

Engenharia de Software

**Aula: Introdução a
Engenharia de Software e
Conceitos Iniciais.**



Prof. Anderson Augusto Bosing

O que é **Software**?

Consiste em uma série de programas separados + arquivos de configuração + documentação do sistema + documentação do usuário + ,se for o caso, sites na WEB para os usuários fazerem download de informações recentes. Adaptado Sommerville (2014).

Pressman (2011) afirma que todo projeto de software nasce de uma necessidade de negócio, seja para adaptar algo já existente ou para criar um novo produto ou serviço.



O que é **Engenharia de Software**?

É uma disciplina da engenharia que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação.
Sommerville (2014).

Engenharia de Software é o estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais.
Pressman(2011)



O que é **Engenharia de Software**?

- Em suma, desenvolver software é uma atividade que não se confunde com escrever programas para computador.
- Desenvolvimento de software envolve procedimentos que exigem a abordagem simultânea e integra de aspectos técnicos e gerenciais.
- Se ocupa de todos os aspectos da produção do software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema.
- Trabalha com técnicas, teorias, métodos e ferramentas que auxiliam na produção do software.
- É uma área do conhecimento voltada para:
 - ✓ Especificação de software
 - ✓ Desenvolvimento de software
 - ✓ Validação de software
 - ✓ Evolução de software.



Prof. Anderson Augusto Bosing

O que é Engenharia de Software?

Em resumo é uma disciplina de engenharia que se preocupa com todos os aspectos de produção de software.



Prof. Anderson Augusto Bosing

Mas por que Engenharia?

Pois deve ser um processo sistêmico e regrado.



Prof. Anderson Augusto Bosing

Qual a diferença entre engenharia de software e ciência da computação?

Ciência da computação foca a teoria e os fundamentos; engenharia de software preocupa-se com o lado prático do desenvolvimento e entrega de softwares úteis.



Prof. Anderson Augusto Bosing

Quais são os principais desafios da engenharia de software?

**Lidar com o aumento de diversidade, demandas
pela diminuição do tempo para entrega e
desenvolvimento de software confiável.**



Prof. Anderson Augusto Bosing

Quais **diferenças** foram feitas pela Internet na **engenharia** **de software?**

A Internet tornou serviços de software disponíveis e possibilitou o desenvolvimento de sistemas altamente distribuídos baseados em serviços. O desenvolvimento de sistemas baseados em Web gerou importantes avanços nas linguagens de programação e reuso de software.

- ✓ Pagamentos por uso de acordo com o uso.
- ✓ Software as a Service(SaaS).
- ✓ Sistemas baseadas na web, em vez de sistemas locais.
- ✓ WebServices e API's possibilitam que funcionalidades sejam acessadas via internet.



Quais são os atributos de um bom software??

Um bom software deve prover a funcionalidade e o desempenho requeridos pelo usuário; além disso, deve ser confiável e fácil de manter e usar.



Prof. Anderson Augusto Bosing

Quais são os atributos essenciais de um bom software?

Manutenibilidade

O software deve ser escrito de forma que possa evoluir para atender às necessidades dos clientes. Esse é um atributo crítico, porque a mudança de software é um requisito inevitável de um ambiente de negócio em mudança



Prof. Anderson Augusto Bosing

Quais são os atributos essenciais de um bom software?

Confiança e proteção

A confiança do software inclui uma série de características como confiabilidade, proteção e segurança. Um software confiável não deve causar prejuízos físicos ou econômicos no caso de falha de sistema. Usuários maliciosos não devem ser capazes de acessar ou prejudicar o sistema.



Prof. Anderson Augusto Bosing

Quais são os atributos essenciais de um bom software?

Eficiência

O software não deve desperdiçar os recursos do sistema, como memória e ciclos do processador. Portanto, eficiência inclui capacidade de resposta, tempo de processamento, uso de memória etc.



Prof. Anderson Augusto Bosing

Quais são os atributos essenciais de um bom software?

Aceitabilidade

O software deve ser aceitável para o tipo de usuário para o qual foi projetado. Isso significa que deve ser compreensível, usável e compatível com outros sistemas usados por ele.



Prof. Anderson Augusto Bosing

Engenheiro de Software

A função de um profissional como Engenheiro de Software:

- ✓ Atuar no desenvolvimento de aplicativos para dispositivos móveis, como smartphones, tablets, jogos e softwares.
- ✓ Atuar na área de Gestão que é o gerenciamento de negócios e projetos de empresas de computação e software.
- ✓ Atuar em projetos, desenvolvimento, implantação e evolução de softwares complexos, corretos, disponíveis, seguros e tolerantes a falhas e com usabilidade.
- ✓ Desenhar, especificar, programar e testar soluções que atendam às necessidades do mercado, da sociedade, das organizações e dos indivíduos, levando em conta os impactos sociais.



Prof. Anderson Augusto Bosing

Engenheiro de Software

O profissional Engenheiro de Software produz um software que podemos dividir em:

Produtos genéricos - software de caixa ou de prateleira.

A especificação do que o software deve fazer é de propriedade do desenvolvedor de software e as decisões sobre as mudanças de software são feitos pelo desenvolvedor.

Produto sob encomenda - quando um cliente solicita a uma empresa de software que desenvolva um software específico às necessidades da empresa.

A especificação do que o software deve fazer é propriedade do cliente para o software e eles tomam decisões sobre as mudanças necessárias no software.



Prof. Anderson Augusto Bosing

Tipos de Aplicações de Software

Aplicações stand-alone.

Essas são as aplicações executadas em um computador local, como um PC. Elas contêm toda a funcionalidade necessária e não precisam estar conectadas a uma rede. Exemplos de tais aplicações são aplicativos de escritório em um PC, programas CAD, software de manipulação de fotos etc.

Aplicações interativas baseadas em transações

São aplicações executadas em um computador remoto e são acessadas pelos usuários a partir dos seus próprios PCs ou terminais. Essas incluem aplicações web tais como para e-commerce.



Tipos de Aplicações de Software

Sistemas de controle embutidos.

São sistemas de software de controle que controlam e gerenciam dispositivos de hardware. Numericamente, provavelmente existem mais sistemas embutidos do que qualquer outro tipo de sistema.

Sistemas de processamento de lotes

São sistemas corporativos projetados para processar dados em grandes lotes. Eles processam grande número de entradas individuais para criar as saídas correspondentes. Exemplos de sistemas de lotes incluem sistemas periódicos de cobrança, como sistemas de cobrança telefônica, e sistemas de pagamentos de salário.



Prof. Anderson Augusto Bosing

Tipos de Aplicações de Software

Sistemas de entretenimento.

São sistemas cuja utilização principal é pessoal e cujo objetivo é entreter o usuário. A maioria desses sistemas é de jogos de diferentes tipos. A qualidade de interação com o usuário é a característica particular mais importante dos sistemas de entretenimento.

Sistemas para modelagem e simulação.

São sistemas que incluem vários objetos separados que interagem entre si, desenvolvidos por cientistas e engenheiros para modelar processos ou situações físicas. Esses sistemas geralmente fazem uso intensivo de recursos computacionais e requerem sistemas paralelos de alto desempenho para executar.



Prof. Anderson Augusto Bosing

Tipos de Aplicações de Software

Sistemas de coleta de dados.

