CENTRO PAULA SOUZA

Faculdade de Tecnologia da Zona Sul

**RUP**

**Rational Unified Process**

Orientador(a): Prof(a). Denise

São Paulo

2017

**RUP**

**Rational Unified Process**

Arthur Nunez Andrade

Dênisson Araújo Duarte

Donovan Sales de Souza

Janilo Gama dos Santos

José Andersson Pessoa de Vasconcelos

Leonardo Ferreira Pereira Sposito Santiago

Luciano Ribeiro do Nascimento

Vinícius Canuto dos Santos

Welker Barbosa

Trabalho apresentado a FATEC Zona Sul como parte dos requisitos para conclusão da disciplina de Engenharia de Software

Orientador(a): Prof. Denise

São Paulo

2017

Sumário

[Conceito 6](#_Toc493441275)

[1. História 7](#_Toc493441276)

[2. Metodologia 7](#_Toc493441277)

[2.1. Iterativo e incremental 7](#_Toc493441278)

[3. Ciclo de Vida 9](#_Toc493441279)

[3.1. Iniciação ou Concepção 10](#_Toc493441280)

[3.2. Elaboração 11](#_Toc493441281)

[3.3. Construção 12](#_Toc493441282)

[3.4. Transição 13](#_Toc493441283)

[4. UML 14](#_Toc493441284)

[4.1 Tipos de Diagrama 14](#_Toc493441285)

[5. Elementos 15](#_Toc493441286)

[5.1. Papéis e atividades 15](#_Toc493441287)

[5.2. Um papel, suas atividades e artefatos 16](#_Toc493441288)

[5.2.1 Equipe de Análise e Negócio 16](#_Toc493441289)

[5.2.2. Equipe de Desenvolvimento e Integração 17](#_Toc493441290)

[5.2.3. Equipe de Testes 17](#_Toc493441291)

[5.2.4. Equipe de Gestão 18](#_Toc493441292)

[5.2.5. Papéis Adicionais 19](#_Toc493441293)

[5.3 Disciplina 19](#_Toc493441294)

[5.4 Fluxo de Trabalho 20](#_Toc493441295)

[6. Processos 20](#_Toc493441296)

[6.1. Modelagem de Negócio 20](#_Toc493441297)

[6.2. Requisitos 20](#_Toc493441298)

[6.3. Análise e Design 21](#_Toc493441299)

[6.4. Implementação 21](#_Toc493441300)

[6.5. Testes 21](#_Toc493441301)

[6.6. Implantação 21](#_Toc493441302)

[6.7. Gerenciamento de Projeto 22](#_Toc493441303)

[7. Tamanho da equipe 22](#_Toc493441304)

[8. Estudo de Caso: Usando o IBM RUP como o quadro de metodologia 22](#_Toc493441305)

[Conclusão 25](#_Toc493441306)

[Referências Bibliográficas 27](#_Toc493441307)

**Introdução**

Este artigo tem como objetivo definir o conceito de Rational Unified Process (RUP), explicando a forma como é utilizado e todo o seu processo. Preliminarmente devemos conhecer a história e conceito, que serão expostos a seguir e no decorrer do trabalho, entender quais os métodos utilizados.

Com essa base pode-se estudar com melhor precisão a temática e assim esclarecer eventuais dúvidas.

É importante observar que a pesquisa realizada utiliza-se de método dedutivo e como técnica a pesquisa a bibliográfica.

# Conceito

O Rational Unified Process (RUP), é um processo de Engenharia de Software. Criado pela Rational Software Corporations, e adquirido em fevereiro de 2003 pela IBM, o RUP tornou-se referência na área de Software, fornecendo técnicas a serem seguidas a todos membros das equipes de desenvolvimento, afim de aumentar sua produtividade.

O principal objetivo do RUP é a garantia de produção de softwares de alta qualidade que atendam às necessidades dos usuários finais, cumprindo cronogramas e orçamentos previsíveis. Este processo permite ao cliente a possuir a visão de como o sistema será construído na fase de Implementação, através da geração do modelo de projeto, podendo também, opcionalmente, acompanhar um modelo de análise para garantir a robustez do software.

