

UNIDADE 1 – METODOLOGIAS DE DESENVOLVIMENTO DE SISTEMAS

MÓDULO 1 – INTRODUÇÃO SOBRE METODOLOGIA DE DESENVOLVIMENTO DE SISTEMAS

01

1 - CONCEITUANDO METODOLOGIA E MÉTODO

Neste módulo aprenderemos os conceitos de metodologia, método, processo e todas as anuências sobre o tema, os quais são muito utilizados no dia a dia em toda a área da Tecnologia da Informação (TI). Veremos também a importância do uso de um processo definido e as equipes que apoiam o uso de metodologias e processos de desenvolvimento. Também mostraremos os problemas da não utilização de um processo para desenvolvimento do sistema, problemas esses que podem prejudicar o desempenho da empresa.

Os termos que estudaremos nos acompanharão em todo o estudo e em toda nossa vida no mundo das TIC (Tecnologia da Informação e Comunicação).

Começaremos com a definição da palavra **metodologia**, para chegarmos ao conceito de MDS (Metodologia de Desenvolvimento de Sistemas), que é nosso foco. Para um melhor entendimento, relacionamos abaixo as definições encontradas nos principais dicionários.

| | |
|--------------------|---|
| Michaelis → | 1. Estudo científico dos métodos. 2. Arte de guiar o espírito na investigação da verdade. 3. Filos. Parte da Lógica que se ocupa dos métodos do raciocínio, em oposição à Lógica Formal. M. didática: teoria dos procedimentos de ensino, geral ou particular para cada disciplina; didática teórica. |
| Aurélio → | Parte de uma ciência que estuda os métodos aos quais ela se liga ou de que se utiliza: metodologia linguística. |
| Priberam → | 1. Arte de dirigir o espírito na investigação da verdade. 2. Aplicação do método no ensino. |
| Wikipedia → | A Metodologia é o estudo dos métodos. As etapas a seguir num determinado processo. Tem como objetivo captar e analisar as características dos vários métodos indispensáveis, avaliar suas capacidades, potencialidades, limitações ou distorções e criticar os pressupostos ou as implicações de sua utilização. |

02

Além de ser uma disciplina que estuda os métodos, a metodologia é também considerada uma forma de conduzir a pesquisa ou um conjunto de regras para ensino de ciência e arte.

A metodologia é a explicação minuciosa, detalhada, rigorosa e exata de toda ação desenvolvida no **método** (caminho) do trabalho de pesquisa. É a explicação do tipo de pesquisa, dos instrumentos utilizados (questionário, entrevista etc.), do tempo previsto, da equipe de pesquisadores e da divisão do trabalho, das formas de tabulação e tratamento dos dados, enfim, de tudo aquilo que se utilizou no trabalho de pesquisa.

Observe que os dicionários tradicionais tratam metodologia como o “estudo e aplicação dos métodos”. Já o site Wikipedia traz uma definição mais próxima do que estamos buscando para o termo Metodologia de Desenvolvimento de Sistema.

A metodologia é o estudo dos métodos. As etapas a seguir num determinado processo. Tem como objetivo captar e analisar as características dos vários métodos indispensáveis, avaliar suas capacidades, potencialidades, limitações ou distorções e criticar os pressupostos ou as implicações de sua utilização.

03

A palavra **método** seria o que, então? Vejamos o que dizem os principais dicionários:

| | | |
|------------------|---|---|
| Michaelis | → | Conjunto dos meios dispostos convenientemente para alcançar um fim e especialmente para chegar a um conhecimento científico ou comunicá-lo aos outros. 2 Ordem ou sistema que se segue no estudo ou no ensino de qualquer disciplina. |
| Aurélio | → | Maneira de dizer, de fazer, de ensinar uma coisa, segundo certos princípios e em determinada ordem. / Maneira de agir. / Obra que reúne de maneira lógica os elementos de uma ciência, de uma arte etc. / Botânica. Modo de classificar as espécies vegetais. / Filosofia. Marcha natural e racional do espírito para a verdade: método cartesiano. |
| Priberam | → | 1. Ordem pedagógica na educação. 2. Tratado elementar. 3. Processo racional para chegar a determinado fim. 4. Maneira de proceder. 5. Processo racional para chegar ao conhecimento ou demonstração da verdade. 6. Obra que contém disposta numa ordem de progressão lógica os principais elementos de uma ciência, de uma arte. |

Temos agora as acepções das palavras **metodologia** e **método**.

De modo geral, podemos dizer que metodologia é o campo que estuda os melhores caminhos e procedimentos para se chegar ao resultado final esperado. Entrando no mundo da TI, temos a **Metodologia de Desenvolvimento de Sistemas (MDS)**, que consiste num conjunto de estudos de melhores práticas para o desenvolvimento de sistemas.

04

Podemos então definir um provável conceito para **Metodologia de Desenvolvimento de Sistemas** (MDS), que seria:

Conjunto de conhecimentos que incluem fases, procedimentos, regras, técnicas, ferramentas, documentação e treinamentos, gerenciados a fim de desenvolver um sistema ou o conjunto de métodos gerenciados a fim de desenvolver um sistema.

Segundo Steven Gordan e Judith Gordan,

uma **metodologia** é um conjunto prescrito e documentado de práticas, ferramentas, documentos, relatórios e frequentemente, anotações.

Existe uma crença por trás da metodologia, uma filosofia, onde possuem valores, princípios e práticas para utilização das metodologias.

Muitas empresas criam suas metodologias próprias com base em técnicas conhecidas no mercado. O mercado oferece pacotes comerciais de *software* de gestão de processos para vários tipos de metodologias, como por exemplo o **RUP** (Rational Unified Process) – Processo Racional Unificado.

05

2 - CONCEITUANDO PROCESSO

Outro conceito muito utilizado atualmente é **processo**. Hoje se ouve falar em processo em todos os lugares e momentos: “O processo está errado.” “As pessoas não seguem o processo.” Afinal, o que é processo?

Para definição de processo, iremos ao BPMCBOK (Business Process Management Commom Body of Knowledge), que é traduzido para o português como Corpo Comum de Conhecimentos em Gerenciamento de Processos de Negócio. BPMCBOK é exatamente um conjunto de conhecimentos comuns sobre BPM mantido por uma associação internacional Para saber mais sobre o CBOK, acesse o site <http://www.abpmp-br.org/>

Processo é um conjunto de atividades interdependentes, ordenadas no tempo e espaço de forma encadeada, que ocorrem como resposta a eventos e que possui um objetivo, início, fim, entradas e saídas bem definidos.

Essas atividades são geralmente interfuncionais ou interorganizacionais que trabalham juntas para criar um produto ou serviço final. Atividades são apresentadas no contexto da sua relação entre si para proporcionar uma visão da sequência e do fluxo. Isso inclui um conjunto definido de atividades ou comportamentos realizados por humanos, sistemas ou uma combinação dos dois e têm um ou mais resultados que podem levar ao fim do processo ou uma entrega a outro processo.

Pressman descreve o processo de *software* como uma série de passos importantes previsíveis, ou seja, um roteiro que ajude a criar um resultado de alta qualidade e dentro do prazo estabelecido. O roteiro então é chamado de **Processo de *software***.

Caro aluno, você em um determinado momento verá que as pessoas falam de metodologia, método e processo como se fossem as mesmas coisas. Pelas definições acima essas definições andam juntas e entrelaçadas. A diferença que o processo pode ser independente, mas alguns somente começam com a finalização de outros processos. Por exemplo, o processo de instalação dos bancos de um carro na montadora, depende do processo de instalação do carpete ter finalizado.

Segundo Steven Gordon e Judith Gordon as organizações mais sofisticadas tratam o desenvolvimento de sistemas como um processo. Apesar dos sistemas serem diferentes uns dos outros, eles seguem um roteiro previsível, bem definido e manejável.

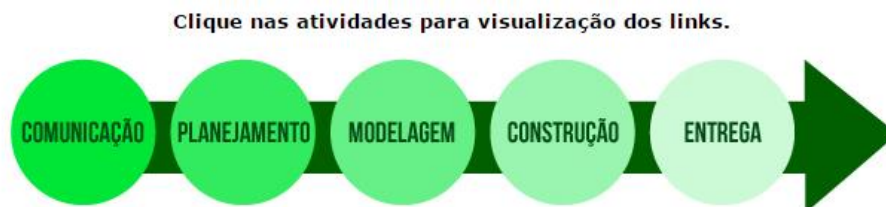
Pressman

Roger S. Pressman é um engenheiro de *software* escritor e consultor, norte-americano, presidente da R.S. Pressman Associates. Publicou diversos artigos e livros sobre tecnologia e gerenciamento.

06

3 - ATIVIDADES DE UMA METODOLOGIA

Segundo Pressman, a metodologia utilizando processo genérico para construção de um *software* compreende em cinco atividades:



Essas cinco atividades metodológicas definidas por Pressman podem ser utilizadas para desenvolvimento de sistemas de qualquer tamanho, complexos ou não.

Comunicação

Antes de iniciar qualquer trabalho, é vital e importante a comunicação com o cliente. É uma obrigação entender as necessidades do cliente e definir as funções e características do *software*.

Planejamento

É comum a utilização de um documento chamado Plano de Projeto, esse documento contém o trabalho que deve ser realizado e o que não deve ser realizado, as etapas e fases do trabalho, os riscos mais prováveis, os recursos que serão necessários, ou seja, é um documento que servirá como uma ferramenta de apoio na continuidade do projeto e o cliente deverão concordar com ele quando estiver

pronto.

Modelagem

É necessário criar um esboço do que será realizado, esse trabalho é para não termos surpresas no decorrer do desenvolvimento do *software*. Modelos, protótipos, o Analista de sistemas deverá usar tudo o que for possível para compreender as necessidades dos clientes.

Construção

Essas atividades são a construção de códigos e os testes necessários para deixar a aplicação confiável e sem erros.

Entrega

O *software* pronto é entregue para o cliente, que avalia já pensando em uma futura possível melhoria.

07

Atividades complementares

Outras atividades da metodologia são importantes também e servem de apoio; ao longo do desenvolvimento do *software* são aplicadas para gerenciar, controlar o progresso, a qualidade, as mudanças e gerenciar os riscos. As atividades de apoio típicas segundo Pressman são:

- Controle e acompanhamento de projeto;
- Administração de riscos;
- Garantia da qualidade de *software*;
- Revisões técnicas;
- Medições;
- Gerenciamento da configuração de *software*;
- Gerenciamento da reusabilidade;
- Preparo e produção de artefatos de *software*;

Essas atividades podem ser utilizadas ou não. Cada organização, de acordo com o seu tamanho, define sua maneira de trabalhar, sua metodologia, seus métodos e processos.

Controle e acompanhamento de projeto

A equipe avaliará o progresso do projeto de acordo com o Plano de Projeto e tomar as medidas necessárias para cumprir o cronograma. Em alguns casos é necessário realizar ajustes nos cronogramas,

tudo deve ser negociado com o cliente.

Administração de riscos

A equipe avaliará se os riscos levantados estão afetando o desenvolvimento do projeto. Os riscos devem ser gerenciados de perto.

Garantia da qualidade de *software*

A equipe avaliará o desenvolvimento do projeto e atuará em tempo de execução para que o *software* tenha a qualidade desejada.

Revisões técnicas

As revisões são importantes para evitar a propagação do erro. Equipes fazem revisões antes de prosseguir com as atividades seguintes.

Medições

Coleta informações do processo de desenvolvimento. Podem ser usadas para avaliar os requisitos solicitados e entregues. Outra opção é avaliar os custos do desenvolvimento, se estão aquém do combinado inicialmente.

Gerenciamento da configuração de *software*

Gerencia as mudanças e suas consequências. Mantém a base de documentos e códigos preservados organizando por versões de *software*, versões essas que possuem códigos, telas, documentos etc.

Gerenciamento da reusabilidade

Define como será realizado o reuso dos códigos no desenvolvimento do *software*. Essa atividade reduz o tempo de programação, mas é necessário ter um padrão de codificação na organização.

Preparo e produção de artefatos de *software*

É a parte da documentação do *software*, importantíssimo hoje, sistemas antigos documentados podem ser substituídos com maior facilidade.