São sistemas que coletam dados de seu ambiente com um conjunto de sensores e enviam esses dados para outros sistemas para processamento. O software precisa interagir com sensores e frequentemente é instalado em um ambiente hostil, por exemplo, dentro de uma máquina ou em um lugar Remoto.

Sistemas de sistemas.

São sistemas compostos de uma série de outros sistemas de software. Alguns deles podem ser produtos genéricos de software, como um programa de planilha eletrônica. Outros sistemas do conjunto podem ser escritos especialmente para esse ambiente.



Prof. Anderson Augusto Bosing

PONTOS IMPORTANTES

- A engenharia de software é uma disciplina de engenharia que se preocupa com todos os aspectos de produção de software.
- Atributos essenciais do produto de software são manutenibilidade, confiança, proteção, eficiência e aceitabilidade.
- Existem muitos tipos diferentes de sistemas e cada um requer ferramentas de engenharia de software e técnicas apropriadas para o seu desenvolvimento.



Prof. Anderson Augusto Bosing

VAMOS PRATICAR ?

Aponte a câmera do seu
smartphone.



Prof. Anderson Augusto Bosing

Ética na Engenharia de Software

- A engenharia de software envolve responsabilidades mais amplas do que a simples aplicação de habilidades técnicas.
- Engenheiros de software devem se comportar de uma maneira honesta e eticamente responsável para serem respeitados como profissionais.
- Comportamento ético é mais do que simplesmente agir em concordância com a lei, envolve seguir um conjunto de princípios moralmente corretos.



Prof. Anderson Augusto Bosing

Questões de responsabilidade profissional

- **Confidencialidade**

Os engenheiros devem respeitar a confidencialidade de Seus empregadores ou clientes, independentemente de haver ou não um acordo de confidencialidade formal assinado entre eles.

- **Competência**

Engenheiros não devem falsear seus níveis de competência. Eles não devem aceitar trabalhos que estão fora da sua competência.



Prof. Anderson Augusto Bosing

Questões de responsabilidade profissional

- **Direitos de propriedade intelectual**

Engenheiros devem estar cientes das leis locais que regulam a propriedade intelectual, tais como patentes, direitos autorais, etc. Eles devem ser cuidadosos para assegurar que a propriedade intelectual dos empregadores e clientes esteja protegida.

- **Uso indevido de computador**

Engenheiros de software não devem usar suas habilidades técnicas para uso indevido de computadores de outras pessoas. A variação do mau uso do computador vai desde relativamente trivial (brincar com jogos na máquina de um empregador, por exemplo) a extremamente sérios (disseminação de vírus).



VAMOS PRATICAR ?

Aponte a câmera do seu
smartphone.



Prof. Anderson Augusto Bosing

Engenharia de Software

**Aula: Conceitos de
Sistemas da Informação,
Mitos da Engenharia de
Software.**



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Gerenciamento

Mito 1. "Se a equipe dispõe de um manual repleto de padrões e procedimentos de desenvolvimento de software, então a equipe está apta a encaminhar bem o desenvolvimento. A equipe já não tem tudo o que ela precisa saber?"

Realidade: o manual é útil, confuso, muitas vezes, fica defasado rapidamente, não é tão completo quanto deveria, não é adaptável a todos os projetos ou, ainda, não busca melhorar o tempo de entrega com foco na qualidade, sendo assim, o manual apenas não é o suficiente.



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Gerenciamento

Mito 2. "Se o desenvolvimento do software estiver atrasado, basta aumentar a equipe para honrar o prazo de desenvolvimento".

Realidade: Existe um custo em tempo para que um desenvolvedor passe a render, e os desenvolvedores que já estão no projeto deverão parar para auxiliar os novos colaboradores. Certas coisas exigem tempo, porém, quando bem planejadas, a inclusão de novos desenvolvedores ajudará.



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Cliente

Mito 3. “Uma descrição breve e geral dos requisitos do software é o suficiente para iniciar o seu projeto... maiores detalhes podem ser definidos posteriormente”.

Realidade: quanto maior a definição dos requisitos, mais correto será o desenvolvimento e menor será o retrabalho.



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Desenvolvedor

Mito 4. “Após a edição do programa e a sua colocação em funcionamento, o trabalho está terminado.”

Realidade: O que ocorre na realidade é completamente diferente disto. Segundo dados obtidos que 50 a 70% do esforço de desenvolvimento de um software é despendido após a sua entrega ao cliente (manutenção).



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Desenvolvedor

Mito 5. “A equipe de desenvolvimento entende que, enquanto o software não estiver em uso, não é possível validar sua qualidade.”

Realidade: Inúmeros tipos de testes podem e devem ser feitos durante o desenvolvimento, visando encontrar a menor quantidade possível de erros durante a fase de uso.



Prof. Anderson Augusto Bosing

Mitos da Engenharia de Software

Mito do Desenvolvedor

Mito 6. “A engenharia de software obriga a equipe de desenvolvimento a entregar uma documentação volumosa e trabalhosa de ser criada.”

Realidade: A engenharia de software visa à entrega de um produto de qualidade, e não à entrega de uma documentação volumosa.



Prof. Anderson Augusto Bosing

O que é Sistema da Informação?

Sistema de Informação, sigla S.I., é um conjunto de componentes inter-relacionados (pessoas, hardware, software, redes de comunicações e recursos de dados) que coletam (ou recuperam), processam, armazenam e distribuem informações destinadas a apoiar a tomada de decisões, a coordenação e o controle de uma organização. (LAUDON; LAUDON, 2004, p. 7).



Conceitos de Informação

DADOS são fatos que ainda não foram trabalhados: Nome de Empregado e Registro de Identidade.

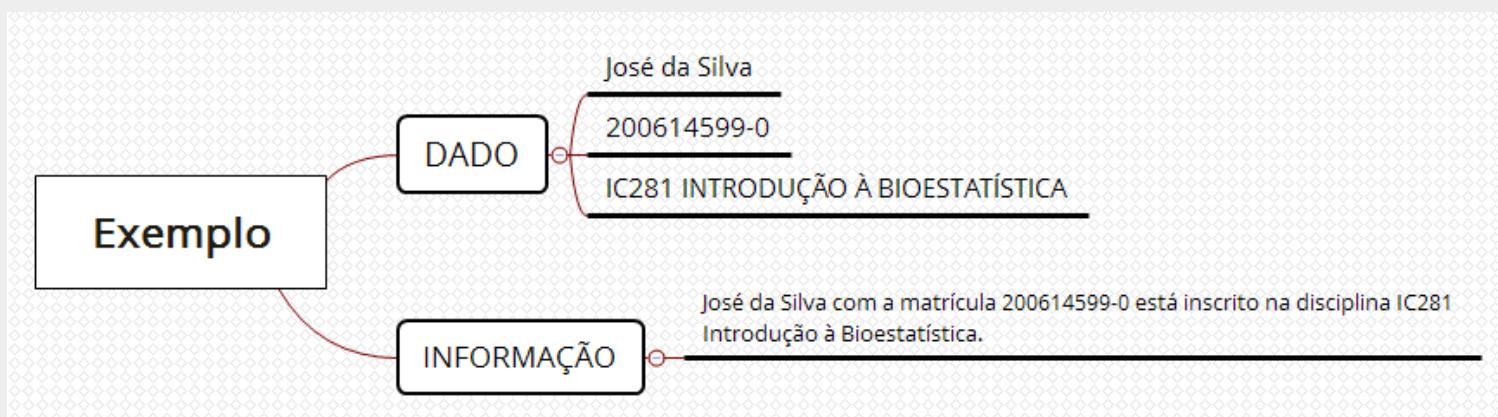


INFORMAÇÃO são dados organizados ou ordenados de forma significativa de modo que adquirem um valor adicional. Ex: Total de horas trabalhadas em uma semana, Estoque médio mensal, etc.