O Rational Unified Process tem como característica a especificação dos papéis de cada membro inserido no desenvolvimento do produto, e define como e quando as alterações deverão ser feitas, descrevendo todas as metas de maneira precisa para que sejam alcançadas a cada iteração e incrementação.

# 1. História

Sua concepção se deu no começo dos anos 80, pela Rational Software Corporation®. Fundada por Paul Levy e Mike Devlin, a Rational se dedicou a desenvolver com sucesso sistemas largos e complexos.

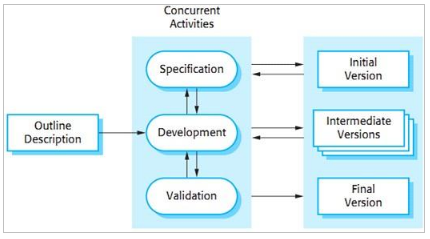
Antes do surgimento do RUP, o desenvolvimento de software era baseado em processos como: cascata e espiral, que encareciam o produto, pois não previam alterações de requisitos e dificultavam a comunicação da equipe.

O RUP surgiu em 1996 visando fortalecer a comunicação entre os envolvidos no projeto e organizar o desenvolvimento de software. Trazendo as características consideradas aproveitáveis dos modelos anteriores, mas agregando um conjunto de boas práticas e a modelagem do sistema.

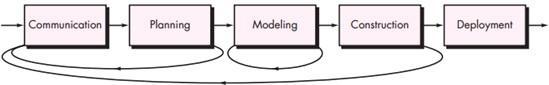
# 2. Metodologia

## 2.1. Iterativo e incremental

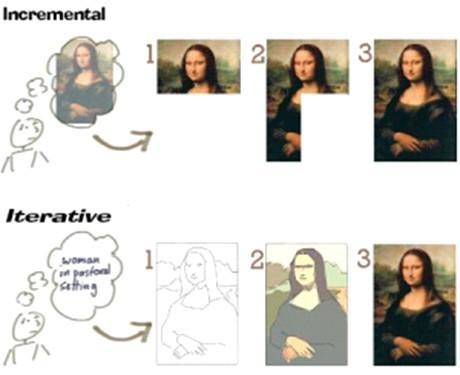
No modelo de processo de software incremental tem-se a ideia de se desenvolver uma implementação inicial, apresentá-la ao usuário e evoluir o software ao longo do tempo de acordo com o feedback do usuário. Especificação, desenvolvimento e validação ocorrem em paralelo para cada um dos incrementos do software que estão em construção. Sommerville apresenta o modelo incremental da seguinte forma:



No modelo iterativo, a cada iteração uma ou mais atividades são executadas antes de se iniciar uma nova atividade. Ou seja, o fluxo de execução do processo de desenvolvimento ocorre desta maneira:



O modelo iterativo muitas vezes é confundido com o modelo incremental. Um processo de desenvolvimento de software é incremental quando a cada rodada é desenvolvido um pedaço inteiro do software. Já no iterativo, a cada iteração, se avança no conhecimento do projeto, novos requisitos são eliciados e a arquitetura do software é revisada. No modelo incremental, necessariamente teremos entrega de software ao término de uma iteração, o que não é verdade para o modelo iterativo. Exemplo:



**I. Fonte:** [**http://itsadeliverything.com/revisiting-the-iterative-incremental-mona-lisa**](http://itsadeliverything.com/revisiting-the-iterative-incremental-mona-lisa)

Então, se o Leonardo da Vinci tivesse desenhado um pedaço do quadro da Monalisa de cada vez, o pintor teria realizado um desenvolvimento incremental. Mas, na verdade, ele começou desenhando um esboço, depois preencheu as áreas com tinta e depois pintou os detalhes, ou seja, ele desenvolveu iterativamente o quadro da Monalisa. Ele foi conhecendo e especificando os requisitos da Monalisa cada vez mais a cada iteração.

O detalhe é que os modelos iterativo e incremental não são excludentes. O RUP, por exemplo, é um processo de desenvolvimento de software iterativo e incremental.