08

4 - MELHORIA NOS PROCESSOS

As organizações aprendem com os erros e sucessos. As organizações possuem um conjunto de ferramentas que auxiliam na institucionalização, automatização e ainda apoiam os trabalhos da auditoria. Essas auditorias são benéficas, elas verificam se os caminhos (processos) estão sendo seguidos como definidos originalmente. A automatização dos processos ajuda na medição do trabalho

que está sendo realizado, possibilitando os ajustes (melhorias) nos processos. Sempre que há um ajuste, há uma medição. Medições do processo são importantíssimas para eventuais correções. Pode-se chegar à conclusão de que o processo não precisa ser alterado.

Grandes empresas executam ferramentas de verificação da **qualidade** do processo. Essas ferramentas medem os processos executados em cada projeto que está sendo construído ou cada sistema que está em manutenção.

Várias empresas possuem equipes voltadas para garantir a qualidade dos projetos, essas equipes atuam como guardiões dos processos e metodologias de desenvolvimento. Esses guardiões fazem verificações nos projetos para verificar se estão seguindo os padrões definidos pela organização. Essas verificações, além de corrigirem o rumo dos projetos, são insumos para a melhoria dos processos e das metodologias. Essas correções são efetuadas nos processos e a cada projeto é feita uma nova análise para verificar se as correções fizeram os efeitos desejados.

Metodologias e processos de desenvolvimentos servem para garantir a entrega do projeto para o cliente sem traumas, pois é um caminho conhecido, estudado, treinado e com um aporte de ferramentas que acompanham todo o desenvolvimento.

09

Equipes de **melhoria de processos** são importantíssimas, pois elas são responsáveis por implantar as melhorias nos processos e metodologias de desenvolvimento das empresas. Somente grandes empresas investem em equipes de melhoria de processos, outras trabalham com comitês de melhorias.

Comitês de melhorias são grupos de pessoas com conhecimento em um determinado assunto, que são responsáveis em discutir o assunto em pauta, ou seja, uma estrutura temporária para realizar uma melhoria nos processos que apoiam o desenvolvimento dos projetos.

Equipes fixas ou comitês responsáveis pela melhoria são importantes e não há uma sugestão para o que é melhor para a empresa. O importante é ter uma estrutura que veja sempre como possa melhorar continuamente o desenvolvimento dos projetos.

Qualquer metodologia ou processo de desenvolvimento deve ser ajustado de acordo com o andamento dos projetos. Isso faz parte da melhoria da qualidade. A melhoria das metodologias e dos processos depende da medição e verificação dos projetos, que inclui ações para aumentar a efetividade e a eficiência do projeto, fornecendo benefícios adicionais para as partes envolvidas.

Na maioria dos casos, a implementação de melhorias na qualidade exigirá preparação de requisitos de mudanças ou tomada de ações corretivas e serão gerenciadas de acordo como os procedimentos do controle integrado das mudanças.

10

Importância do uso de uma Metodologia ou um Processo de desenvolvimento

A utilização de uma metodologia ou um processo de desenvolvimento é importantíssima, porque propicia estabilidade, controle e organização para as atividades do desenvolvimento do projeto. A não utilização de metodologia ou um processo específico pode gerar uma desordem no desenvolvimento do *software* prejudicando a entrega e tornando o produto mais caro.

Mesmo utilizando uma metodologia ou um processo, o índice de fracasso é grande, segundo o Chaos Report (Standish Group), relatório frequentemente apresentado em palestra de gerenciamento de projetos de *software*, o qual informa dados do cenário de projetos americanos de 2009.

A figura abaixo mostra que apenas 32% dos projetos são entregues com sucesso. O gráfico mostra que a utilização de uma metodologia ou um processo de desenvolvimento também não garante o sucesso do projeto. Porque não garante o sucesso?

Existe um conjunto de fatores, que depende de cada projeto. Podemos elencar alguns como exemplos: falta de patrocínio, excesso de exigência dos processos de desenvolvimento, falta de equipes, prazo incompatível, requisitos incompletos etc. Continuando a examinar o gráfico temos 24% de projetos que falharam ou simplesmente foram cancelados, 44 % de projetos que atrasaram o que significa um prejuízo enorme para a empresa. Projetos que atrasam podem elevar os custos do projeto, podem prejudicar outros projetos da empresa e pior, prejudicar a imagem da empresa para os acionistas e para o mercado.



Imagine um projeto que não utiliza nenhum padrão de desenvolvimento, como seria? O caos para o projeto e para a organização, essa seria a resposta mais adequada.

11

Alguns projetos são iniciados sem **nenhum processo ou metodologia para apoiar sua construção**. Não é feita qualquer documentação e só se pensa na programação direta do produto e na entrega ao cliente. Caso o cliente resolva alterar alguma coisa no sistema será um problema enorme para a equipe de projetos. O tempo é a pior coisa para projetos que não seguem nenhum processo ou metodologia de desenvolvimento. Quanto maior o tempo, mais se esquece dos detalhes do sistema, pior ainda quando saem pessoas da equipe. Pessoas são promovidas, viajam, ficam doentes ou arrumam outros empregos. Processos de desenvolvimento proporcionam um conforto para o projeto e para os envolvidos no projeto ou especificamente os *stakeholders*. *Stakeholders* são os clientes, acionistas, pessoas do projeto, gerentes, etc. Pessoas que de alguma forma podem afetar o resultado do projeto. Na TI chamamos de PAPEL cada uma das funções que os *stakeholders* executam durante o desenvolvimento do sistema. Por exemplo: a equipe A possui um analista, que também executa as atividades de programador. Então, em um determinado momento ele está no papel do analista e em outro momento ele está no papel de programador.

Outros problemas que afetam o desenvolvimento do *software* é o **excesso de exigências** contidas em uma determinada metodologia ou processo. O projeto é o foco principal da empresa de TI, mas nunca se pode deixar que o purismo na utilização de um método afete o sucesso do projeto, do mesmo modo que não podemos deixar de lado a utilização e exigências metodológicas de documentação. Há de se ter um meio termo; a equipe de melhoria da empresa deve ser formada por pessoas que passaram por equipes de desenvolvimento, de preferência, analistas seniores.

Outro ponto importante: **não existe a melhor metodologia ou a pior**. Cada uma serve para um determinado tipo de projeto, depende do conhecimento da empresa e das práticas que são adotadas e estipuladas.

Geralmente empresas grandes deixam a cargo dos escritórios de projetos escolherem a melhor metodologia para desenvolver os seus projetos. Isso depende da *expertise* dos técnicos, gerentes, gestores e da empresa. O escritório de projetos orienta e conduz o gerente de projeto no planejamento do projeto e na realização de todas as etapas da metodologia, conforme estipulado pela organização.

12

RESUMO

As metodologias e processos são utilizados em diversas áreas no mundo todo e por grandes empresas, empresas estas que pensam nos acionistas, nos lucros e na imagem no mercado. A TIC evolui e os processos de desenvolvimento de projetos de *software* também. Pressman descreve o processo de *software* como uma série de passos importantes previsíveis, ou seja, um roteiro que ajude a criar um resultado de alta qualidade e dentro do prazo estabelecido. O investimento com qualidade do *software* é muito grande. Equipes de melhoria de processos são importantíssimas, pois elas são responsáveis por implantar as melhorias nos processos e metodologias de desenvolvimento das empresas. Para se tornar um profissional da área de melhoria de processo, o técnico deve ter uma bagagem de conhecimento sobre desenvolvimento de sistemas, sobre a empresa, deve ser dedicado ao estudo e também proativo. Não é uma área para se ficar esperando as coisas acontecerem, é preciso buscar as informações e problemas antes de estes acontecerem. O uso de metodologias e processos de desenvolvimento não garante o sucesso do projeto, mas a não utilização é o caos para o projeto e principalmente para a empresa. Por esse motivo as metodologias são tão importantes e sua evolução é fundamental, pois o mercado sempre está em evolução exigindo prazos mais curtos e clientes mais exigentes.

UNIDADE 1 – METODOLOGIAS DE DESENVOLVIMENTO DE SISTEMAS

MÓDULO 2 – PRINCIPAIS METODOLOGIAS DE DESENVOLVIMENTO DE *SOFTWARE*

01

1 – REQUISITOS

Percebemos no módulo anterior a importância do uso de um processo de desenvolvimento e também os problemas da não utilização. Atualmente se fizermos um levantamento nas empresas que desenvolvem sistemas verificaremos que todas utilizam alguma metodologia de desenvolvimento.

As metodologias de desenvolvimento de software surgiram com a evolução da TIC (Tecnologia da Informação e Comunicação) e atualmente há três mais utilizadas:

- metodologia estruturada,
- metodologia orientada a objetos,
- metodologia ágil.

O modelo estruturado ou simplesmente cascata, que surgiu por volta dos anos 1960 de acordo com as linguagens de desenvolvimento da época, por exemplo, o COBOL, ainda é muito utilizado por grandes bancos do governo. O modelo cascata é referência para quase todas as metodologias e modelos.

Conhecendo um pouco mais da história e da atualidade, a linguagem de programação COBOL foi criada por volta dos anos 60 para fins comerciais e atualmente mais de 50% dos códigos dos programas no mundo são nessa linguagem.

Entrando na década de 80, com a evolução das linguagens de programação, surgiu a metodologia orientada ao objeto (OO), voltada também para a linguagem de programação, por exemplo, Java.

Outra metodologia que iremos abordar nesse módulo e da qual atualmente muito se ouve falar é o método ágil ou metodologias ágeis, surgido nos anos 2000, após uma reunião com cabeças importantes da comunidade mundial de desenvolvimento de *software*.

02

Antes, porém, de estudarmos com detalhes cada uma das metodologias de desenvolvimento, convém abordarmos um conceito que permeará toda nossa matéria: requisito. É fundamental, portanto, ressaltar o conceito de **requisito**, pois estes nos acompanharão em todo o curso e também no dia a dia de trabalho.

Vejamos o conceito de requisito segundo dicionários utilizados atualmente.

| | | |
|------------------|---|--|
| Aurélio | → | s.m. Condição que se deve satisfazer para alcançar certo fim. / Exigência de ordem legal para que determinado processo possa ter andamento. |
| Michaelis | → | Adj. (lat. requisitu) desus. Requisitado, requerido. Sm. 1 Condição a que se deve satisfazer para que uma coisa fique legal e regular. 2 Exigência imprescindível para a consecução de certo fim. 3 Qualidades, dotes, predicados exigidos para certa profissão. |
| Priberam | → | 1. Coisa necessária e indispensável. 2. Condição indispensável; exigência. |
| Wikipedia | → | No âmbito da engenharia, um Requisito consiste da definição documentada de uma propriedade ou comportamento que um produto ou serviço particular deve atender. Na abordagem clássica de engenharia, conjuntos de requisitos são tipicamente utilizados como informações fundamentais para a fase de projeto de um produto ou serviço, especificando as propriedades e funções necessárias (ou desejáveis) a serem consideradas no desenvolvimento do projeto em questão. O conceito de requisito é também utilizado formalmente na ciência de computação, engenharia de software e engenharia de sistemas, referindo-se à definição de uma característica, atributo, habilidade ou qualidade que um sistema (ou qualquer um de seus módulos ou sub-rotinas) deve necessariamente prover para ser útil a seus usuários. |

De forma abreviada, podemos conceituar requisito para a TIC como sendo a necessidade do cliente, englobando o comportamento ou a condição de um produto, ou seja, as propriedades que o sistema deverá possuir para atender às necessidades do cliente em seu negócio.

No decorrer do nosso estudo, veremos que o requisito é tão importante que constitui uma área na TIC conhecida como **engenharia de requisitos**, a qual contempla todas as atividades que contribuem para a identificação, análise, negociação, especificação, documentação e validação dos requisitos, bem como da sua manutenção ao longo do tempo.

03

A metodologia estruturada é composta por uma sequência de fases e atividades, com entradas e saídas, regras, notação, programação, documentação, técnicas de levantamento, de gestão com foco no processo global.

Com os métodos de Yourdon (78) e DeMarco (79) popularizou a análise estruturada com base na decomposição funcional estruturada de sistemas através de Diagramas de Fluxo de Dados (DFDs), que descrevem o sistema como um conjunto de processos funcionais que realizam transformações de informação e comunicam-se através de fluxos de informação. Essa metodologia fornece um conjunto coerente e integrado de métodos e regras, que no seu conjunto formam uma técnica estruturada de análise e projeto de sistemas baseada nos seguintes conceitos: abordagem top-down, modelagem

funcional, representação gráfica do modelo, dicionário de dados e a descrição de processos e procedimentos, além de trabalho em equipe e documentação.