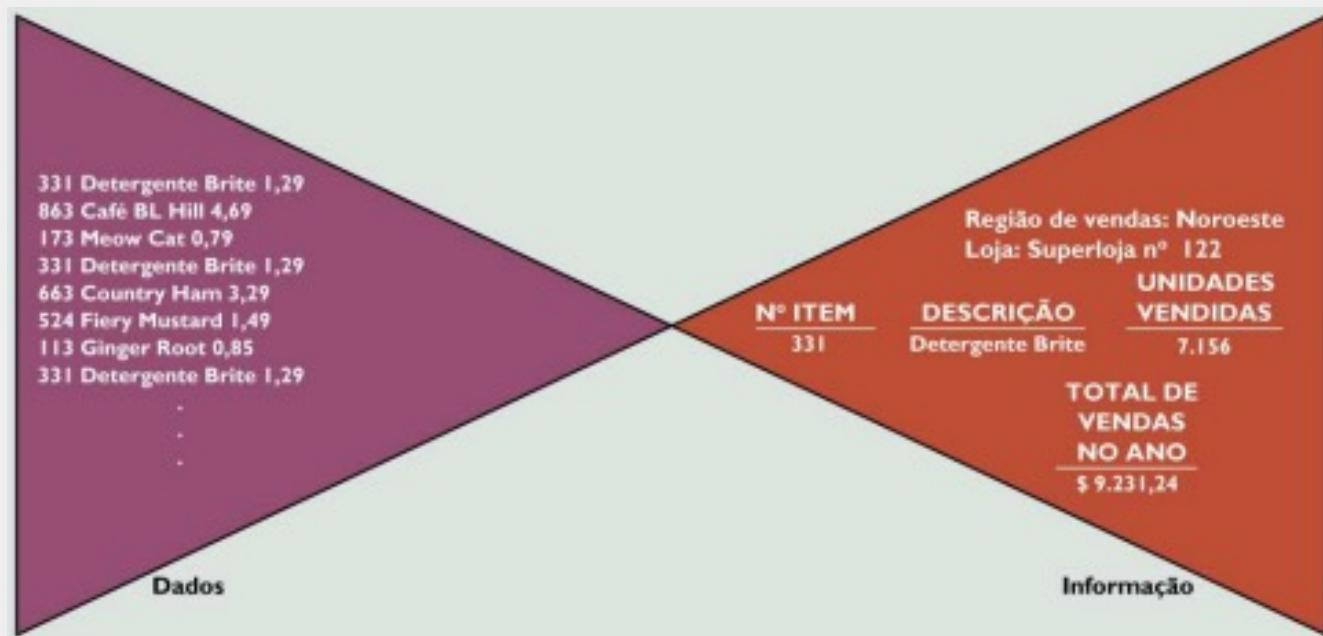
Conceitos de Informação

A transformação de dados em informação é tarefa principal dos sistemas de informação.

A transformação de dados em informação é um processo ou uma série de tarefas logicamente relacionadas, executadas para atingir um resultado definido



Conceitos de Informação



Conceitos de Informação

O processo de definição das relações entre dados requer conhecimento.

- Conhecimento são as regras, diretrizes, procedimentos usados para selecionar, organizar e manipular os dados, com a finalidade de torná-los úteis para uma tarefa específica.



Através da informação as pessoas adquirem conhecimento.



Prof. Anderson Augusto Bosing

Conceitos de Informação



Entrada: envolve captação e reunião de elementos que entram no sistema para serem processados. Ex. matéria-prima, energia, dados e esforço humano.

Processamento: envolve processos de transformação que converte insumo (entrada) em produto. Ex. controle de estoque, respiração humana ou cálculos matemáticos.

Saída: envolve a transferência de elementos produzidos por um processo de transformação até o seu destino final. Ex. produtos acabados, serviços, informações gerenciais.

Conceitos de Informação



Ambiente: é o conjunto de elementos que não pertencem ao sistema, mas qualquer alteração no sistema pode mudar ou alterar os seus elementos e qualquer alteração nos seus elementos pode mudar ou alterar o sistema.

Realimentação: pode ser considerado como a reintrodução de uma saída sob forma de informação. Ou ainda, é uma saída usada para fazer ajustes ou modificações nas atividades de entrada ou processamento.

Exemplo: Folha de Pagamento

Entrada:

- As horas trabalhadas pelos funcionários devem ser levantadas (captadas) através dos cartões de horas do funcionário.

Processamento:

- As horas trabalhadas de cada funcionário devem ser convertidas em pagamento líquido.
- Multiplicação das horas trabalhadas pela taxa de pagamento por hora do empregado, para se obter o pagamento bruto.
- Se o pagamento de horas semanais trabalhadas superar 40 horas, o pagamento de horas extras também pode ser determinado.
- Então as deduções são subtraídas do pagamento bruto para se obter o pagamento líquido.

Saída:

- cheques de pagamentos a empregados, relatórios para gerentes e informações fornecidas para acionistas, bancos, agências governamentais e outros grupos.



Prof. Anderson Augusto Bosing

VAMOS PRATICAR ?

Desenvolvendo um SI para uma ONG

- Você está em contato com o coordenador de uma Organização Não Governamental chamada Parceiros Voluntários;
- A Idéia desta ONG é cadastrar profissionais voluntários (parceiros) e também instituições que necessitam dos serviços destes profissionais;
- Dessa forma, os profissionais cadastrados informariam sua especialidade, tempo e horário disponível, e as instituições a serem favorecidas pelos trabalhos voluntários, repassariam as informações necessárias para o encaminhamento de tais profissionais;
- O diretor da organização solicita o trabalho de sua equipe para desenvolver um Sistema de Informação para a ONG;
- Primeiramente descreva este sistema em termos de ENTRADAS – PROCESSAMENTOS e SAÍDAS, em seguida, identifique quais dados precisariam ser armazenados pelo sistema.



Prof. Anderson Augusto Bosing

Engenharia de Software

**Aula: Processo e Ciclo de
Vida de software.**



Prof. Anderson Augusto Bosing

Qual é o valor da informação para uma empresa?



Qual é o valor da informação para uma empresa?

Está surgindo um novo tipo de bem econômico: a **informação**.

A empresa que dispõe de mais informações sobre seu processo de negócio está em **vantagem**.

Em consequência, surgiu a necessidade de gerenciar informações de uma forma **adequada e eficiente**.



Qual é o valor da informação para uma empresa?

O **objetivo** principal e final da construção de um sistema de informações é a adição de valor à empresa.

Isso implica que a produtividade nos processos da empresa deve aumentar de modo significativo.

O sistema deve ser economicamente justificável.



Qual é o valor da informação para uma empresa?

- <https://adssettings.google.com/authenticated>



Qual é o valor da informação para uma empresa?

Uma característica é a complexidade de seu desenvolvimento, que aumenta à medida que o tamanho do sistema cresce.



Qual é o valor da informação para uma empresa?

Para a construção de sistemas de software mais complexos, é necessário um planejamento inicial. Equivalente ao projeto das plantas da engenharia civil também deve ser realizado. Essa necessidade leva ao conceito de modelo, tão importante no desenvolvimento de sistemas.