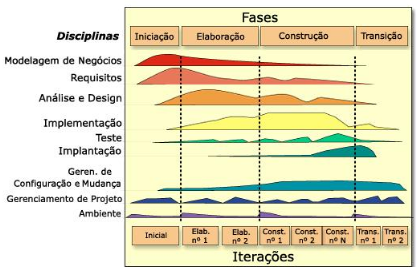
# 3. Ciclo de Vida

Entenda-se Ciclo de Vida de um sistema como o conjunto de atividades agrupadas em fases, que possuem um desenvolvimento sequencial, sendo colocadas em prática desde o início até a obsolência do sistema, podendo se repetir quantas vezes for necessária. No modelo de projeto RUP, a fim de melhor visualizar e dividir a dimensão do projeto e as ênfases necessárias em cada parte de seu ciclo de vida, divide-se o projeto em quatro fases, onde, em cada uma das fases, a dinâmica de produção e esforços humanos serão direcionados com mais destaque a determinado assunto, de acordo com a fase na qual se encontra o projeto.

As quatro fases do ciclo de vida de um projeto RUP são:

* Iniciação ou Concepção;
* Elaboração;
* Construção;
* Transição.

O tempo de duração de cada fase do ciclo de vida do projeto e o que se produz em determinada fase, é chamado de iteração. Os produtos de cada fase serão, posteriormente, usados na documentação do projeto, além de permitir uma melhor visualização e acompanhamento.



**Abordagem de todas as disciplinas durante as fases do projeto**

## 3.1. Iniciação ou Concepção

A fase de concepção, como o nome sugere, tem como principal objetivo, determinar todos os planos, características, necessidades e metas do projeto, ou seja, conceber o projeto desde seu início, estabelecendo todos os seus objetivos e expectativas do cliente, promovendo o consenso e acordo de todos os investidores do projeto, através de toda a documentação obtida nesta fase. A fase da iniciação pode ter uma duração mais longa ou menor, a depender do escopo do projeto. Se o objetivo é gerar um projeto de um sistema até então inexistente, esta fase tende a ser mais longa em comparação a um projeto que visa promover melhorias em um sistema já existente. Isto porque, um sistema que já existe e tem suas funcionalidades já definidas, esta fase tem como objetivo apenas assegurar que este seja compensatório e que sua realização seja possível e/ ou viável.

A fase de concepção tem como principais metas a delimitação das funcionalidades e viabilidade do projeto. Para isso, algumas ações são necessárias, vejamos:

* Estabelecer as funções do software e verificar as condições-limite para

sua realização, incluindo uma visão operacional para o futuro usuário e/

ou administrador;

* Critérios que definam o que deve ou não estar no sistema;
* Verificar e realizar a projeção dos possíveis erros do sistema e, ainda, as situações de uso crítico e demonstrar, pelo menos, uma opção básica de cenários e interface, que direcionarão as principais mudanças no design;
* Estimar o custo total e calcular os riscos gerais do projeto;

Para isso, deve-se estabelecer as metas, principalmente pesquisar e estudar o contexto no qual o sistema será incluído, gerenciar os riscos e minimizar custos.

A meta desta fase é ganhar a confiança do cliente, lhe apresentando a melhor solução para que suas expectativas sejam superadas. As palavras chaves desta fase são viabilidade e planejamento.

## 3.2. Elaboração

O principal objetivo desta fase é fornecer planos arquitetônicos do projeto, a fim de que se produza uma base estável para a fase da construção. Isto porque é, nesta fase, que a preocupação com os requisitos e avaliação dos riscos são mais importantes, uma vez que a partir do que for concluído nesta fase é que o projeto será construído. Primeiramente é realizado um exame dos requisitos mais significativos. Deve se certificar que a arquitetura que está sendo utilizada é estável o suficiente, que os riscos sejam eliminados e que a arquitetura suportará todos os requisitos do sistema. O preço e tempo de desenvolvimento.

