, projetada, implementada e testada, desta forma o sistema será incrementado até que todo o ciclo de desenvolvimento esteja finalizado. Esse modelo iterativo e incremental tem sido bem aceito e utilizado por várias metodologias de desenvolvimento de software.

É um modelo de processo moderno derivado também do Processo Unificado de Desenvolvimento de Software associado. Normalmente descrito a partir de três perspectivas:

• Uma perspectiva dinâmica que mostra as fases ao longo do tempo;

• Uma perspectiva estática que mostra atividades de processo;

• Uma perspectiva prática que sugere boas práticas. (Ian Sommerville, 2006, Engenharia de Software, 8ª. edição.)

O objetivo do RUP é assegurar uma produção de alta qualidade de software, que realiza a necessidade do usuário seguindo prazos e orçamento.

Conforme [Kruchten 2003], um produto de qualidade deve ter ausência de defeitos e, principalmente, deve atender aos propósitos desejados. Alguma coisa com qualidade deve fazer o que as pessoas querem que ela faça. Se alguma coisa é livre de defeitos, mas não faz o que as pessoas querem que ela faça, essa coisa é um produto desnecessário.

A qualidade de software não pode ser avaliada de maneira isolada. Softwares são desenvolvidos pelas organizações através de procedimentos. Um método pobre ou a ausência de uma metodologia pode ser a causa da baixa qualidade. Sendo assim, a avaliação da qualidade está diretamente relacionada com a qualidade de processos e metodologias utilizadas.

## Os Princípios do RUP

O IBM Rational Unified Process (RUP) é um framework de processo de engenharia de software que fornece um conjunto de práticas testadas na indústria para desenvolvimento de software e gerência de projetos (Shuja, 2007). Trata-se de um processo proprietário, desenvolvido pela Rational Software, atualmente subsidiária da IBM, que usa abordagem orientada a objetos e preconiza a utililização da notação UML (Unified Modeling Language) para documentação. É organizado em disciplinas (workflows) onde são distribuídas tarefas e responsabilidades e gerados produtos de trabalho (artefatos). O ciclo de vida é dividido em fases seqüenciais, as quais podem ser subdivididas em iterações.

Não existe uma fórmula exata para aplicação do RUP, isso dependerá de cada projeto e organização. Contudo algumas características diferenciam o RUP de outros métodos:

* Adaptação do processo:projetos possuem tamanhos de escopo diferentes exigindo uma constante adequação;
* Balanceamento das prioridades dos envolvidos: um grande desafio durante o desenvolvimento do software, mas garante alinhamento do negócio;
* Colaboração e comunicação entre as equipes: com os sistemas sendo arquitetado em um lado e o código sendo desenvolvido de outro, a comunicação efetiva e unificada é um fator de sucesso;
* Demonstração de valor iterativamente: para um processo de desenvolvimento orientado ao negócio, demanda que o time de projeto demonstre o retorno do investimento em uma abordagem repetitiva ao longo do projeto;
* Elevação no nível de abstração: uma abordagem de sucesso para sanar ascomplexidades é aprimorar o uso de padrões e serviços, reutilizando componentes em vez de construir o software do zero;
* Foco contínuo na qualidade: desde o levantamento de requisito e construção da documentação. A qualidade do sistema é afetada pela sua arquitetura. A inspeção de código e os testes unitários garantem que o código está de acordo com os padrões estabelecidos. Os testadores testam baseados nos cenários de casos de uso, ou com requisitos não funcionais. A responsabilidade de encontrar erros deve ser compartilhada entre todos da equipe.

Abaixo a Figura 1.1demonstra os princípios do RUP dentro de uma linha continua de processos.

Figura 1.1 – Princípios do RUP

Com a utilização de uma metodologia de desenvolvimento de software como o RUP, é possível obter:

* Qualidade de software;
* Produtividade no desenvolvimento, operação e manutenção de software;
* Controle sobre desenvolvimento dentro de custos, prazos e níveis de qualidade desejados;
* Estimativa de prazos e custos com maior precisão.