Gerenciamento de Complexidade

**Pode haver diversos modelos de um mesmo sistema,
cada qual descrevendo uma perspectiva.**

Exemplo: Um Carro pode ter um modelo para representar sua parte elétrica, outro para a parte aerodinâmica etc.



Gerenciamento de Complexidade

Por meio de modelos podemos fazer estudos e prever comportamentos do sistema em desenvolvimento.

Modelos revelam as características essenciais de um sistema. Detalhes não relevantes e que só aumentariam a complexidade do problema podem ser ignorados.



Comunicação entre as Pessoas Envolvidas

O desenvolvimento de um sistema envolve a execução de uma grande quantidade de atividades.

Os modelos de um sistema servem para promover a difusão de informações relativas ao sistema entre os indivíduos envolvidos em sua construção.



Redução dos Custos no Desenvolvimento

Seres humanos estão sujeitos a cometer erros.

Certamente a correção desses erros é menos custosa quando detectada é realizada ainda nos modelos do sistema.

O que é melhor?

Corrigir um erro na projeto de um carro ou recolher todos os carros depois de fabricados.



Previsão do Comportamento Futuro do Sistema

O comportamento do sistema pode ser discutido mediante uma análise dos seus modelos.

Os modelos servem como um “laboratório”, em que diferentes soluções para um problema relacionado à construção do sistema podem ser experimentadas.



Processo de software

Sommerville (2014) afirma que um “processo de software é um conjunto de atividades relacionadas que levam à produção de um produto de software”.

Pressman (2011) complementa dizendo que “processo é uma metodologia para as atividades, ações e tarefas necessárias para desenvolver um software de alta qualidade”.



Processo de software

- É uma **série de passos** (um **ROTEIRO**).
- Para criar **EM TEMPO** um **SOFTWARE** de **ALTA QUALIDADE**, sem estourar o **ORÇAMENTO**.
- Como “escolher” um processo?
- ✓ As **CARACTERÍSTICAS DA APLICAÇÃO** (domínio do problema, tamanho, complexidade etc);
- ✓ A **TECNOLOGIA** a ser adotada na sua construção (paradigma de desenvolvimento, linguagem de programação, mecanismo de persistência etc), a organização;
- ✓ **ONDE** o produto será desenvolvido;
- ✓ **O PERFIL DA EQUIPE** de desenvolvimento.



Prof. Anderson Augusto Bosing

Engenharia de Software - Ciclo de vida

- Independente da metodologia que será utilizada para desenvolvimento do software sempre existirá um ciclo de vida existente em todas as metodologias.
- Ciclo de vida pode ser definido como uma “estrutura contendo processos, atividades e tarefas envolvidas no desenvolvimento, operação e manutenção de um produto de software, abrangendo a vida do sistema desde a definição de seus requisitos até o término de seu uso”.



Prof. Anderson Augusto Bosing

Atividade

Em geral, um ciclo de vida de software envolve as etapas:

- ✓ **Planejamento**
- ✓ **Análise e Especificação de Requisitos**
- ✓ **Projeto**
- ✓ **Implementação**
- ✓ **Testes**
- ✓ **Entrega e Implantação**
- ✓ **Operação**
- ✓ **Manutenção**



Realizar pesquisa sobre cada uma das etapas acima:

Quais as principais atividades da etapa.

Quais são os profissionais de tecnologia da informação envolvidos nessa etapa.

Quais atividades são realizadas nesta etapa.

A etapa gera algum produto de trabalho como insumo para etapas posteriores.



Prof. Anderson Augusto Bosing

Engenharia de Software - Ciclo de vida

➤ Em geral, um ciclo de vida envolve as etapas:

- ✓ Planejamento
- ✓ Análise e Especificação de Requisitos
- ✓ Projeto
- ✓ Implementação
- ✓ Testes
- ✓ Entrega e Implantação
- ✓ Operação
- ✓ Manutenção



Prof. Anderson Augusto Bosing

Ciclo de vida - Planejamento

- Fornece uma estrutura que possibilita ao gerente fazer estimativas iniciais de recursos, custos e prazos;
- O escopo do software é estabelecido;
- Um plano de projeto deve ser elaborado configurando o processo a ser utilizado;
- Esta atividade faz parte da **gerência de projeto**.



Prof. Anderson Augusto Bosing

Ciclo de vida - Análise e Especificação de Requisitos

- O escopo do software é refinado;
- Nessa fase, a interação entre quem desenvolverá e o cliente é muito grande, contudo não se discute como será feito o software, mas sim o que ele deve fazer.
- Devem ser analisados o domínio do problema e o domínio da solução.
- São usadas as histórias de usuário (User Stories), que são pequenas histórias escritas para auxiliar na definição do requisito entre quem desenvolve e o cliente.
- Todo esse material é compilado e gera um relatório que servirá para os tomadores de decisão definirem, ou não, a continuidade e o desenvolvimento do software.



Prof. Anderson Augusto Bosing

Ciclo de vida – Projeto

- Utiliza a fase anterior como insumo.
- Essa fase é voltada ao programador do software.
- Definição de como seria a interface que o usuário opera, quais cores seriam utilizadas na interface, como seria o fluxo de funcionamento de cada botão existente na interface, em qual banco de dados ficariam os dados gerados.
- O projeto, diferente da fase anterior, define como as coisas acontecerão.



Prof. Anderson Augusto Bosing

Ciclo de vida – Implementação

- Essa fase deve traduzir o projeto em um software, utilizando ferramentas e linguagens adequadas.
- Dadas as inúmeras metodologias de desenvolvimento, cada programador pode desenvolver o código do sistema de uma forma diferente.



Prof. Anderson Augusto Bosing

Ciclo de vida – Testes

- **Teste de unidade:** cada componente é testado individualmente, sem conexão com outros componentes.
 - ✓ Exemplo: testar apenas o método que faz a soma dos valores das vendas, dados dois números, ele deve retornar a soma dos dois.
- **Teste de módulo:** um módulo é um agrupamento de pequenos componentes, que tem uma função específica.
 - ✓ Exemplo: testar a geração do relatório de vendas de um produto.
- **Teste de subsistemas:** são testados módulos integrados que controlam as interfaces do sistema.
 - ✓ Exemplo: testar as interfaces do sistema de compra e venda.



Prof. Anderson Augusto Bosing

Ciclo de vida – Testes

- **Teste de sistema:** testa a integração dos subsistemas que formam o sistema principal.
 - ✓ Exemplo: realizar login, operações e geração de relatórios em um sistema.
- **Teste de aceitação:** testes realizados com dados reais fornecidos pelos clientes. Último teste antes de colocar o sistema em operação.
 - ✓ Exemplo: quando há muitos usuários na base e o login não acontece de forma instantânea.



Prof. Anderson Augusto Bosing

Ciclo de vida – Entrega e Implantação

- O software deve ser instalado em ambiente produção.
- Envolve:
 - ✓ Treinamento de usuários;
 - ✓ Configuração do ambiente de produção;
 - ✓ Conversão bases de dados (se necessário).



Prof. Anderson Augusto Bosing

Ciclo de vida – Operação

Após os testes, entrega e implantação, o software passa a ser utilizado de fato em um ambiente de produção.