As metas da fase de elaboração incluem:

* Focar em todas as soluções e riscos da arquitetura do sistema;
* Produzir um protótipo de design, com fins de demonstração do produto final do sistema;
* Estabelecer como será incluído o ambiente de suporte.

## 3.3. Construção

Aqui é o ponto do ciclo de vida do projeto onde este será programado, onde serão testadas as funcionalidades e verificadas as possíveis falhas do sistema, bem como, após todas as correções, este será concluído. A rapidez e eficiência são sempre metas a serem atingidas

Esta fase de construção é focada na programação, no gerenciamento de recursos, controle de operação e conclusão do projeto e, para isso, os seguintes passos serão observados:

* Minimização dos custos, otimização de recursos e melhor aproveitamento

do sistema;

* Realização do maior número de testes possíveis para assegurar que o software, os locais e usuários estão funcionando com qualidade para que o sistema seja implementado;
* Correção das falhas de sistemas verificadas nos testes, otimização das funções e melhorias de design.

A principal característica da fase de construção é a constatação em um ambiente de teste beta, após todos os testes e correções, de que o produto está pronto para a implantação.

## 3.4. Transição

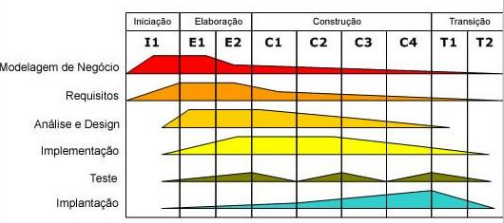
A fase de transição é marcada pela implementação do sistema para uso e teste diretamente pelo usuário. Isto permite que pequenas correções sejam feitas, baseadas nas necessidades e uso real do sistema pelo usuário final.

Nesta fase, o que mais importa é o *feedback* do usuário, que possibilitará o ajuste fino do sistema, personalizando-o de acordo com as expectativas do usuário após o uso de teste.

As principais atividades desta fase são:

* Entregar o software para que o usuário o teste e forneça um feedback;
* Realizar todas as correções necessárias e solicitadas pelo usuário, com o fim de fazer ajustes finos no produto, configuração, usabilidade e instalação;
* Treinamento dos usuários e da equipe de manutenção;
* Implementar e entregar definitivamente o produto.

O ponto alto da fase de transição é certificar se o produto está pronto para ser instalado e mantido pelo cliente. A principal preocupação é a implementação definitiva do sistema e, a partir daí, o ciclo de vira terá terminado.

**Fases do RUP**

# 4. UML

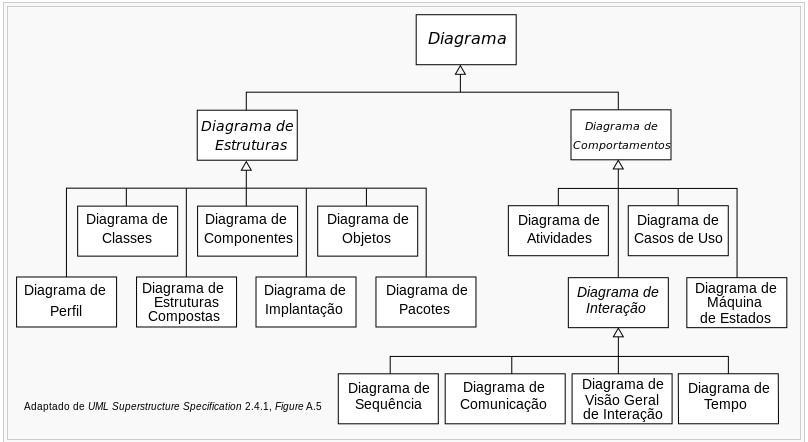
RUP usa a abordagem da orientação a objetos em sua concepção e é projetado e documentado utilizando a notação UML (Unified Modeling Language) para ilustrar os processos em ação.

O UML é uma linguagem para visualização, especificação, construção e desenvolvimento, a qual permite aos desenvolvedores uma melhor visão do produto de seus trabalhos em diagramas padronizados.

O RUP é dirigido por casos de uso, que são utilizados para capturar os requisitos funcionais e definir as tarefas de cada iteração. Cada iteração lida com um cenário ou conjunto de casos de uso durante todo o tempo.