Atualmente é comum as pessoas chamarem a metodologia Estruturada de Cascata. Mas na época a Estruturada era referência, na verdade era um padrão. Ocorreram vários ensaios de métodos baseados no conceito estruturado, ensaios que na verdade eram derivações de uma abordagem de cima para baixo (*top-down*), da modelagem por funcionalidades, dentre outros elencados acima. Vale ressaltar uma prática muito eficiente, a qual algumas empresas ainda utilizam nos dias atuais: a construção de dicionários de dados, onde contempla o que significa cada atributo do sistema, o nome já diz, DICIONÁRIO.

As metodologias estruturadas tinham como objetivo formalizar o processo de identificação de requisitos, para reduzir as interpretações erradas e introduzir algumas técnicas baseadas em melhores práticas no processo de levantamento de requisitos e de análise e desenho das necessidades dos clientes.

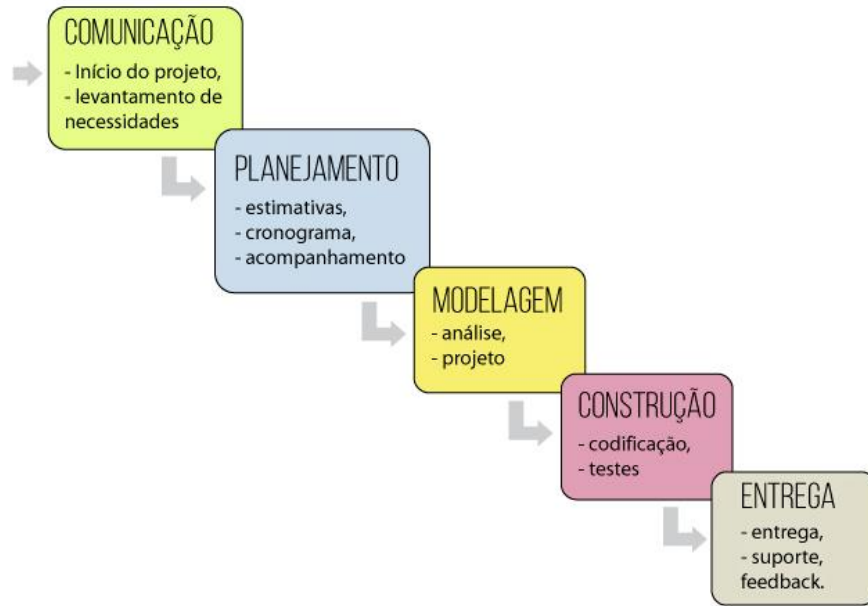
O princípio das linguagens da época é semelhante à metodologia estruturada, ou seja, blocos de códigos para cada funcionalidade. Focaremos esse estudo no **modelo cascata**, que era a base para quase todas as metodologias estruturadas de desenvolvimento dos sistemas.

Diagramas de Fluxo de Dados

O diagrama de fluxos de dados (DFD) é uma representação gráfica do "fluxo" de dados através de um sistema de informação, modelando seus aspectos de processo. O diagrama fornece apenas uma visão do sistema, a visão estruturada das funções, ou seja, o fluxo dos dados. Em geral, o DFD é uma etapa preliminar usada para criar uma visão geral do sistema que será elaborado.

04

Segundo Pressman, o modelo **cascata** algumas vezes foi chamado de modelo clássico. Sugere uma abordagem sequencial e sistemática para o desenvolvimento de *software*, começando com o levantamento de necessidades por parte do cliente, avançando pelas fases do planejamento, modelagem, construção e entrega, todo o projeto acaba culminando no suporte contínuo do *software* concluído. Grande parte dos bancos possui inúmeros sistemas que foram construídos utilizando metodologia estruturada ou simplesmente cascata, bancos esses que possuem sistemas com mais de 30 anos e que continuam em pleno funcionamento.



Modelo cascata - Pressman

05

Vantagens e desvantagens do modelo cascata

O modelo cascata é voltado à documentação do sistema e uma das principais vantagens é que só avança para a tarefa seguinte quando o cliente valida e aceita os produtos da tarefa atual. O modelo pressupõe que o cliente participa ativamente no projeto e que sabe muito bem o que quer, pois todos os requisitos são levantados no início. Outro ponto positivo é que o cronograma atende todo o projeto, ou seja, você terá uma data início e fim bem definidos.

Alguns problemas eram notados da utilização da metodologia estruturada servindo de **desvantagem**:

- alguns sistemas cresciam durante o projeto, isso gerava um retrabalho, pois era necessário revisitar as fases já cumpridas;
- a grande necessidade de manutenção dos softwares gerava um problema, pois se executava os grandes processos sequenciais novamente, o que gerava prazos longos de entrega;
- alterações dos requisitos ao longo do projeto causavam transtornos e atraso no cronograma gerando retrabalho;
- difícil reutilização de módulos, os erros de levantamento de requisitos eram vistos tardiamente e o custo de reconstrução era elevado.

06

3 - METODOLOGIA ORIENTADA A OBJETO

A orientação por objetos se baseia em uma nova forma de ver e analisar o mundo, pois visualiza a forma com que as pessoas percebem e expressam a realidade que está a sua volta. As pessoas classificam e dividem os diferentes objetos existentes com base nas diferenças e semelhanças de acordo com o comportamento e características dos objetos. Pressman cita o exemplo da “cadeira” para definir o que é uma classe.

A cadeira é uma subclasse de uma classe maior, a mobília, e essa tem um conjunto de atributos que podem compor essa classe, como as dimensões, custo, peso, localização, cor. Independente de ser cadeira, mesa, todos fazem parte da mesma classe, a **mobília**, ou seja, a cadeira herda todos os atributos definidos para a classe, daí vem o conceito de herança. Sendo assim a orientação por objetos identifica e define cada objeto para que seja reutilizado posteriormente.

Alguns novos conceitos foram introduzidos com a OO, que teve também a sua criação pautada em linguagem de programação, como exemplo citamos a linguagem JAVA. Conceitos como o de objeto, classe (citado acima), encapsulamento, herança (citado acima), abstração, etc. No caso de objeto, pode ser real ou abstrato que contenha uma informação, para o conceito de encapsulamento vem de encapsular, que em programação orientada a objetos significa separar o programa em partes, o mais isolado possível. A ideia é tornar o software mais flexível, fácil de modificar e de criar novas implementações. A abstração representa uma entidade do mundo real e sua utilização diminui o impacto se houver alguma mudança no código, pois tem um baixo acoplamento. Esses conceitos serão tratados rigorosamente na disciplina de Programação Orientada a Objetos.

Atualmente o processo unificado é utilizado para apoiar o desenvolvimento orientado a objetos. O processo unificado intitulado hoje como RUP, foi criado pela *Rational Software Corporation* e foi adquirida em fevereiro pela IBM. Focaremos nossos estudos no RUP, que significa **Processo Unificado Rational**.

07

Segundo a IBM, o RUP é um *framework* de processo abrangente que fornece práticas testadas pela indústria de *software* e sistemas de entrega e de execução, para uma gestão eficaz do projeto. É um dos muitos processos contidos na biblioteca de processos da Rational, que oferece as melhores práticas de orientação adequada para o seu desenvolvimento em particular ou necessidade do projeto.

RUP promove o desenvolvimento iterativo e organiza o desenvolvimento de *software* e sistemas em fases, cada uma composta por uma ou mais iterações executáveis do *software* e em estágio de desenvolvimento.

O principal objetivo do RUP é atender as necessidades dos usuários garantindo uma produção de *software* de alta qualidade que cumpra um cronograma e um orçamento previsíveis.

O RUP define perfeitamente os papéis (responsáveis), como as coisas deverão ser feitas e quando

devem ser realizadas, descrevendo todas as metas de desenvolvimento especificamente para que sejam alcançadas. No RUP temos vários papéis, por exemplo: gerente de projetos, analista de requisitos, analista de testes, arquiteto etc.

08

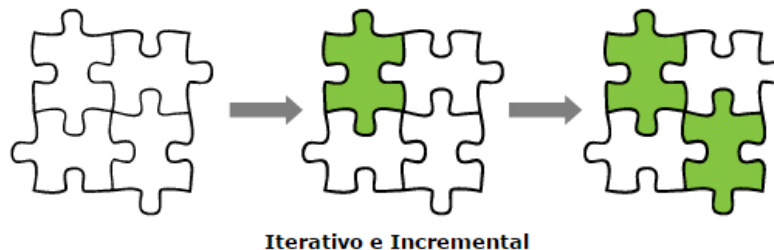
O RUP organiza o desenvolvimento de *software* em quatro fases:

- iniciação,
- elaboração,
- construção e
- transição.

Cada fase tem um papel fundamental para que o objetivo seja cumprido. Além das fases, o RUP possui nove disciplinas.

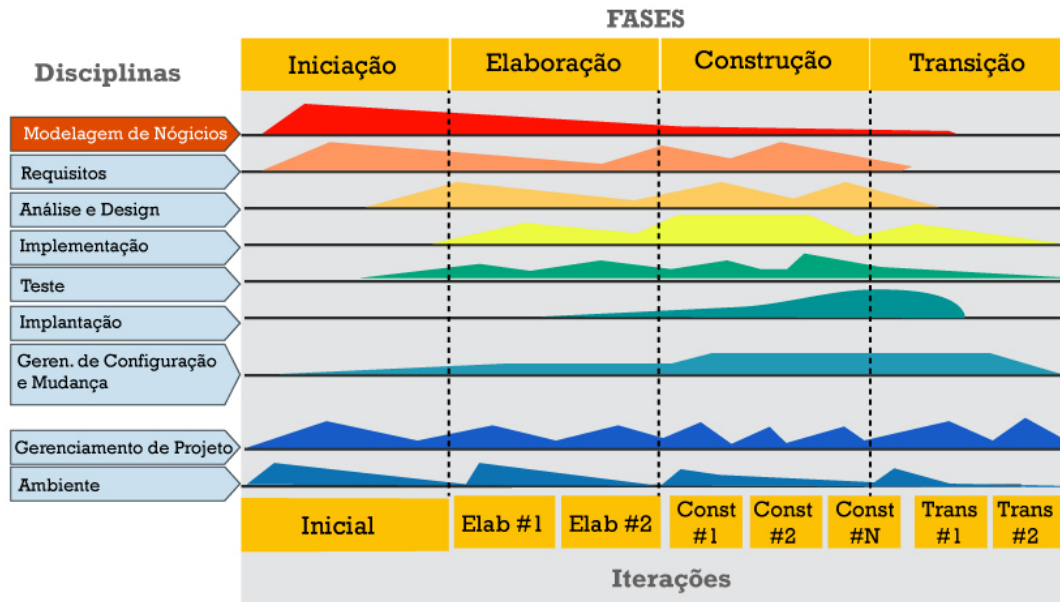
Podemos conceituar **disciplina** para a TIC como sendo uma área de conhecimento da engenharia de *software* que tem como objetivo apoiar a construção do sistema durante o seu ciclo de vida.

O processo unificado é centrado na arquitetura e seu desenvolvimento é iterativo e incremental, o que significa que cada fase possui “n” iterações conforme as disciplinas. Essas iterações são partes do projeto que vão se juntando ao *software* ou incrementando.



09

O **gráfico das baleias** apresenta a intensidade de utilização de cada uma das nove disciplinas, quanto maior a barriga da baleia maior a intensidade de atuação na fase. Essa intensidade significa execução de tarefas de cada disciplina nas iterações planejadas. O principal desafio no início do projeto é concluir a arquitetura do *software* ou aplicação.



Arquitetura é o que faz o sistema funcionar de acordo com as necessidades do cliente, como por exemplo, a linguagem com a qual se vai construir o *software*, os servidores, quantidade de usuários, necessidade de integração com outros sistemas etc.

10

Disciplina é uma palavra muito utilizada na TIC, vale a pena entender melhor o seu significado. Conceituando disciplina segundo dicionários utilizados atualmente, temos:

| | | |
|------------------|---|--|
| Michaelis | → | sf (lat. disciplina) 1 obsol. Ensino, instrução e educação. 2 Relação de submissão de quem é ensinado, para com aquele que ensina; observância de preceitos ou ordens escolares: Disciplina escolar. 3 Sujeição das atividades instintivas às refletidas. 4 Observância estrita das regras e regulamentos de uma organização civil ou estatal: Disciplina militar. 5 Conjunto de conhecimentos científicos, artísticos, linguísticos etc., que se professam em cada cadeira de um instituto escolar. 6 Obediência à autoridade. 7 Procedimento correto. 8 Castigo, mortificação. |
| Priberam | → | 1. Conjunto de leis ou ordens que regem certas coletividades. 2. Boa ordem e respeito. 3. Submissão, obediência. 4. Instrução e educação. 5. Ensino. 6. Ação dirigente de um mestre. 7. Estudo de um ramo do saber humano. 8. Autoridade. 9. Obediência à autoridade. |
| Wikipedia | → | Disciplina é uma palavra que tem a mesma etimologia da palavra "discípulo", que significa "aquele que segue". Também é um dos nomes que se pode dar a qualquer área de conhecimento estudada e ministrada em um ambiente escolar ou acadêmico. Geralmente diz respeito a uma Ciência ou Técnica, ou subderivados destas. Aqueles que seguem uma disciplina podem assim ser chamados de discípulos. |

Na TIC, as disciplinas constituem um conjunto de instruções de um determinado assunto, que se encaixam umas com as outras.