Prof. Anderson Augusto Bosing

Ciclo de vida – Manutenção

- **Manutenção corretiva:**
 - ✓ Correção dos erros encontrados pelo cliente e que não foram detectados nas fases de testes anteriores.
- **Manutenção adaptativa:**
 - ✓ Adaptação do software relacionado as mudanças do ambiente externo.
- **Manutenção evolutiva:**
 - ✓ Mudanças não previstas nos requisitos originais, visando a melhorias de desempenho e novas funcionalidades.
- **Manutenção preventiva:**
 - ✓ A iniciativa parte da equipe de desenvolvedores, visando evitar futuros problemas e melhorias em futuras manutenções.



Prof. Anderson Augusto Bosing

Engenharia de Software

**Aula: Metodologias
Tradicionais de
Desenvolvimento de
Software.**



Prof. Anderson Augusto Bosing

O desenvolvimento de um software é uma tarefa simples?



O desenvolvimento de um software é uma tarefa simples?

O desenvolvimento de software é uma atividade complexa. Essa complexidade corresponde ao desenvolvimento dos seus diversos componentes: software, hardware, procedimentos etc.

Isso se reflete no alto número de projetos de software que não chegam ao fim, ou que extrapolam recursos de tempo e dinheiro alocados.



O desenvolvimento de um software é uma tarefa simples?

Chaos Report, um estudo clássico feito pelo Standish Group sobre projetos de desenvolvimento:

- Porcentagem de projetos que terminam dentro do prazo estimado: 10%.
- Porcentagem de projetos que são descontinuados antes de chegar ao fim: 25%.
- Porcentagem de projetos acima do custo esperado: 60%.
- Atraso médio nos projetos: um ano.



O desenvolvimento de um software é uma tarefa simples?

Lidar com a complexidade e minimizar os problemas envolvidos no desenvolvimento de software envolvem a definição de processos de desenvolvimento de software.

Um processo de desenvolvimento de software compreende todas as atividades necessárias para definir, desenvolver, testar e manter um produto de software.



Metodologias de Desenvolvimento de Software

- Conforme a necessidade de desenvolvimento de sistemas foi crescendo, as atividades realizadas pelos responsáveis no desenvolvimento se repetiam a cada novo software.
- A partir do estudo dessas atividades, foram surgindo metodologias e processos, visando padronizar e agilizar o tempo do projeto.



Prof. Anderson Augusto Bosing

Metodologias de Desenvolvimento de Software

- 1. Modelo em Cascata ou Sequencial Linear;**
- 2. Modelo de Prototipação;**
- 3. Modelo RAD;**
- 4. Modelos Evolucionários:**
 - 4.1. Incremental;**
 - 4.2. Espiral;**
 - 4.3. Montagem de componente;**
 - 4.4. Desenvolvimento concorrente.**
 - 4.5. Modelo de Métodos Formais;**
 - 4.6. Técnicas de 4GT.**



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

- Baseado em projetos de engenharia clássicos ou ciclo da engenharia convencional, foi consolidado em 1970 por Royce;
- Organiza o processo em uma sequência linear de fases.
- É utilizado quando os requisitos são muito bem definidos.
- O modelo de ciclo de vida em cascata foi o primeiro modelo a ser conhecido em engenharia de software e está na base de muitos ciclos de vida utilizados hoje em dia. Este consiste basicamente num modelo linear em que cada passo deve ser completado antes que o próximo passo possa ser iniciado.
- O modelo em cascata é um exemplo de um processo dirigido a planos, isto é, as atividades devem ser programadas e planejadas antes de serem iniciadas.



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

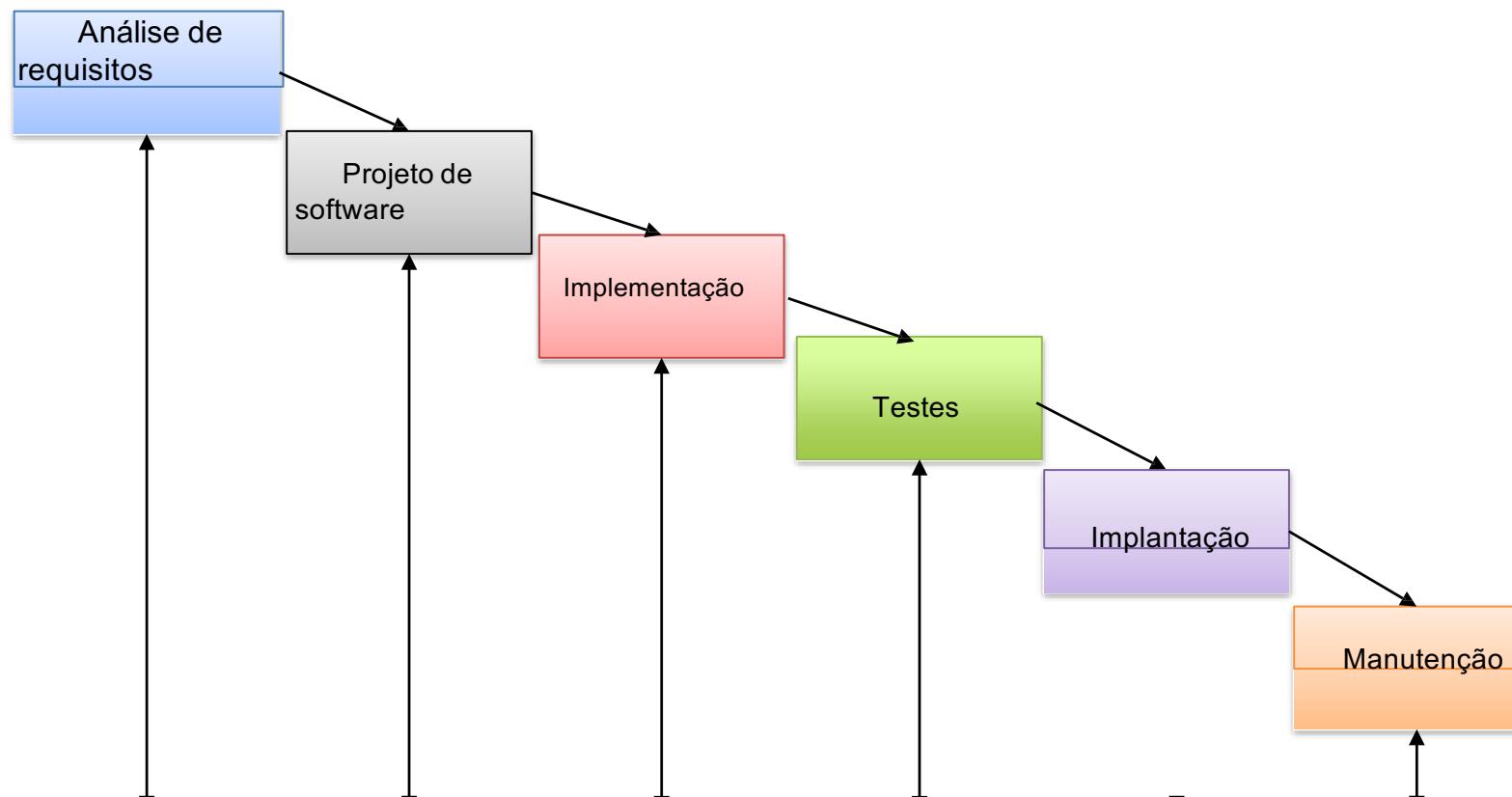
> Características:

- ✓ Ajuda os desenvolvedores a descrever o que precisam fazer (útil);**
- ✓ Ajuda a explicar o processo de desenvolvimento aos clientes (simples e fácil);**
- ✓ Os produtos intermediários são finalizados para começar próximo estágio e servem de insumo para o seu desenvolvimento;**
- ✓ Seu enfoque está nos documentos e artefatos (requisitos, projetos, códigos).**



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear



Representação clássica do modelo em Cascata - Pressman



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

- As funções, as restrições e os objetivos do sistema são estabelecidos por meio da consulta aos usuários do sistema.
- Deve-se compreender o domínio da informação, a função, desempenho e interfaces exigidos.
- Os requisitos (para o sistema e para o software) são documentados e revistos com o Cliente.