## 4.1 Tipos de Diagrama

O UML possui 14 tipos de diagrama divididos em duas grandes categorias: Estruturais e Comportamentais. Sete tipos de diagramas representam informações estruturais, e os outros sete representam tipos gerais de comportamento, incluindo quatro em uma sub-categoria que representam diferentes aspectos de interação. Estes diagramas podem ser visualizados de forma hierárquica, como apresentado no padrão de diagrama de classes a seguir:



**IModelo adaptado de UML Superstructure Specification**

# 5. Elementos

RUP (Rational Unified Process) possui cinco elementos principais, sendo eles: papéis, atividades, artefatos, disciplinas e fluxos de trabalho.

## 5.1. Papéis e atividades

Um papel é uma definição abstrata de um conjunto de atividades executadas e dos respectivos artefatos.

Normalmente os papéis são desempenhados por uma pessoa ou um grupo de pessoas que trabalham juntas em equipe.

Um membro da equipe do projeto geralmente desempenha muitos papéis distintos.

Os papéis não são pessoas; pelo contrário, eles descrevem como as pessoas se comportam no negócio e quais são as responsabilidades que elas têm.

Apesar de a maioria dos papéis serem desempenhados por pessoas que fazem parte da organização, as pessoas de fora da organização têm um papel importante: por exemplo, o papel do envolvido do projeto ou produto que está sendo desenvolvido.

## 5.2. Um papel, suas atividades e artefatos

Os papéis têm um conjunto de atividades coerentes por eles executadas. Essas atividades são estreitamente relacionadas e combinadas em termos de funcionalidade, e é recomendável que elas sejam executadas pela mesma pessoa.

As atividades estão fortemente relacionadas aos artefatos. Os artefatos fornecem a entrada e saída para as atividades e o mecanismo pelo qual as informações são transmitidas entre as atividades.

### 5.2.1 Equipe de Análise e Negócio

Os papéis do Analista organizam os papéis envolvidos principalmente na identificação e na investigação de requisitos.

* **Analista de Sistemas**

O papel analista de sistemas lidera e coordena a identificação de requisitos e a modelagem de casos de uso, delimitando o sistema e definindo sua funcionalidade; por exemplo, estabelecendo  quais são os atores e casos de uso existentes e como eles interagem.

Uma pessoa que atua como analista de sistemas é um bom facilitador e possui habilidades de comunicação acima da média. É fundamental que os profissionais que desempenham este papel tenham conhecimento dos domínios do negócio e da tecnologia.

Entre os papéis desta equipe, podemos destacar outros como:

* Designer de Negócios;
* Revisor do Modelo de Negócios;
* Analista do Processo de Negócios;
* Revisor de Requisitos;
* Especificador de Requisitos;
* Designer de Interface de Usuário;
* Usuário Chave (Key User);

### 5.2.2. Equipe de Desenvolvimento e Integração

Os papéis do Desenvolvedor organizam os papéis envolvidos principalmente no design de integração e desenvolvimento de software.

* **Desenvolvedor**

O desenvolvedor de software é um profissional com conhecimento multidisciplinar, não apenas em informática, mas também em comunicação, gerência, negócios. Sua função principal é codificar de acordo com os requisitos técnicos especificados, no entanto, além de codificar consegue ter uma visão mais ampla do software que desenvolve, trabalhando intensamente com o design da aplicação.

Entre os papéis desta equipe, podemos destacar outros como:

* Designer de Cápsula;
* Revisor de Código;
* Designer de Banco de Dados;
* Implementador;
* Integrador;
* Arquiteto de Software;
* Revisor de Arquitetura;
* Revisor de Design;
* Designer;

### 5.2.3. Equipe de Testes

Os Papéis do Testador organizam os papéis que lidam com habilidades específicas exclusivas para teste. Observe que existem papéis adicionais envolvidos na disciplina Teste que juntos ampliam as habilidades básicas de outros papéis. Esses papéis adicionais podem ser encontrados nos demais papéis organizados pelas habilidades básicas que estendem (por exemplo, Gerente, Designer, Analista).