Não é obrigatório o uso de todas as disciplinas em um projeto e nem de todas as tarefas descritas em cada disciplina, mas para isso é necessário que o gerente de projetos entenda as necessidades do projeto e que faça o seu plano de projeto somente com as atividades que serão necessárias.

11

Vantagens e desvantagens do RUP

Uma grande vantagem do RUP é que este funciona como uma biblioteca de elementos (tarefas, artefatos e funções) e pode ser ajustado conforme as necessidades da empresa de acordo com os padrões do Processo Unificado. Esses ajustes podem ser realizados por uma ferramenta (RMC – Rational Method Composer) que consegue abrir o RUP e realizar as mudanças. O RUP possibilita definir vários papéis durante o desenvolvimento do aplicativo. Há uma descrição de cada papel, artefato e função a ser executada e como tudo funciona durante o desenvolvimento do projeto, assim os profissionais sabem exatamente quais tarefas deverão executar. Um profissional pode executar vários papéis.

Em termos de desvantagens, do mesmo modo quando definimos inúmeros papéis para o projeto também devemos ter equipe para isso, e se definimos uma pessoa para vários papéis corremos o risco de atrasar o projeto, caso essa pessoa tenha algum imprevisto, pode atrasar suas tarefas ou até o planejamento inteiro.

Outra desvantagem é o excesso de atividades definidas; o preciosismo e o excesso de burocracia geram grande quantidade de documentos a serem construídos, que nem sempre são imprescindíveis para o projeto específico, devido ao tamanho do projeto e o tempo de entrega ao cliente.

12

4 - METODOLOGIA ÁGIL

A metodologia ágil surgiu a partir de um manifesto realizado em fevereiro de 2001 por um grupo de pensadores renomados e independentes sobre desenvolvimento de *software*, e às vezes concorrentes entre si, que concordaram com o Manifesto para Desenvolvimento Ágil de *Software*. O manifesto for *Agile Software Development*, ou simplesmente **Manifesto Ágil** que diz:

O movimento formalizou princípios básicos que dão suporte aos métodos ágeis de desenvolvimento de software. Foram criados então os 12 princípios da construção de um software rápido:

“Estamos descobrindo maneiras melhores de desenvolver software fazendo-o nós mesmos e ajudando outros a fazê-lo. Através deste trabalho, passamos a valorizar:

Indivíduos e interação entre eles mais que processos e ferramentas;

| | |
|--|-----------------------------------|
| Software em funcionamento | mais que documentação abrangente; |
| Colaboração com o cliente | mais que negociação de contratos; |
| Responder a mudanças | mais que seguir um plano. |
| Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.” | |

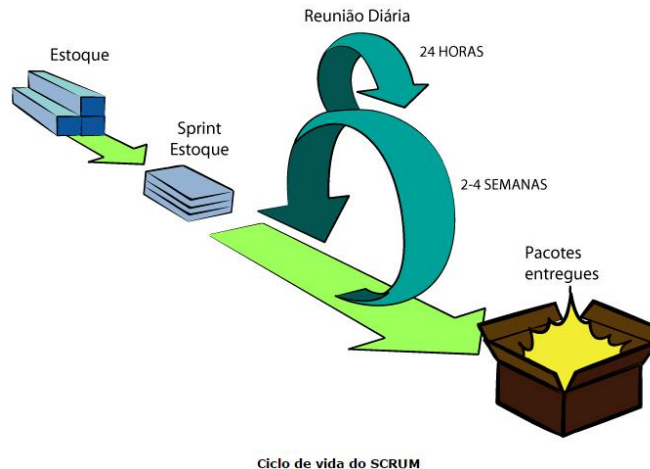
1. Nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de software de valor.
2. Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças, para que o cliente possa tirar vantagens competitivas.
3. Entregar software funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos.
4. Pessoas relacionadas a negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto.
5. Construir projetos ao redor de indivíduos motivados. Dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho.
6. O método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é através de uma conversa cara a cara.
7. Software funcional é a medida primária de progresso.
8. Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários, devem ser capazes de manter indefinidamente, passos constantes.
9. Contínua atenção à excelência técnica e bom design, aumenta a agilidade.
10. Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito.
11. As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis.
12. Em intervalos regulares, o time reflete como ficar mais efetivo, então, se ajustam e otimizam seu comportamento de acordo.

13

A metodologia ágil não está ligada ou vinculada necessariamente a uma linguagem de programação, como ocorreu com a estruturada e a orientada a objetos. Hoje é possível, na maioria das vezes, abstrair a linguagem de programação da metodologia, ou seja, é possível desenvolver qualquer linguagem com qualquer metodologia, a diferença estará nos riscos e adaptações que deverá acontecer.

A metodologia ágil agrega filosofia e visa consolidar as práticas de desenvolvimento de *software* ditas "ágeis", ou **métodos ágeis**, tais como o Scrum, XP, Lean entre outras.

Essa metodologia é também baseada no processo unificado, pelo fato de compartilharem alguns princípios, como desenvolvimento iterativo incremental, orientação a objeto, foco em qualidade e entregas, dentre outros.



O método ágil tem como principal objetivo otimizar o processo de desenvolvimento, visando melhorar o cumprimento dos prazos e agilizar as entregas, possibilitando também a melhoria contínua, por meio de uma maior interação da equipe e da resolução eficaz de problemas.

Vale destacar que o objetivo da construção do cenário ágil foi prover um ambiente que apresente a filosofia de desenvolvimento ágil, em seus princípios de gerenciamento de projetos e de práticas para o desenvolvimento de *software* com foco nas entregas rápidas, tendo em vista a necessidade de oferecer processos mais simples e otimizados para a realidade da empresa.

SCRUM

Metodologia ágil para gestão e planejamento ágil de software de forma iterativa e incremental.

XP

Programação EXtrema – método ágil para desenvolvimento de *software*;

Lean

Método que aplica os princípios de eliminação de desperdício e valorização da qualidade e do conhecimento, dentre outros.

14

Vantagens e desvantagens do método ágil

Excelente para equipes com alto grau de maturidade, autossuficientes e autogerenciáveis e com iniciativas e conhecimento do processo que fazem o projeto ser entregue mais rápido, o gestor ou

cliente trabalha junto validando os requisitos e fazendo testes a cada implementação. A equipe adquire conhecimento no negócio a ser implementado, o que diminui o tempo de documentação dos requisitos. Número reduzido da equipe é outro fator vantajoso e o mais importante é a possibilidade de entregas semanais.

Algumas vantagens podem tornar-se desvantagens, por exemplo, se o cliente ou gestor estiver comprometido com outros projetos, o método ágil perde sua eficiência, pois se torna um projeto com os tempos de um projeto normal. Outra vantagem que pode virar desvantagem é quando a equipe sofre alguma baixa, pois o novo profissional alocado no projeto pode demorar a se colocar a par dos eventos, o que compromete os prazos. Projetos com mais de um gestor podem ocasionar problemas na conciliação da agenda. Outro fator importante é conseguir montar uma equipe autossuficiente, geralmente as equipes são montadas com pessoas que estão disponíveis e não com as pessoas necessárias para o projeto.

15

RESUMO

A melhor metodologia é aquela que melhor se adapta a empresa e aos técnicos que irão desenvolver. Se uma metodologia ou os processos foram desenvolvidos em um nível corporativo, ela somente será efetiva se atender as equipes de projetos que a utilizam. O cenário ideal é você poder adaptar à sua própria necessidade os processos de desenvolvimento. O processo puramente cascata ou unificado e até mesmo ágil são bons e ruins dependendo do caso de cada empresa e de cada projeto a ser desenvolvido.

Modelos de processos estruturados ou cascata são mais antigos, que tinham como objetivo inicial formalizar o processo de identificação de requisitos e eram realizados em blocos sequenciais, atualmente temos uma boa parte de projetos mantidos por esses métodos. Modelos de processo incremental ou simplesmente Processo Unificado são iterativos e incrementais, ou seja, o projeto vai sendo construído aos poucos percorrendo pelas fases várias vezes até ficar pronto, hoje são a maioria no desenvolvimento, modelos ágeis são bem visto pelo mercado, têm na sua essência o processo unificado e estão ganhando proporções e sendo adaptados para grandes empresas. Hoje o desafio é fazer certo com agilidade e sem burocracia, cumprir os prazos e atender os requisitos solicitados pelos clientes.

UNIDADE 1 – METODOLOGIAS DE DESENVOLVIMENTO DE SISTEMAS

MÓDULO 3 – APLICAÇÃO DAS METODOLOGIAS DE DESENVOLVIMENTO DE SOFTWARE

01

1 - FINALIDADE DO USO DE METODOLOGIA

Modelos de desenvolvimentos podem trazer inúmeros benefícios se aplicados de maneira correta de acordo com as características da empresa e do cliente. O que não podemos deixar de elencar é que todo projeto é iniciado pela comunicação e entendimento do que se vai construir. Esse entendimento é a compreensão e o aprendizado de tudo que for necessário para construção do sistema, que é o produto final do cliente.

O entendimento das necessidades que pode determinar o tipo de metodologia ou modelo que se vai utilizar. Digo “pode” porque **a melhor metodologia que a empresa utiliza é aquela que ela domina, não a que o mercado indica**. Vejam que aprendemos no módulo anterior que as grandes empresas possuem áreas que fazem as melhorias nos processos a partir de um levantamento da qualidade do *software* que está sendo desenvolvido.

Essas *Equipes de Qualidade* são as responsáveis por buscar as melhores práticas no mercado, as quais serão estudadas e possivelmente serão utilizadas na sua completude ou adaptadas às características da organização. As equipes da qualidade devem ser especialistas na Engenharia de *Software* para melhor apoiar o desenvolvimento das aplicações e realizar as melhorias necessárias.

Algumas empresas determinaram em seus processos de desenvolvimento a extinção do uso de algumas metodologias. No caso a metodologia estruturada é a que está em desuso no âmbito do desenvolvimento dos softwares. O mercado adaptou seus projetos e manutenções no uso do **processo unificado**, que é a abordagem iterativa e incremental.

É importante definirmos o que chamamos de **projeto** e **manutenção**:



Abordaremos mais detalhadamente sobre esse assunto no próximo módulo.

02

2 - CONCEITO DE ENGENHARIA DE *SOFTWARE*

Nesse módulo introduziremos o conceito de Engenharia de *Software*, que nos acompanhará para sempre na nossa vida em TIC.

O conceito de engenharia surgiu inicialmente em 1968, em uma conferência sobre a crise do *software*.

Para Sommerville, engenharia de *software* é o ramo da engenharia cujo foco é o desenvolvimento de

sistemas de *software* de alta qualidade dentro de custos adequados.

Software é abstrato e intangível. Não é limitado por materiais ou controlado por leis da física ou por processos manufaturados. Não existem limitações físicas para o *software*, que pode se tornar algo extremamente complexo e difícil de ser compreendido. A engenharia de *software* não está relacionada apenas aos processos de desenvolvimento de *software*, mas também às atividades de gerenciamento de projetos, desenvolvimento de ferramentas, métodos, teorias que apóiam a construção e produção de *software*.

Segundo Fritz Bauer, engenharia de *software* é o estabelecimento e o emprego de sólidos princípios de engenharia de modo a obter um *software* de maneira econômica, que seja confiável e funcione de forma eficiente em máquinas reais.

Para Pressman a engenharia de *software* deve estar fundamentada em um comprometimento organizacional com qualidade. A engenharia de *software* é a liga que mantém as camadas de tecnologias coesas e possibilita o desenvolvimento de *software* de forma racional e dentro do prazo.

Esses conceitos são importantes, por isso, estudaremos e aprofundaremos mais sobre engenharia de *software* no decorrer do curso.