Analise de Requisitos



Modelo em Cascata ou Sequencial Linear

- Realiza todo o planejamento.
- São realizados os testes e as estimativas do desenvolvimento e criados os cronogramas para posterior acompanhamento do desenvolvimento do software.
- Define a arquitetura do sistema geral. É a tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie.

Projeto de Software



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

- Escreve os códigos, as interfaces com os clientes, a arquitetura do software e a estrutura de dados, para atender a todas as regras de negócio levantadas na primeira fase.
- Recomenda-se realizar testes unitários nos módulos durante essa fase.

Implementação



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

- O teste se concentra nos aspectos lógicos internos do software, garantindo que todas as instruções tenham sido testadas;
- Se concentra também nos aspectos funcionais externos, para descobrir erros e garantir que a entrada definida produza resultados que concordem com os esperados.

Testes



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

» O sistema é instalado e colocado em operação.

Implantação

» Normalmente, é a fase mais longa do processo, pois tratará do suporte ao cliente, resolvendo os problemas quando o cliente der o feedback e implementando novos requisitos conforme necessidade.

Manutenção



Prof. Anderson Augusto Bosing

Modelo em Cascata ou Sequencial Linear

- Apesar do modelo ser encadeado, passando de fase apenas com o término da fase anterior, na prática, acaba acontecendo uma sobreposição de algumas fases, ou seja, uma fase serve de insumo para a próxima fase e pode descobrir um problema na fase anterior, levando a fase anterior a ser modificada.
- Essa abordagem apresenta a vantagem de produzir uma documentação muito consistente em cada fase, porém é recomendada apenas quando os requisitos são pré-estabelecidos, bem conhecidos e com pouca probabilidade de mudança com o passar do tempo.



Prof. Anderson Augusto Bosing

Modelo de prototipação

- Laudon (2014) afirma que prototipagem “consiste em montar um sistema experimental rapidamente e sem muitos gastos para submetê-lo à avaliação de usuários finais. O protótipo é uma versão funcional de um sistema de informação, ou de parte dele, mas deve ser considerado apenas um modelo preliminar.”
- De acordo com Pressman (2011), o protótipo serve como um mecanismo para identificação dos requisitos do software, pois, se ele é elaborado, o desenvolvedor tenta usar partes de programas existentes ou aplicar ferramentas que possibilitem programas executáveis serem gerados rapidamente.



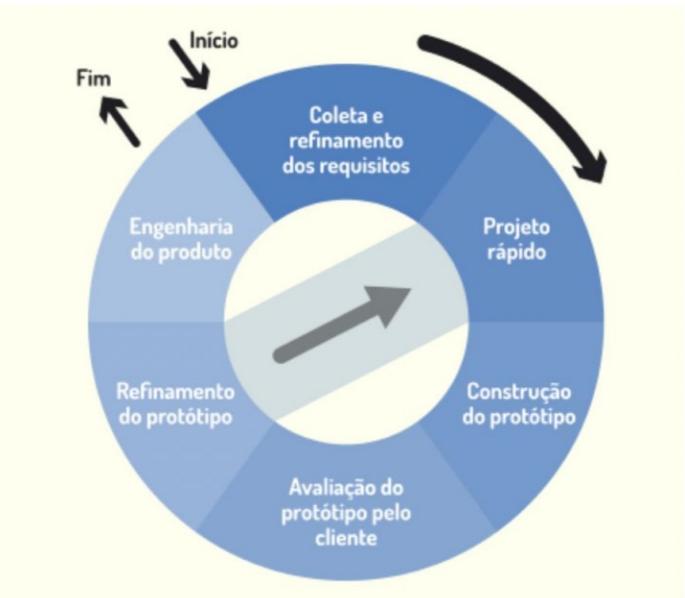
Prof. Anderson Augusto Bosing

Modelo de prototipação

Coleta e refinamento dos requisitos: ocorre a identificação dos requisitos junto ao cliente.

Nessa fase, o engenheiro mantém contato direto com o cliente apenas para captar as necessidades básicas de informação.

Projeto rápido: fase que define como será a iteração da prototipação e na qual acontece a criação de um projeto rápido, que contempla a parte do software que estará visível ao cliente. Abordagens de entrada e formatos de saída.



Modelo de prototipação

Construção Protótipo: implementação rápida do projeto.

Avaliação do protótipo pelo cliente: fase de utilização do protótipo desenvolvido na fase anterior pelo usuário final, a fim de validar o software, sugerir mudanças e aperfeiçoamentos.

Refinamento do Protótipo: cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido. Ocorre neste ponto um processo de interação que pode conduzir à primeira atividade até que as necessidades do cliente sejam satisfeitas e o desenvolvedor compreenda o que precisa ser feito.



Modelo de prototipação

Engenharia de Produto:
Identificados os Requisitos, o protótipo deve ser descartado e a versão de produção deve ser construída considerando os critérios de qualidade.



Prof. Anderson Augusto Bosing

Modelo de prototipação

➤ Vantagens:

- ✓ Todo o requisitos de sistema não tem que ser completamente determinado antecipadamente.
- ✓ A satisfação e interação do cliente no desenvolvimento do protótipo;
- ✓ Testes através do protótipo para atingir o objetivo final proposto.

➤ Desvantagens:

- ✓ O processo de prototipação pode dar ao usuário final a impressão que praticamente qualquer sugestão pode ser implementada, não importa qual estágio do processo de desenvolvimento se está.
- ✓ Além disso, para o usuário final não está claro o porquê da demora para entregar a aplicação final depois que uma versão demo do sistema foi exibida.
- ✓ O desenvolvedor frequentemente faz uma implementação comprometida (utilizando o que está disponível) com o objetivo de produzir rapidamente um protótipo. Depois de um tempo ele se familiariza com essas escolhas, e esquece que elas não são apropriadas para o produto final.



Prof. Anderson Augusto Bosing

Desenvolvimento Rápido de Aplicação - RAD

- É o modelo sequencial linear mas que enfatiza um desenvolvimento extremamente rápido.
- A “alta velocidade” é conseguida através de uma abordagem de construção baseada em componentes.
- O desenvolvimento rápido de aplicação é adequado para construção de aplicações em curto espaço de tempo, utilizando o método incremental da prototipação, com o ciclo extremamente curto de desenvolvimento, que só é possível quando temos grande entendimento dos requisitos e o projeto é curto e simples.
- É um modelo de processo de desenvolvimento de software iterativo e incremental que enfatiza um ciclo de desenvolvimento extremamente curto (entre 60 e 90 dias).