* **Designer de Teste**

O papel Designer de Teste é responsável por definir a abordagem de teste e assegurar sua correta implementação. O papel envolve identificar as técnicas, ferramentas e diretrizes apropriadas para implementar os testes necessários e dar orientação sobre os correspondentes requisitos de recursos para o esforço de teste.

Entre os papéis desta equipe, podemos destacar outros como:

* Testador (Tester);
* Analista de Teste;
* Gerente de Testes;

### 5.2.4. Equipe de Gestão

Os papéis dos gestores organizam os papéis envolvidos principalmente no gerenciamento e na configuração do processo de engenharia de software.

* **Gerente de Projeto/Sistemas**

A função de um gerente de projetos é planejar, controlar e executar projetos que podem ser pontuais, com uma data para começar e acabar, ou que podem ser a longo prazo, como o objetivo de negócio de uma empresa. Durante a execução de um projeto, define papéis, atribui tarefas, acompanha e documenta o andamento da sua equipe através de ferramentas e técnicas apuradas, administra investimentos e integra as pessoas para trabalharem juntas por um só objetivo. Monitora possíveis riscos e estar sempre preparado para mudar de estratégia rapidamente, se necessário.

Entre os papéis desta equipe, podemos destacar outros como:

* Engenheiro de Processo;
* Gerente de Controle de Mudança;
* Gerente de Configuração;
* Gerente de Implantação;
* Revisor do Projeto;
* Gerente de Testes;

### 5.2.5. Papéis Adicionais

Os papéis adicionais organizam os papéis que envolvem funções diversas ou de suporte no projeto. Tais como:

* Desenvolvedor do Curso
* Artista Gráfico
* Especialista em Ferramentas
* Administrador de Sistema
* Redator Técnico

## 5.3 Disciplina

O conjunto de atividades relacionadas formam uma disciplina. Dividir o projeto em disciplinas facilita a compreensão das atividades e do projeto como um todo, porém dificulta o planejamento das atividades.

O RUP utiliza nove tipos de disciplinas, sendo divididas em disciplinas de processo e disciplinas de suporte.

• As disciplinas de processo são: modelagem de negócios, requisitos, análise e projeto, implementação, teste e distribuição.

• As de suporte são: configuração e gerenciamento de mudanças, gerenciamento de projeto, e ambiente.

## 5.4 Fluxo de Trabalho

O Fluxo de trabalho tem como objetivo sequenciar atividades de forma lógica para obter um resultado valioso ao projeto.

Diagramas de sequência, diagramas de atividades e diagramas de colaboração são exemplos de formas representativas de fluxos de trabalho.

O RUP utiliza três tipos de fluxos de trabalho:

• Fluxos de trabalho principais, associados com cada disciplina;

• Fluxos de trabalho de detalhe, para detalhar cada fluxo de trabalho principal;

• Planos de iteração, que mostram como a iteração deverá ser executada.

# 6. Processos

## 6.1. Modelagem de Negócio

Definição de Modelagem de Negócios é uma técnica utilizada para compreensão dos processos dos negócios de uma empresa. A finalidade de modelar um negócio é criar uma abstração, que é uma visão simplificada do negócio.

## 6.2. Requisitos

Abrange a Modelagem do modelo genérico do processo, verificando quais são os requisitos para alcançar objetivo e analisar de forma mais detalhada a análise do domínio do problema, revisando os riscos que o projeto pode sofrer e a arquitetura do projeto começa a ter sua forma básica. Indagações como "O plano do projeto é confiável?", "Os custos são admissíveis?" são esclarecidas nesta etapa que se torna muito importante pois verifica a real viabilidade do projeto.

## 6.3. Análise e Design

Desenvolve ou adquire os componentes de Software. O principal objetivo desta fase é a construção do sistema de software, com foco no desenvolvimento de componentes e outros recursos do sistema.

## 6.4. Implementação

Caracteriza-se por implementar as classes do modelo de projeto em termos de componentes (código fonte ou executável, etc.).