03

3 - MODELOS DE MATURIDADE

Percebemos que as metodologias estão totalmente ligadas e dentro da engenharia de *software*, pois engloba processos, métodos, bem como as ferramentas que serão utilizadas.

Outro aspecto que vale estudar são os modelos externos que servem para a melhoria dos processos de *software* das empresas. Com o aumento da dependência em serviços da TIC, além da diversidade das tecnologias, fábricas de *softwares* lutam para manter em alto nível os serviços para os seus clientes.

Algumas empresas trabalham de forma reativa, ou seja, ficam aguardando os problemas acontecerem para atuar, o que traz prejuízos para a empresa e compromete o projeto e a manutenção dos sistemas. Devem-se adotar práticas proativas e estruturadas de trabalho. Desta forma, salientamos que a qualidade é fator crítico de sucesso para o desenvolvimento de sistemas. Para que a empresa se mantenha no setor competitivo, nacional e internacional, é essencial que as organizações coloquem a eficiência e a eficácia dos seus processos em foco nas empresas, visando à oferta de serviços conforme padrões internacionais de qualidade.

No mercado atual existem **modelos de maturidade**, que são padrões de processos explicitamente definidos, gerenciados, medidos, controlados e eficazes, ou seja, são diretrizes que podem auxiliar uma organização a melhorar sua forma de fazer negócios.

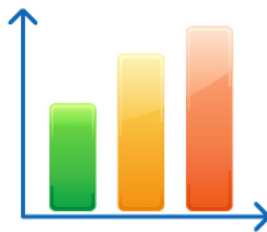
Entretanto, a maioria das abordagens disponíveis para melhoria tem seu foco em uma parte específica

do negócio e não utilizam uma abordagem sistêmica em relação aos problemas enfrentados por grande parte das organizações. Ao focar na melhoria de uma única área da empresa, esses modelos infelizmente não conseguem emplacar e perpetuam barreiras e compartimentalizações que existem nas organizações.

04

No mercado nacional e internacional, atualmente, há alguns modelos de referência, que possibilitam a avaliação das metodologias e processos de desenvolvimento das empresas. Esses modelos são reconhecidos pela comunidade de engenharia de *software*, sendo que algumas empresas públicas colocam em seus editais, como por exemplo, a exigência que os processos das empresas participantes tenham sido avaliados com sucesso por esses modelos.

Esses modelos são compostos por **níveis de maturidade**, os processos e metodologias são avaliados conforme esses níveis de maturidade e recebem uma avaliação, essa avaliação define em que nível de maturidade está o seu processo de desenvolvimento conforme o modelo que está avaliando.



Hoje o mercado tem empresas especializadas de implantação e avaliação dos modelos de maturidade tendo em vista a importância desses modelos para a melhoria da qualidade das aplicações. Os níveis de maturidade estabelecem patamares de evolução de processos, caracterizando estágios de melhoria dos processos na organização.

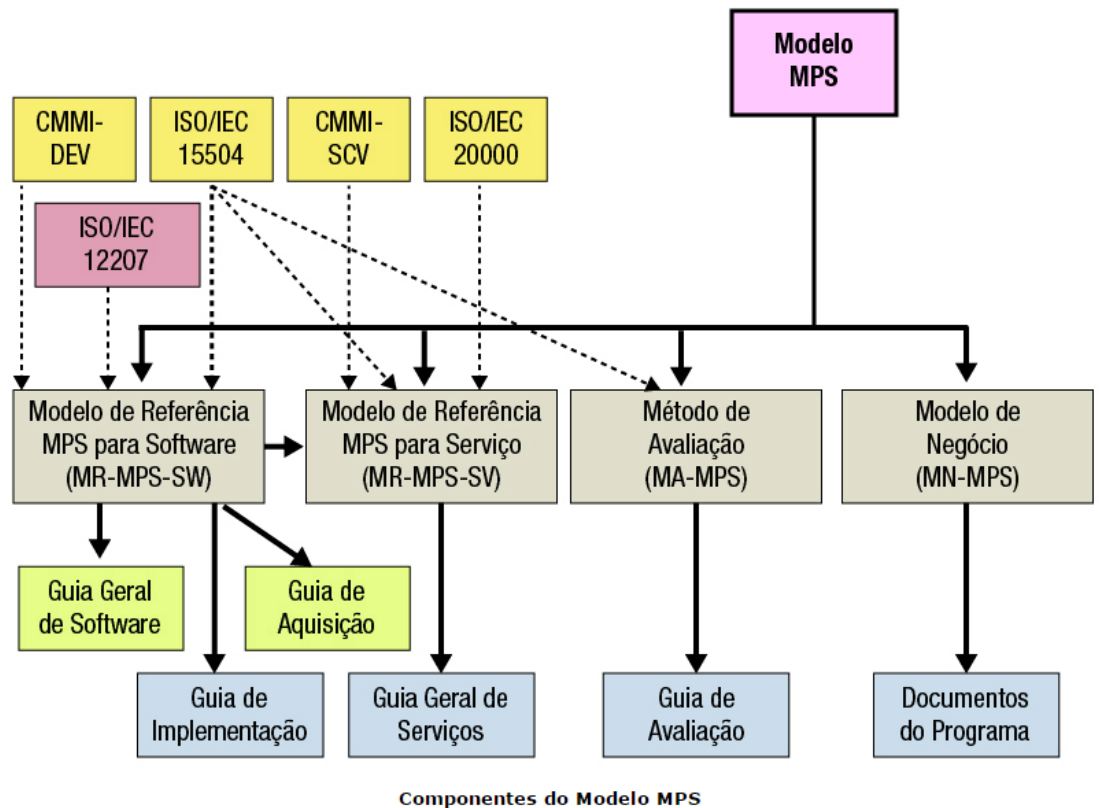
Apresentaremos a seguir dois modelos muito utilizados no mercado internacional e nacional: MPS-BR e o CMMI.

05

a) MPS-BR – Melhoria do Processo de *Software* e Serviços – Brasileiro

Uma das metas do **Programa MPS.BR** é definir e aprimorar um modelo de melhoria e avaliação de processo de *software* e serviços, visando preferencialmente às micro, pequenas e médias empresas, de forma a atender as suas necessidades de negócio e ser reconhecido nacional e internacionalmente como um modelo aplicável à indústria de *software* e serviços.

O modelo MPS estabelece dois modelos de referência de processos de *software* e serviços e um processo/método de avaliação de processos. Esta estrutura fornece sustentação e garante que o modelo seja empregado de forma coerente com as suas definições. O MPS estabelece também um modelo de negócio para apoiar a sua adoção pelas empresas desenvolvedoras de *software* e prestadores de serviços. A figura mostra a estrutura do MPS-BR e suas referências.

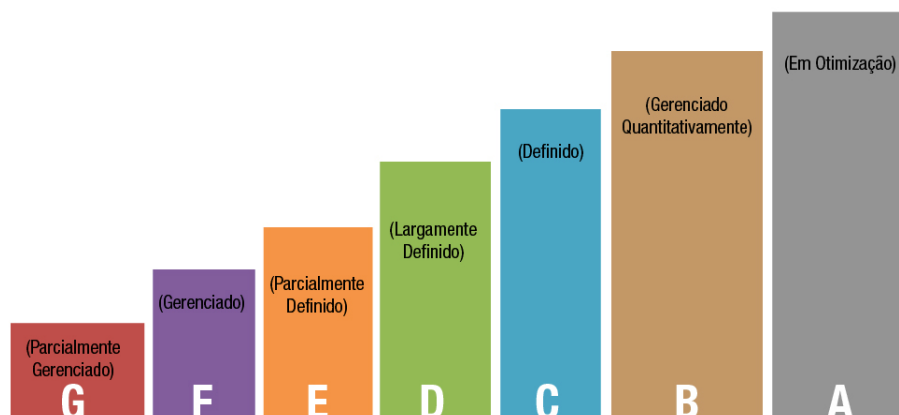


Após uma avaliação, a estrutura possibilita verificar o nível de maturidade em que se encontra uma organização, essa ação permite prever o seu desempenho futuro ao executar um ou mais processos.

Programa MPS.BR

Conforme descrição do guia geral MPS de serviços, o MPS BR é um programa mobilizador de longo prazo, criado em dezembro de 2003, coordenado pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), que conta com apoio do Ministério da Ciência, Tecnologia e Inovação (MCTI), Financiadora de Estudos e Projetos (FINEP), Serviço Brasileiro de Apoio às Micro e Pequenas Empresas (SEBRAE) e Banco Interamericano de Desenvolvimento (BID/FUMIN).

O MR-MPS-SV define sete **níveis de maturidade**:



Conforme o MPS-BR, a escala de maturidade se inicia no nível G e progride até o nível A.

Para cada um destes sete níveis de maturidade é atribuído um perfil de processos que indicam onde a organização deve colocar o esforço de melhoria. O progresso e o alcance de um determinado nível de maturidade do MR-MPS-SV se obtêm quando são atendidos os propósitos e todos os resultados esperados dos respectivos processos e os resultados esperados dos atributos de processo estabelecidos para aquele nível.

A divisão em sete estágios tem o objetivo de possibilitar uma implementação e avaliação adequada às micros, pequenas e médias empresas. A possibilidade de se realizar avaliações considerando mais níveis também permite uma visibilidade dos resultados de melhoria de processos em prazos mais curtos.

Os níveis são acumulativos, ou seja, na passagem para um nível de maturidade superior, os processos anteriormente implementados devem passar a ser executados no nível de capacidade exigido neste nível superior.

Acumulativos

Isso quer dizer que, se a organização está no nível F, esta possui o nível de capacidade do nível F que inclui os atributos de processo dos níveis G e F para todos os processos relacionados no nível de maturidade F (que também inclui os processos de nível G). Isto significa que, ao passar do nível G para o nível F, os processos do nível de maturidade G passam a ser executados no nível de capacidade correspondente ao nível F.

07

Os diferentes níveis de capacidade dos processos são descritos por nove **atributos de processo (AP)**. O alcance de cada atributo de processo é avaliado utilizando os respectivos resultados esperados de atributo de processo.

Alguns processos podem ser excluídos, total ou parcialmente, do escopo de uma avaliação MPS por não serem pertinentes ao negócio da unidade organizacional que está sendo avaliada. Cada exclusão deve ser justificada no Plano de Avaliação. A aceitação das exclusões e suas justificativas é responsabilidade do Avaliador Líder, conforme descrito no Guia de Avaliação [SOFTEX, 2012b].

Saiba+

Conforme site da Softex o desenvolvimento e a melhoria das práticas de serviços são chaves para um melhor desempenho, aumento da satisfação do cliente e a lucratividade do setor [SEI, 2010b]. O site também possui uma gama de informações sobre o modelo para quem está iniciando e para empresas já avançadas no processo de certificação.

Abaixo a tabela com os níveis de maturidade, processos e atributos de processos do MPS-BR.

| Nível | Processos | Atributos de Processo |
|----------|---|--|
| A | | AP 1.1, AP 2.1, AP 2.2, AP 3.1, AP 3.2, AP 4.1, AP 4.2, AP 5.1, e AP 5.2 |
| B | Gerência de Trabalhos - GTR (evolução) | AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2, AP 4.1 e AP 4.2 |
| C | Gerência de Trabalhos - GTR (evolução) | AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2 |
| | Gerência de Continuidade e Disponibilidade dos Serviços - GCD | |
| | Gerência de Decisões - GDE | |
| | Gerência de Liberação - GLI | |
| | Gerência de Riscos - GRI | |
| | Gerência da Segurança da Informação - GSI | |
| | Relato de Serviços - RLS | |
| D | Desenvolvimento do Sistema de Serviços - DSS | AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2 |
| | Orçamento e Contabilização de Serviços - OCS | |
| E | Avaliação e Melhoria do Processo Organizacional - AMP | AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2 |
| | Definição do Processo Organizacional - DFP | |
| | Gerência de Mudanças - GMU | |
| | Gerência de Recursos Humanos - GRH | |
| | Gerência de Trabalhos - GTR (evolução) | |
| F | Aquisição - AQU | AP 1.1, AP 2.1 e AP 2.2 |
| | Gerência de Configuração - GCO | |
| | Gerência da Qualidade - GQA | |
| | Gerência de Problemas - GPL | |
| | Gerência de Portfólio de Trabalhos - GPT | |
| | Medição - MED | |
| G | Entrega de Serviços - ETS | AP 1.1 e AP 2.1 |
| | Gerência de Incidentes - GIN | |
| | Gerência de Nível de Serviço - GNS | |
| | Gerência de Requisitos - GRE | |
| | Gerência de Trabalhos - GTR | |

Níveis de maturidade do MR-MPS-SV, baseado em [MACHADO, 2011]

b) CMMI – Capability Maturity Model Integration

Conforme descrito no livro CMMI para desenvolvedores, o Instituto de Engenharia de *Software* da Universidade de Carnegie Mell (SEI/CMU), em Pittsburg, Estados Unidos realizou vários estudos, que resultou na documentação de um conjunto de melhores práticas para os processos de desenvolvimento de *software* descritas em Modelos de Maturidades. Vários modelos foram criados e evoluídos e consequentemente substituídos por um único modelo, o CMMI, que traduzido significa **Integração de Modelos de Capacidade e Maturidade**.