Prof. Anderson Augusto Bosing

Desenvolvimento Rápido de Aplicação - RAD

- A aplicação deve ser modularizada de forma que cada função deva ser completada em pelo menos 3 meses.
- Cada modulo pode ser alocado a uma equipe distinta e depois serem integradas para formar um todo.
- Criação e reutilização de componentes.
- No RAD, há o sequenciamento existente no modelo cascata e a ideia de apresentações parciais ao cliente para validação e continuação do desenvolvimento, vinda do modelo de prototipação, porém, nesse modelo, o que é apresentado ao cliente é, de fato, parte do software, e não apenas um protótipo.



Prof. Anderson Augusto Bosing

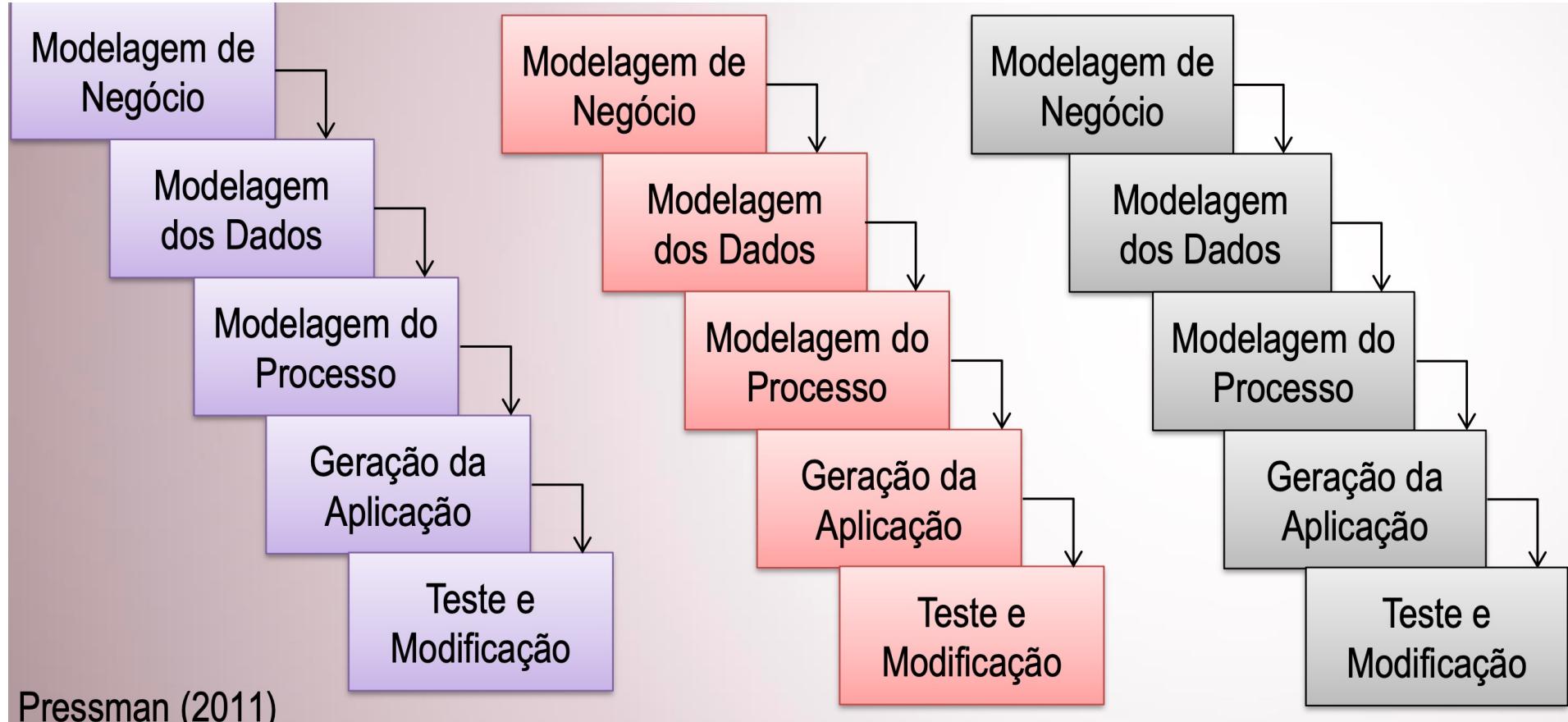
Desenvolvimento Rápido de Aplicação - RAD

- A aplicação deve ser modularizada de forma que cada função deva ser completada em pelo menos 3 meses.
- Cada modulo pode ser alocado a uma equipe distinta e depois serem integradas para formar um todo.
- Criação e reutilização de componentes.
- No RAD, há o sequenciamento existente no modelo cascata e a ideia de apresentações parciais ao cliente para validação e continuação do desenvolvimento, vinda do modelo de prototipação, porém, nesse modelo, o que é apresentado ao cliente é, de fato, parte do software, e não apenas um protótipo.



Prof. Anderson Augusto Bosing

Desenvolvimento Rápido de Aplicação - RAD



Modelo Incremental

- Segundo Pressman (2011) “modelos evolucionários são iterativos. Apresentam características que possibilitam desenvolver versões cada vez mais completas do software”.
- Dessa forma, o primeiro modelo evolucionário que surgiu é o modelo incremental de desenvolvimento de software.
- Para Sommerville (2014) o sistema incremental tem o objetivo de reduzir o retrabalho custoso do modelo cascata, possibilitando ao cliente postergar decisões e requisitos conforme necessidade, mas mantendo o poder de gerenciamento que o modelo oferece.



Prof. Anderson Augusto Bosing

Modelo Incremental

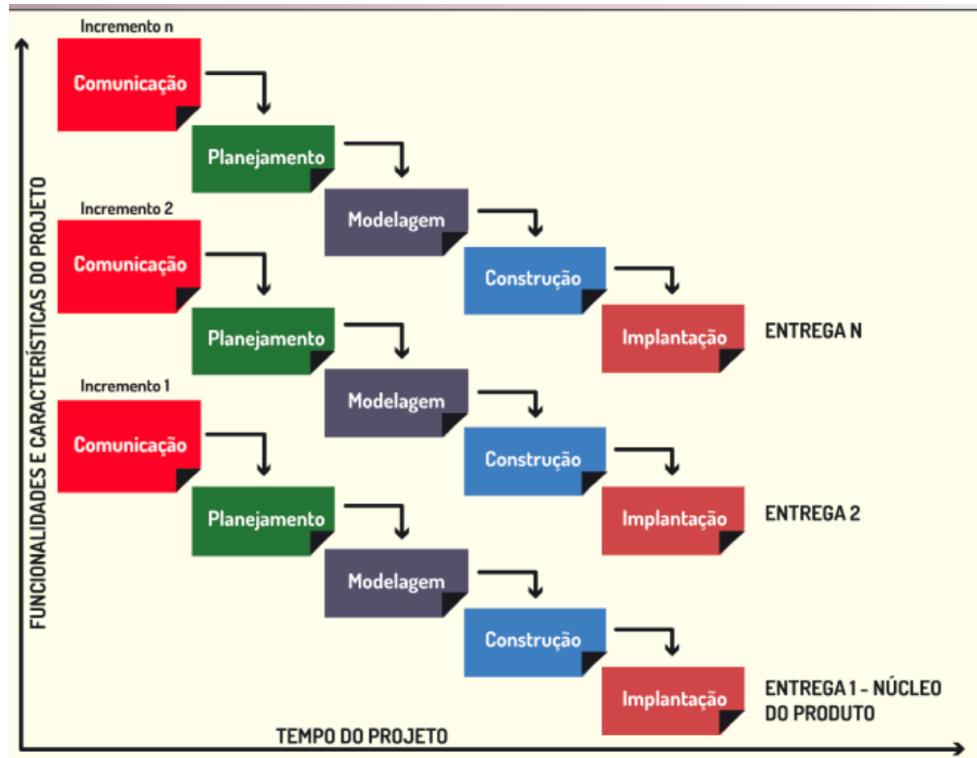
- Essa metodologia divide a fase de desenvolvimento do software em ciclos completos, passando por todas as fases existentes no modelo clássico, levantamento e análise de requisitos, projeto, implementação e testes.