## 6.5. Testes

O RUP propõe uma abordagem iterativa, o que significa que deve-se testar todo o projeto. Isto permite encontrar defeitos tão cedo quanto possível, o que reduz radicalmente o custo de reparar o defeito. Os testes são realizados ao longo de quatro dimensões da qualidade: confiabilidade, funcionalidade, desempenho da aplicação, e o desempenho do sistema.

## 6.6. Implantação

Implantação tem como objetivo produzir com sucesso lançamentos de produtos e entregar o software para seus usuários finais.

## 6.7. Gerenciamento de Projeto

A finalidade do gerenciamento de projetos é, fornecer um framework para gerenciar projetos intensivos de software, assim como diretrizes práticas para planejar, montar a equipe, executar e monitorar os projetos.

No entanto, esta disciplina do RUP, não tenta cobrir todos os aspectos do gerenciamento do projeto como:

• Gerenciamento de pessoal: contratação, treinamento, ensino;

• Gerenciamento de orçamento: definição, alocação etc.;

• Gerenciamento de contratos, com fornecedores e clientes.

# 7. Tamanho da equipe

O RUP é um processo considerado pesado e preferencialmente aplicável a grandes equipes de desenvolvimento e a grandes projetos, porém o fato de ser amplamente customizável torna possível que seja adaptado para projetos de qualquer escala. Para a gerência do projeto, o RUP provê uma solução disciplinada de como assinalar tarefas e responsabilidades dentro de uma organização de desenvolvimento de software.

# 8. Estudo de Caso: Usando o IBM RUP como o quadro de metodologia

* **Introdução:**

Este estudo de caso é uma análise das experiências reais sobre o processo de como a Ford e Ford Financial Credit (esta do grupo Ford) implantaram a metodologia iterativa utilizando o RUP como estrutura.

A Ford Financial Credit é uma das maiores financiadoras de automóvel, financiando mais de 11.000 revendedoras em mais de 40 países e tendo financiado mais de 50 milhões de veículos desde 1959, possuindo ao todo, mais de 700 profissionais de TI.

* **Motivação:**

Uma das principais motivações para a implementação da metodologia foi a dificuldade do grupo em compartilhar entregas e relatórios de seus projetos entre as equipes, pois cada equipe possuía seu próprio modelo de projeto e suas metodologias. Quando um membro migrava de equipe, era necessário que este aprendesse o novo método e, consequentemente, isso levava muito tempo, o que acabava atrasando o processo.

Havia também o problema de desvio das funções, onde, em um ambiente não padronizado, não havia funções específicas para cada membro e, por este motivo, os membros acabavam realizando funções fora de suas atribuições e especializações.

Outra razão que justificou a implantação desta metodologia foi a identificação tardia dos riscos do projeto, que acabava gerando perda de trabalho realizado e, por conta disto, perda de tempo na realização do projeto.

* **Proposta:**

O RUP foi escolhido porque um de seus pilares é a personalização, ou seja, é completamente possível que se customize o projeto de acordo com as necessidades da empresa.

Primeiramente, os responsáveis pela implantação identificaram os especialistas no assunto e, em seguida, criaram dois grupos: o primeiro realizaria a personalização do projeto de acordo com as características da empresa e, o segundo, implementaria o que foi desenvolvido.

* **Desenvolvimento:**

O primeiro passo do desenvolvimento foi identificar todos os detalhes para que o projeto fosse feito sob medida para a empresa. Nesta etapa foram colhidas todas as informações, as necessidades e as particularidades da empresa, inclusive o plano de carreira e funções.

Em seguida, foram identificados todos os artefatos que poderiam ser aplicados na empresa e quais os processos que seriam realmente úteis. Foram documentados os papéis e responsabilidades das equipes e, a partir disto, houve a criação de um material de apoio de leitura recomendada.

Na sequência do desenvolvimento, foi definida a organização de suporte e criado um web site de suporte, chamado *Unified Solution Delivery Methodology*, ou simplesmente*, Unified SDM*, onde há cerca de 20 artefatos, fluxos de trabalhos simplificados e equipes de apoio, inclusive serviços de *coaching.*

Em um segundo momento do desenvolvimento, foram ministradas palestras e um treinamento efetivo para todos os membros da empresa, desde o mais alto escalão até os funcionários de base.