O CMMI é um modelo de referência que contém boas práticas necessárias para melhoria e maturidade corporativa das disciplinas específicas. Esse modelo tem apoio do Departamento de Defesa dos estados Unidos.

Segundo o Instituto o objetivo do CMMI é melhorar os processos de desenvolvimento da organização, bem como melhorar o gerenciamento do desenvolvimento, os processos de aquisição, manutenção dos sistemas, o gerenciamento da qualidade, as melhores práticas aplicadas a melhoria e mudança de processos da organização.

O CMMI fornece práticas de avaliação de maturidade de processos de acordo com o modelo. É formado por um pacote de produtos que reúne vários modelos de processo, associados a manuais, material de treinamento e avaliação. Esses modelos são integrados e transcendem as disciplinas.

09

Segundo o SEI o CMMI para Desenvolvimento consiste das melhores práticas relativas às atividades de desenvolvimento e manutenção aplicadas a produtos e serviços. Ele abrange práticas que cobrem o ciclo de vida do produto desde a concepção até a entrega e manutenção, e se concentra no trabalho necessário para construção e manutenção do produto em sua totalidade.

Equipes da Qualidade das empresas podem utilizar o CMMI para abordar a melhoria e avaliar seus processos utilizando duas **representações** diferentes contidas no modelo:

- contínua e
- por estágios.

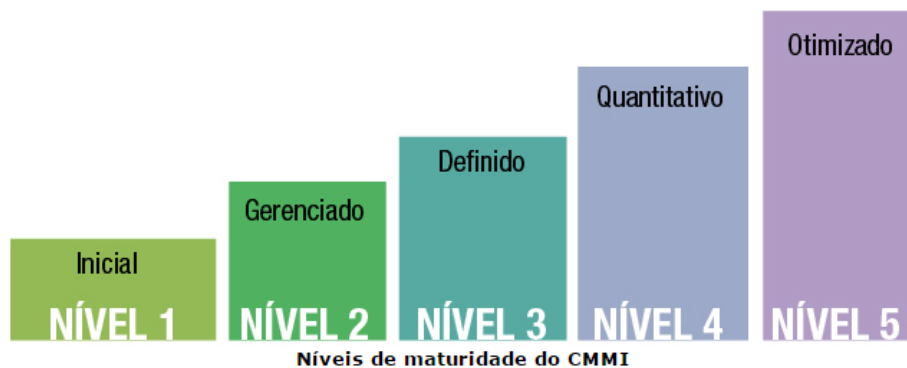
A representação mais conhecida e utilizada no mercado é a **por estágios**.

A representação por estágios, segundo o modelo, utiliza conjuntos predefinidos de áreas de processo para definir um caminho de melhoria para uma organização. Esse caminho de melhoria é caracterizado por níveis de maturidade. Cada nível de maturidade contém um conjunto de áreas de processos que caracterizam diferentes comportamentos organizacionais.

É baseado em um modelo, que enfoca um estágio por vez. A conquista de cada estágio assegura que foi estabelecida uma infraestrutura adequada de processos que servirá de base para o próximo estágio. Esses estágios equivalem aos níveis de maturidade. A conquista de cada nível assegura que foi estabelecida uma base de melhoria adequada para o próximo nível de maturidade, permitindo uma melhoria incremental e duradoura. Os níveis de maturidade são medidos pela satisfação das metas específicas e genéricas associadas a cada conjunto predefinido de áreas de processo.

10

Existem cinco níveis de maturidade, numerados de 1 a 5. Cada um é um estágio ou camada, que representa a base para as atividades de melhoria contínua de processo: 1. Inicial; 2. Gerenciado; 3. Definido; 4. Gerenciado Quantitativamente; 5. Em Otimização.



Um nível de maturidade é um marco evolutivo definido para melhoria de processo. Cada nível de maturidade representa o amadurecimento de um importante subconjunto dos processos, onde se prepara para alcançar o próximo nível de maturidade.

Se não se sabe por onde começar e quais processos escolher para serem melhorados, a representação por estágios é a melhor opção, pois ela fornece um conjunto específico de processos para melhorar em cada estágio, determinado por mais de uma década de experiência e pesquisas em melhoria de processo.

11

4 - QUAL CAMINHO PERCORRER?

Atualmente as empresas que buscam certificações nesses modelos de maturidade para participação do mercado nacional e internacional, além da evolução de seus processos, devem apoiar essas melhorias aos objetivos estratégicos da organização. Vale ressaltar que o sucesso depende também dos fatores culturais da empresa, importantíssimo o acultramento da empresa para uma implantação com sucesso de qualquer modelo ou programa de melhoria de processos.

Segundo o CMMI, pesquisas têm mostrado que a melhor maneira de iniciar a melhoria de processo é a construção de um sólido suporte organizacional por meio de um forte patrocínio da gerência sênior.

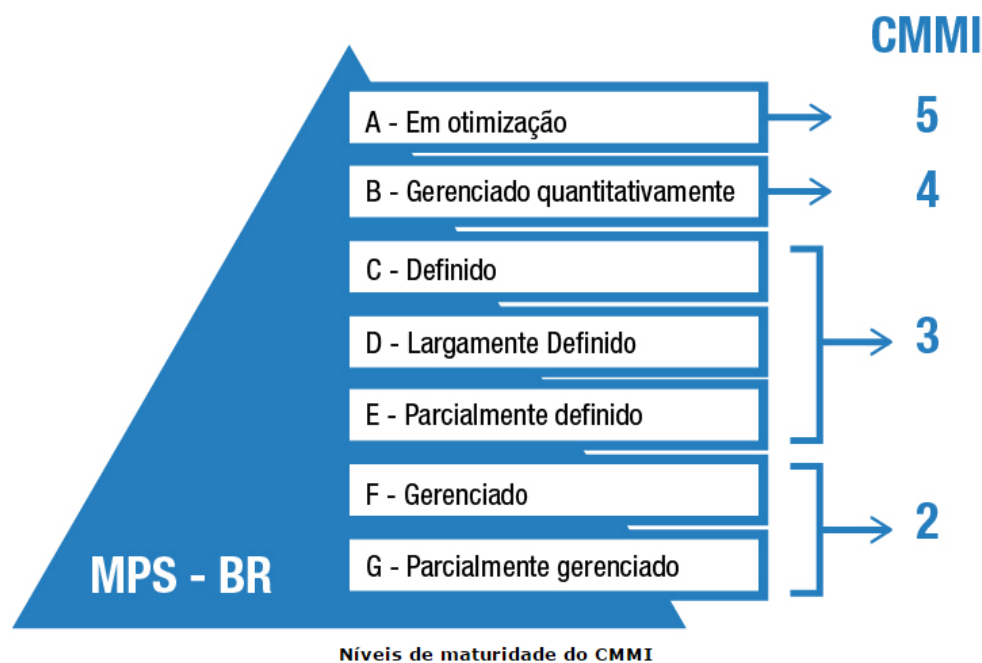
Para conseguir esse patrocínio, é interessante sensibilizá-la sobre os resultados de desempenho obtidos por aqueles que estão utilizando o CMMI para melhorar seus processos.

Para uma organização cuja missão é desenvolver predominantemente sistemas de *software*, o grupo de processos pode incluir engenheiros que representem as diferentes disciplinas técnicas existentes na organização e outros membros escolhidos com base nas necessidades de negócio que direcionam as melhorias.

12

O que o mercado exige das empresas?

Atualmente alguns órgãos do governo federal exigem em seus editais, especificamente contratos de desenvolvimento de *software* e suporte operacional dos sistemas, que as empresas possuam as certificações CMMI ou o MPS-BR. Sendo que a exigência de maturidade para o CMMI é o nível 3 e o MPS-BR é o nível C. Os dois modelos fornecem treinamentos e certificações individuais para os profissionais. A figura abaixo apresenta um comparativo dos níveis de maturidade dos dois modelos. Essa equivalência é entendida e aceita no mercado.



Os modelos de maturidade abrangem um campo no mercado que pode ser explorado e tem boas oportunidades de emprego nesse setor. Empresas grandes necessitam e precisam de equipes específicas para tratar de melhoria de seus processos e de pessoas com o conhecimento nesses modelos de maturidade.

13

RESUMO

Na atualidade as metodologias podem ser utilizadas para a construção de qualquer *software*. O que vai caracterizar o uso de uma ou outra é a metodologia que tem um modelo que mais se adéque ao tipo de linguagem, equipe, característica do produto e claro a filosofia e práticas da empresa que está desenvolvendo. A Engenharia de *Software* surgiu para estudar e detalhar as disciplinas e todos os meandros que são utilizadas no desenvolvimento de nossos sistemas. Ela dá todo o suporte para a adequação das metodologias e a sua utilização para a construção dos sistemas vem contribuindo para o aumento da competitividade da empresa.

Para evoluir os processos da empresa e para garantir a eficiência e eficaz do seu desenvolvimento, buscou, no mercado, conceitos e práticas, conceitos esses que são modelos de referência que medem a maturidade dos processos de *softwares*.

Esses modelos de maturidade são respeitados no Brasil e no mundo e foram construídos e são mantidos por grupos em constante estudo que buscam a excelência no desenvolvimento de *software* e suporte a serviços.

UNIDADE 1 – METODOLOGIAS DE DESENVOLVIMENTO DE SISTEMAS

MÓDULO 4 – FASES DAS PRINCIPAIS METODOLOGIAS DE DESENVOLVIMENTO DE SOFTWARE

01

1 - CONCEITOS IMPORTANTES NO DESENVOLVIMENTO DE PROJETO E MANUTENÇÃO DE SISTEMAS

Nesse módulo iremos ver mais detalhes do processo unificado e métodos ágeis, mais precisamente falaremos do RUP (Processo Unificado Rational) e do SCRUM.

No mercado de trabalho são utilizados alguns termos que percorrem todo o desenvolvimento do sistema, áreas da qualidade, equipe de melhoria, ou seja, termos esses importantes para não ficar perdido em uma reunião de TIC. Abordaremos os seguintes termos: Papéis, Artefatos, Atividades, Fluxos de atividades, Projeto, Sistema e Manutenção.

| | | |
|-----------------------|---|---|
| • Papéis | → | É a descrição do comportamento e das responsabilidades de um indivíduo ou grupo de indivíduos trabalhando juntos em uma equipe. Os papéis são funções que executam atividades (fluxo de atividades) gerando artefatos ao longo do projeto ou manutenção. |
| • Artefatos | → | É uma informação que é produzida, modificada ou utilizada no processo de desenvolvimento. Ex.: Especificação de Caso de uso. |
| • Atividades | → | Define o trabalho que deve ser executado por cada papel ou função (pessoas) para produzir um resultado significativo no processo de construção. As atividades são associadas aos papéis específicos. Ex.: especificar o Caso de Uso – papel analista de sistemas. |
| • Fluxo de Atividades | → | Caminho percorrido para execução das atividades. Ex.: primeiro faz a elicitação dos requisitos, depois especifica os casos de uso. |
| • Projeto | → | É um empreendimento temporário com objetivo de criar um produto ou serviço único, ou seja, o projeto tem data de início e fim e tem como objetivo criar algo diferente dos outros existentes. |
| • Sistema | → | Algumas empresas conceituam sistemas como projetos prontos, ou seja, finalizou o projeto virou sistema. Seguiremos nessa simples conceituação nesse momento. |
| • Manutenção | → | São mudanças e implementações nos sistemas, podem ser melhorias das funcionalidades ou adaptações necessárias. |

02

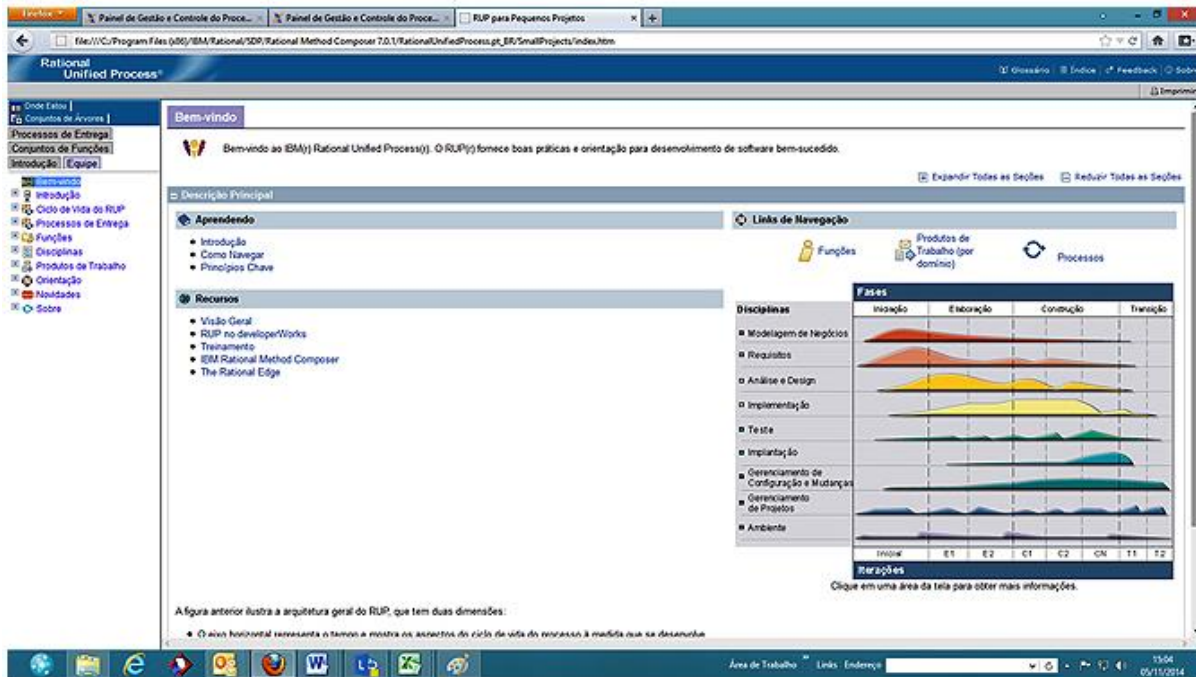
2 - PROCESSO UNIFICADO RACIONAL – RUP

O principal objetivo do RUP é atender as necessidades dos usuários, garantindo uma produção de *software* de alta qualidade, que cumpra um cronograma e um orçamento previsíveis.

Assim, o RUP mostra como o sistema será construído na fase de implementação, gerando o modelo do projeto e, opcionalmente, o modelo de análise que é utilizado para garantir a robustez. O RUP define perfeitamente quem é responsável pelo que, como as coisas deverão ser feitas e quando devem ser realizadas, descrevendo todas as metas de desenvolvimento especificamente para que sejam alcançadas.

Segundo definição do próprio RUP, versão 7.0, em sua essência diz respeito a um desenvolvimento de software bem-sucedido. Há três elementos centrais que definem o RUP:

- Um conjunto subjacente de filosofias e princípios para desenvolvimento de *software* bem-sucedido.
- Uma estrutura de conteúdo de método reutilizável e blocos de construção de processo.
- O método subjacente e a linguagem de definição do processo.



Página principal do RUP 7.0

Um conjunto subjacente de filosofias e princípios para desenvolvimento de software bem-sucedido. Estas filosofias e princípios são a base na qual o RUP foi desenvolvido. Para obter uma visão geral da filosofia do RUP.

Uma estrutura de conteúdo de método reutilizável e blocos de construção de processo. Definida e aprimorada continuamente pelo Rational Software, a família RUP de *plug-ins* de método define uma estrutura de método, a partir da qual você cria suas próprias configurações de método e processos ajustados.

O método subjacente e a linguagem de definição do processo. Inerente a tudo isso está um metamodelo de arquitetura de método unificado. Este modelo fornece uma linguagem para descrever o conteúdo do método e os processos. Essa nova linguagem é uma unificação de diferentes métodos e linguagens de engenharia do processo.

03

Ferramenta de apoio

Durante muitos anos de esforço de desenvolvimento, o RUP evoluiu em uma plataforma de engenharia de processo, chamada **RMC (Rational Method Composer)**, que é a ferramenta ou *software* utilizado para construir o RUP. O RMC permite que equipes definam, configurem, ajustem e pratiquem um processo consistente. O site é criado utilizando páginas HTML gerado dinamicamente, as quais o RMC

permite que sejam publicadas na forma de várias páginas dentro do RUP, cada um representando uma definição de processo configurada e ajustada.

O RUP fornece um local para os engenheiros (especialistas) de processo na comunidade de desenvolvimento de *software*, para compartilharem suas extensões de método como Plug-Ins consumíveis e fornece uma origem valiosa de extensões de método para o coordenador de projeto.

O produto RUP foi desenvolvido visando dois grupos primários de usuários:



- profissionais de desenvolvimento de *software* que trabalham como parte de uma equipe de projeto, incluindo os investidores desses projetos de desenvolvimento de *software*;



- profissionais de engenharia de processo (equipes da Qualidade), especificamente engenheiros de processo de *software* e gerentes.

Profissionais de desenvolvimento de *software*

Os profissionais de desenvolvimento de *software* podem encontrar orientações sobre o que é exigido deles nas funções, também chamado de papéis, definidas no RUP. Um profissional que trabalha em um projeto de engenharia de *software* RUP é designado a uma ou mais funções definidas no RUP, em que cada função particiona um conjunto de tarefas e produtos de trabalho pelos quais essa função é responsável. Também é fornecida orientação sobre como essas funções trabalham em conjunto, em termos das atividades requeridas para representar o processo configurado.

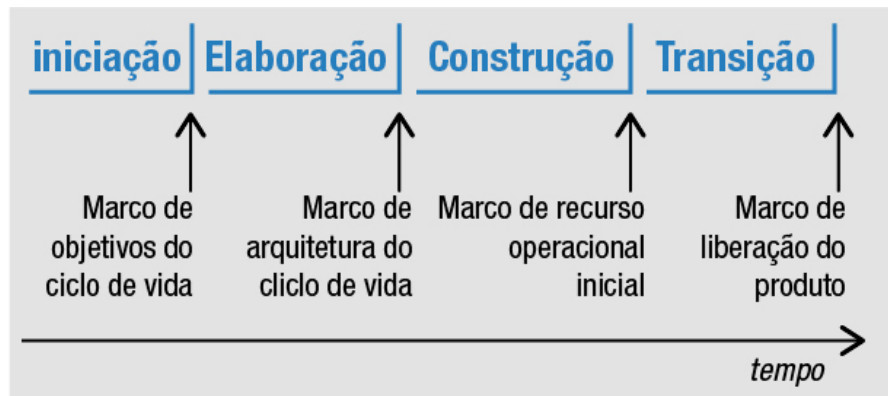
Profissionais de engenharia de processo

Os profissionais de engenharia de processo, ou equipes da qualidade podem obter orientação sobre como definir, configurar, adaptar e implementar os processos de engenharia. A família de produtos RUP fornece várias ferramentas que permitem e simplificam a definição, configuração e adaptação do processo de engenharia.

Fases do RUP

O desenvolvimento iterativo e incremental é uma das práticas núcleo do RUP. Comece a utilizar o RUP navegando pelo site e identificando o gráfico das baleias. Perceba que as disciplinas percorrem pelas fases. O aprendizado deve ser também incremental à medida que você vai navegando pelas disciplinas. Você poderá fazer o *download* do RUP na sua máquina através da internet. O RUP vem dentro do RMC, o qual a equipe poderá configurar de acordo com o projeto ou a metodologia da empresa.

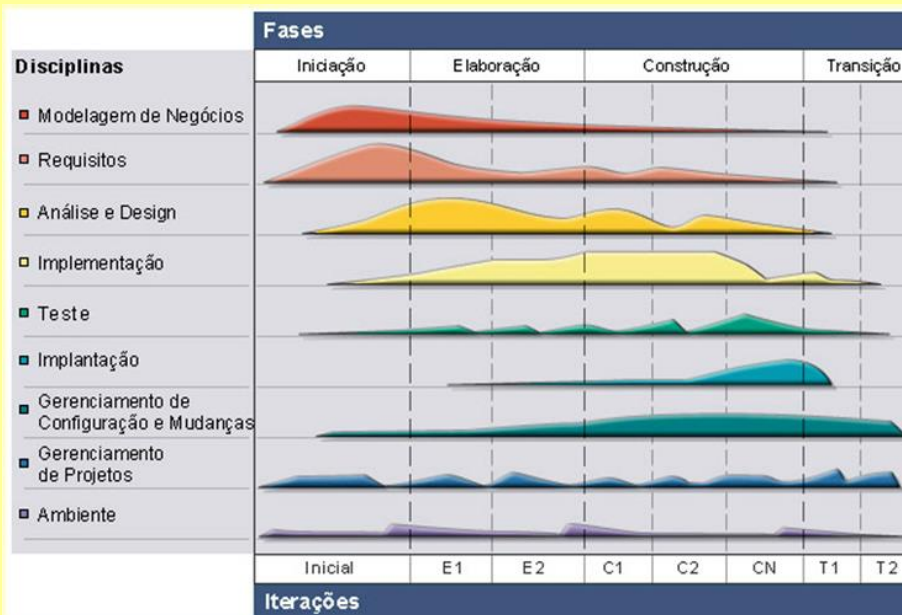
O RUP tem quatro fases: Iniciação, Elaboração, Construção e Transição.



Fases e os marcos de um projeto – RUP 7.0

Conforme o RUP, no ponto de vista do gerenciamento, o ciclo de vida de *software* é dividido em quatro fases sequenciais, cada uma concluída por um marco principal, ou seja, cada fase é basicamente um intervalo de tempo entre dois marcos principais. A cada final de fase, uma avaliação é executada para determinar se os objetivos da fase foram alcançados. Uma avaliação satisfatória permite que o projeto passe para a próxima fase.

Gráfico de Baleias



05

As fases não são idênticas (veja gráfico das baleias) em termos de programação e esforço. Embora isso varie muito de acordo com o projeto, um ciclo de desenvolvimento inicial típico para um projeto de médio tamanho deve prever a seguinte distribuição de esforço e programação:

| | Iniciação | Elaboração | Construção | Transição |
|------------|-----------|------------|------------|-----------|
| Esforço | ~5% | 20% | 65% | 10% |
| Cronograma | 10% | 30% | 50% | 10% |

Esta distribuição pode variar. Por exemplo, ferramentas que gerem código e etapas de teste podem diminuir a fase de construção. Além disso, para um ciclo de evolução ou manutenção, as fases de iniciação e de elaboração seriam consideravelmente menores, já que uma visão e arquitetura básica já estão estabelecidas. São exceções a essa regra os ciclos de evolução, em que ocorre uma redefinição significativa do produto ou da arquitetura.

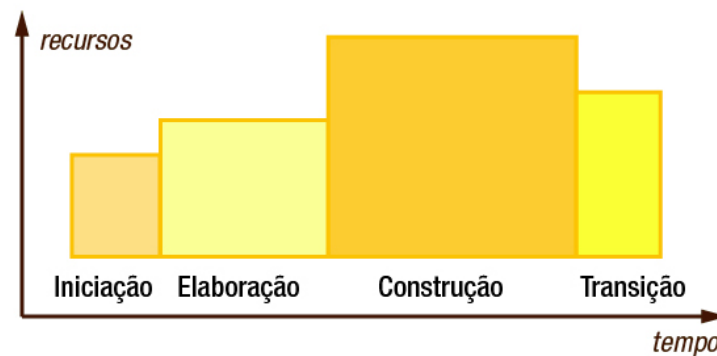
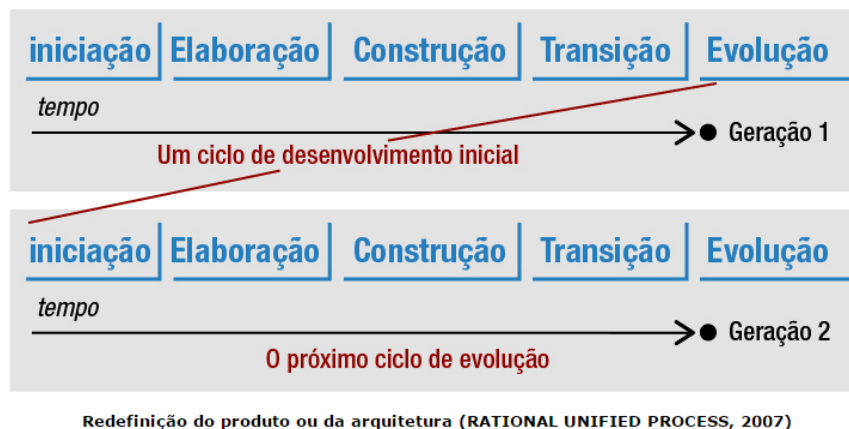


Gráfico esforço/tempo – RUP 7.0

06

Ciclos de desenvolvimento

Segundo o RUP, uma passagem pelas quatro fases é um **ciclo de desenvolvimento** e cada passagem pelas quatro fases produz uma geração do *software*. A menos que o produto "desapareça", ele irá se desenvolver na próxima geração, repetindo a mesma sequência de fases de iniciação, elaboração, construção e transição, mas agora com ênfase diferente nas diversas fases e disciplinas. Esses ciclos subsequentes são chamados **ciclos de evolução** ou **versões**. À medida que o produto atravessa vários ciclos, são produzidas novas versões.