# Conclusão

O RUP é uma metodologia que provê qualidade de software, aumento de produtividade e melhorias na manutenção além de ajudar no controle sobre todas as fases de desenvolvimento do software.

Essa ferramenta visa organizar o desenvolvimento do projeto antes da sua implementação, usando de métodos de correção de erros, de projetos ou levantamento requisitos, durante o desenvolvimento. Com isso a qualidade do mesmo é elevada a um nível superior aos projetos que não utilização esta ferramenta, pois todos os possíveis problemas que poderiam ser encontrados após o desenvolvimento já foram tratados e corrigidos.

Importante ressaltar que, apesar de ser uma metodologia aplicada para qualquer tamanho de projeto, o RUP não é vantajoso para projeto de pequeno porte, pois tem o custo elevado.

**Vantagens:**

* Processo robusto e bem definido com a geração de artefatos importantes: O RUP tem como base os princípios de engenharia de software refletidos na sua abordagem de desenvolvimento iterativa, incremental, orientada a requisitos e baseada em arquitetura;
* Os maiores riscos são atacados primeiro, diminuindo as chances de

fracasso do projeto;

* O RUP captura muitas das melhores práticas do desenvolvimento de software moderno, de forma que possam ser adaptadas para uma grande variedade de projetos e de organizações;
* Desenvolver iterativamente;
* Gerenciar Requisitos;
* Usa arquitetura baseada em componentes;
* Modelagem Visual;
* 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.

**Desvantagens**

* Complexo e trabalhoso para projetos de pequeno porte: Sério investimento em ferramenta de suporte;
* Limitações: Nas áreas de manutenção, gestão de métricas, gestão de pessoal, gestão de reutilização e testes. Exige experiência da equipe.

# Referências Bibliográficas

PRESSMAN, R. (2006). Engenharia de Software. 6ª ed.

KRUCHTEN, P. (2003). Introdução ao RUP - Rational Unified Process. 2ª ed.

MATINEZ, Marina. **RUP**. Disponível em: <<http://www.infoescola.com/engenharia-de-software/rup/>>

MATINEZ, Marina. **UMP**. Disponível em: <<http://www.infoescola.com/engenharia-de-software/uml/>>

Portal IBM Developer Works. Disponível em: <<http://mds.cultura.gov.br/core.base_rup/customcategories/construction_102AF1>>

Portal IBM Developer Works. Disponível em: <http://mds.cultura.gov.br/core.base\_rup/customcategories/transition\_1018F224.html

Portal Fundação da Universidade Federal do Paraná. **Conceitos: Modelo Básico do Rational Unified Process**. Disponível em: <<http://www.funpar.ufpr.br:8080/rup/process/workflow/environm/co_morop.htm>>

Portal Microsoft Developer Network. **Diagramas de classe UML: referência**. Disponível em:

<<https://msdn.microsoft.com/pt-br/library/dd409437.aspx>>

Portal Wikipedia. **IBM Rational Unified Process**. Disponível em: <<https://pt.wikipedia.org/wiki/IBM_Rational_Unified_Process#Disciplina_de_An.C3.A1lise_e_Projeto.28.22Design.22.29>>

BROERING, Evandro. **RUP - Rational Unified Process**. Disponível em: <<http://www.devmedia.com.br/rup-rational-unified-process/4574>>

Portal ProcessosdeSoftware. **Metodologia Rup**. Disponível em: <<https://procesosdesoftware.wikispaces.com/METODOLOGIA+RUP>>

Portal IBM Developer Works. Disponível em:

<https://www.ibm.com/developerworks/rational/library/content/03July/1000/12511251\_bestpractices\_TP026B.pdf>

WEI, John. **A Case Study: Using IBM Rational Unified Process as the Methodology Framework**. Disponível em:

<<http://www.ibm.com/developerworks/rational/library/4474.html>>