As versões podem ser solicitadas por melhorias sugeridas pelos usuários, mudanças no contexto do usuário, mudanças na tecnologia, reação à concorrência e assim por diante.

A maioria das metodologias ágeis tem como base o processo unificado na sua essência, iterativo e incremental.

07

3 - METODOLOGIA ÁGIL – SCRUM

SCRUM é um conjunto de ferramentas e técnicas, ou simplesmente framework, usado para o desenvolvimento ágil. Teve a sua origem em 1986, onde seus autores (Hirotaka Takeuchi e Ikujiro Nonaka) fizeram uma analogia ao rugby, por ser veloz, flexível e ser dependente da realização em trabalho de equipe.

Segundo o guia do Scrum, que é fundamentado na **teoria de controle de processos empíricos**, emprega uma abordagem iterativa e incremental para otimizar a previsibilidade e controlar riscos.

Três pilares sustentam qualquer implementação de controle de processos empíricos, ou seja, processos que já foram executados (experiência) e podem ser melhorados. São os seguintes os pilares:

| O primeiro pilar é a TRANSPARÊNCIA | O segundo pilar é a INSPEÇÃO | O terceiro pilar é a ADAPTAÇÃO |
|--|---|--|
| A transparência garante que aspectos do processo que afetam o resultado devem ser visíveis para aqueles que gerenciam os resultados. Esses aspectos não apenas devem ser transparentes, mas também o que está sendo visto deve ser conhecido. Isto é, quando alguém que inspeciona um processo acredita que algo está pronto, isso deve ser equivalente à definição de pronto utilizada. | Os diversos aspectos do processo devem ser inspecionados com uma frequência suficiente para que variações inaceitáveis no processo possam ser detectadas. A frequência da inspeção deve levar em consideração que qualquer processo é modificado pelo próprio ato da inspeção. Os outros fatores são a habilidade e a aplicação das pessoas em inspecionar os resultados do trabalho. | Se na inspeção determinar, que um ou mais aspectos do processo estão fora dos limites aceitáveis e que o produto será inaceitável, ele deverá ajustar o processo e realizar as adaptações nos sprints. Essas adaptações devem ser feitas o mais rápido possível para minimizar desvios posteriores. Na revisão do Sprint pode definir que adaptações tornarão a próxima Sprint mais produtivo. |

Fases do SCRUM

As fases ocorrem dentro de uma sequência rígida, com o início das atividades de uma etapa acontecendo imediatamente após o término da atividade anterior. São as fases do SCRUM:

- pré-game,
- game e
- pós-game.

A fase determinada como **pré-game** é quando é realizado o planejamento do projeto. No planejamento é definida a equipe do projeto, o escopo do projeto, que possibilita o desenvolvimento da lista de funcionalidade a serem executadas. No SCRUM chamará essa lista de Backlog do Produto, com esse documento a equipe conhecerá toda a lista de requisitos que serão necessárias para construção do sistema e suas prioridades, associado ao valor de negócio para que se consiga calcular o retorno do projeto para o negócio.

Um conceito importante antes de avançarmos no SCRUM é o conceito de **Sprint**, que dá a ideia de rápido, velocidade. Como citado no início, o SCRUM trabalha com iterações, essas séries de iterações são chamadas de ciclos de desenvolvimento ou simplesmente *Sprint* (*Sprint* estoque), que formam as funcionalidades dos sistemas quando integradas.

Na fase determinada como **game** ocorre o desenvolvimento dos *sprints*, reforçando que o Scrum como o RUP prima pelo desenvolvimento iterativo e incremental. Significa que ciclos de desenvolvimento acontecem no projeto, ou seja, um conjunto específico de atividades é repetido continuamente ao longo de um projeto, onde ocorrem entregas sucessivas de funcionalidades em intervalos de tempos.

Além dos *sprints* ocorrem também as reuniões de revisões dos *sprints*, planejamento do *sprint* e uma reunião de retrospectiva, que possibilita o próximo *sprint* mais produtivo.

Outro conceito do Scrum importante é o de **Time-Box**, que é a quantidade de tempo estipulado para a realização de um *sprint*.

Finalmente, a fase de pós-game é o encerramento do projeto, onde ocorre a aprovação do produto criado.

Intervalos de tempos

Esse intervalo de tempo sugere que seja cada sprint de 2 a 4 semanas. Sendo que essa estimativa de tempo (2 a 4 semanas) não pode ser alterada, a fim de com isto garantir a entrega sem atrasos e facilitar o planejamento. Caso ocorra algum erro na estimativa de tempo, a sugestão é que se diminua o escopo do sprint, para não obrigar o aumento da quantidade de tempo planejado para concluir.

Papéis importantes no SCRUM

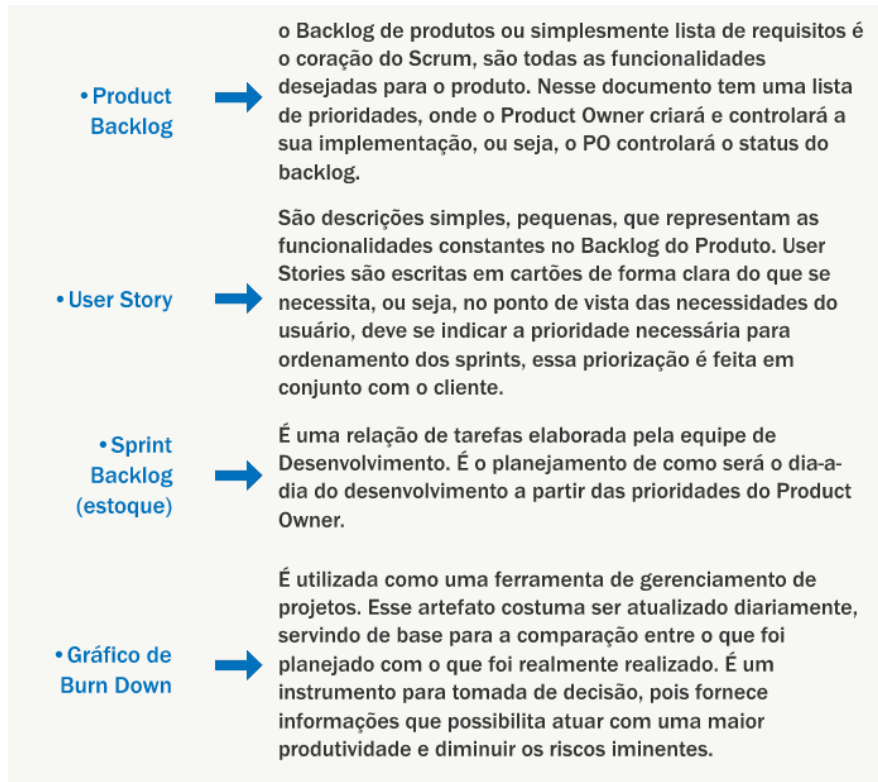
Um conjunto de papéis bem definidos determina, através de atribuições, como será a atuação dos diversos profissionais que participarão de um projeto:

| | |
|--|---|
| PRODUCT OWNER | É o dono do produto, ele possui a visão do que será o projeto e quais os benefícios trará para a empresa. Sua missão é cuidar do Backlog de produtos, planejar, priorizar os requisitos e passar a equipe uma visão clara dos objetivos do projeto. |
| SCRUM MASTER | Pessoa responsável em garantir que o processo seja entendido e seguido, o Scrum Master verifica se a equipe está seguindo as regras do Scrum. Não é o gerente de projetos, pois a equipe é autogerenciável, ele deve fazer com que a equipe entenda e use o autogerenciamento e a interdisciplinaridade. Apesar disso, muitas empresas tratam o Scrum Master como se fosse o gerente de projetos, mas a responsabilidade dele é manter o foco no processo, remover os impedimentos da equipe e auxiliar na comunicação entre a equipe e o dono do produto. No autogerenciamento todos são responsáveis e buscam a agilidade no processo, para isso a equipe deve contar com profissionais de nível sênior, caso contrário, é difícil ter sucesso. |
| EQUIPE DE DESENVOLVIMENTO/ TIME | São os analistas, desenvolvedores, testadores de software, ou seja, os profissionais responsáveis pela criação do produto baseado nas prioridades do Backlog. Como mencionado anteriormente, a equipe deve ser formada por pessoas com conhecimento sênior, além de serem multifuncionais, autogerenciáveis e comprometidas com o projeto. |

Além dos três papéis descritos, certamente também existirão os *stakeholders*, que são outros envolvidos no projeto, mas que não desempenham um papel direto na sua execução. Essas pessoas podem ser usuários, gerentes, diretores ou pessoas que são afetadas pelos resultados do produto final.

Artefatos de Scrum

Por mais que Scrum priorize a entrega do *software* e conceitualmente possui menos documentos que as outras metodologias, a construção e manutenção de alguns documentos neste modelo são de fundamental importância para o controle das atividades. A seguir os documentos do Scrum:



Quando o uso de metodologia atrapalha o desenvolvimento do *software*?

Os processos de desenvolvimento é um balizador para o bom andamento da construção dos sistemas. A metodologia apoia todo o desenvolvimento para que se consiga o sucesso da entrega. O grande problema é quando a metodologia é utilizada de forma burocrática. A metodologia veio para ajudar e agilizar o desenvolvimento, e não para dificultar e atrasar. Essa linha de raciocínio deve ser seguida por todos os técnicos, sendo das equipes de desenvolvimento e das equipes da Qualidade. Um exemplo de exigência de burocracia é parar o processo por causa da recusa de documentos do projeto devido a palavras sem acentuação ou em formato errado. Isso pode gerar prejuízos enormes no custo e nos prazos do projeto. Nesse caso, o ideal é recomendar a correção sem parar o processo.

11

RESUMO

No desenvolvimento de sistemas não existe um truque que facilite a condução de atividades. Diversas metodologias foram propostas ao longo da era da TI, sendo que nenhum destes modelos representa a solução dos problemas para o desenvolvimento dos *softwares*. Atualmente temos modelos que estão na moda e outros que estão se emergindo no mundo TIC. O Processo Unificado, mais necessariamente o RUP (Processo Unificado da Rational) veio e se mantém em boa parte das empresas. Sua estrutura baseada em FrameWork apoia o desenvolvimento desde o início até o fim do projeto. O Modelo

iterativo e incremental estabelece uma tranquilidade para o cliente, pois vai vendo o seu produto nascer e crescer com o passar do desenvolvimento.

O mercado exige cada vez mais rapidez nas entregas, pois atraso no projeto ou um desenvolvimento normal pode ser tardio para o lançamento de produtos no mercado. As metodologias ágeis surgiram para acelerar o desenvolvimento dos sistemas. O Scrum é uma das principais metodologias, que simplificou o processo de desenvolvimento, mantendo a qualidade da aplicação e respondendo mais rapidamente os anseios do mercado.

Sendo uma metodologia ou outra, o importante que a empresa tenha seus processos definidos para o desenvolvimento dos sistemas. Além disso, um padrão adequado ao gerenciamento de um determinado sistema pode-se revelar como um problema para outro sistema.

Assim tanto o RUP como o SCRUM permitem algum tipo de adaptações em seus processos para melhor atender o projeto e a empresa, desde que não percam a suas essências.