- O modelo trabalha com 5 (cinco) fases distintas: comunicação, planejamento, modelagem, construção e implantação.



Prof. Anderson Augusto Bosing

Modelo Incremental



1. **Comunicação:** trata das informações do negócio, como são geradas, processadas e quem utilizará. Define os requisitos.
2. **Planejamento:** define os objetos, suas características e como se relacionarão entre si.
3. **Modelagem:** desenha o processo, visando atender à função do negócio, e define os procedimentos necessários para a manipulação dos objetos de dados definidos na etapa anterior.
4. **Construção:** ocorre a geração da aplicação, normalmente por meio de reutilização de componentes que serão integrados posteriormente.
5. **Implantação:** acontece alguns pequenos testes e a entrega de parte do produto.

Modelo espiral

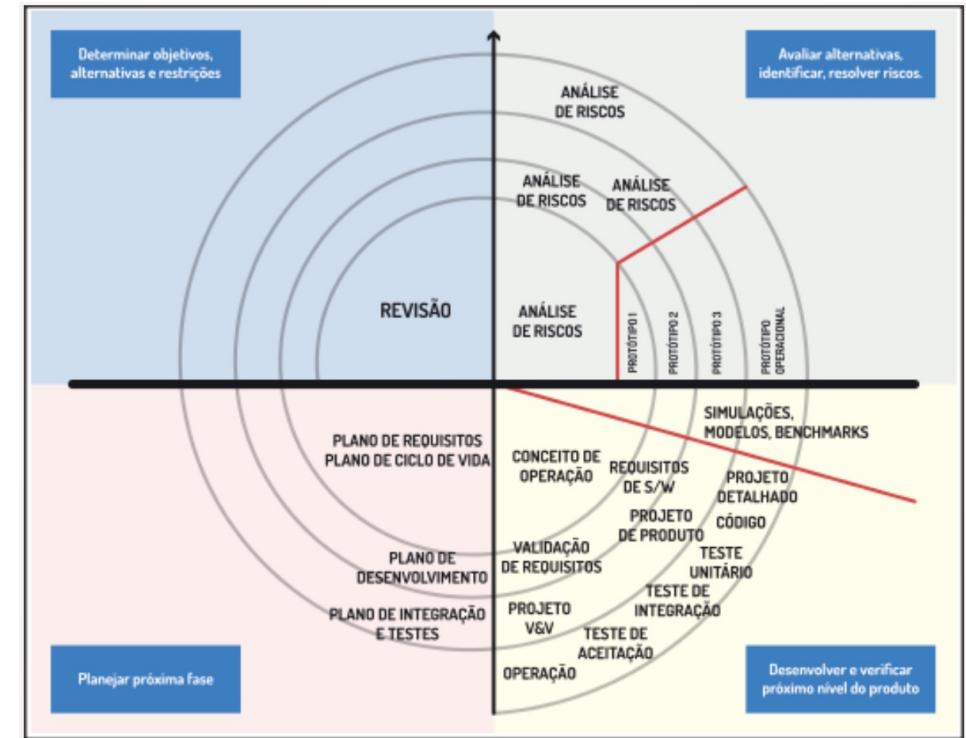
- Pressman (2011) define que esse é um modelo de processo de software evolucionário que acopla a natureza **iterativa da prototipação** com os **aspectos sistemáticos e controlados do modelo cascata**. Fornece potencial para o rápido desenvolvimento de versões cada vez mais completas de software.
- Cada loop na espiral representa uma fase do processo de software.
- As atividades acontecem em forma de espiral, no sentido horário, começando do centro, no ponto indicado como início. O software será construído em versões evolutivas, sendo que, em cada iteração da espiral, temos a entrega de um protótipo ou de uma versão mais completa do sistema.



Prof. Anderson Augusto Bosing

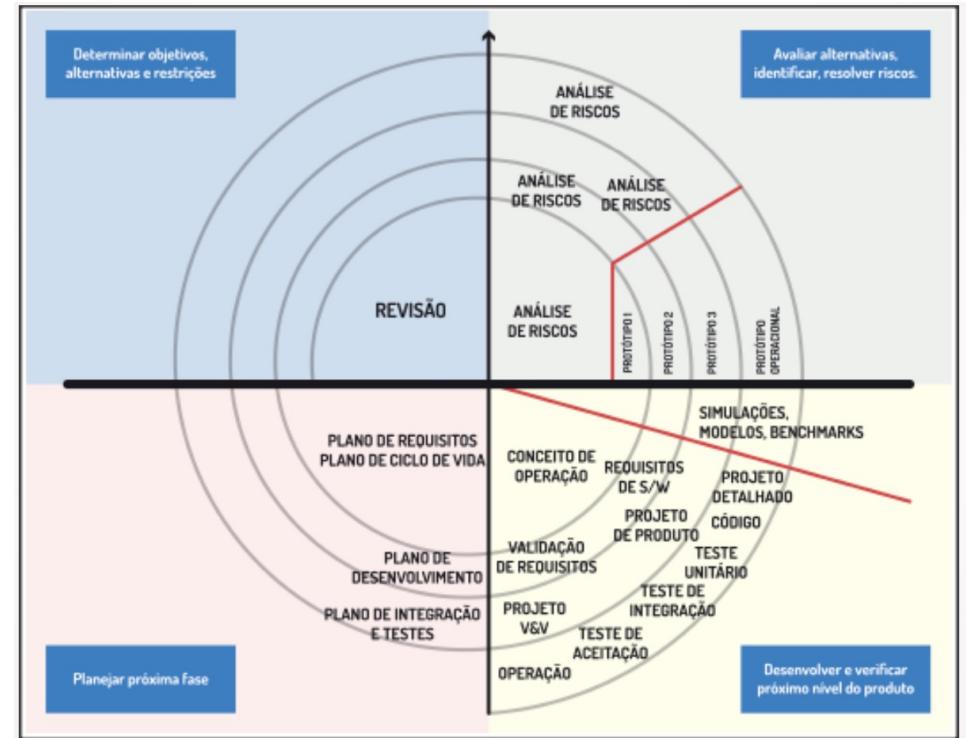
Modelo espiral

- 1. Planejamento e Requisitos** - nessa fase, é feito o levantamento de requisitos junto ao cliente, estabelecendo as restrições e as diretrizes do software, bem como os riscos. É criado um plano de gerenciamento, com cronogramas e estimativas de custos.
- 2. Análise de Riscos** - nesse ponto, é realizado uma análise para cada um dos riscos levantados e estudado maneiras de reduzi-los.
- 3. Desenvolvimento e Validação** – o protótipo é desenvolvido e validado.
- 4. Entrega e Avaliação de Continuidade** - nessa fase, ocorre uma entrega para o cliente e uma avaliação, para decidir se o projeto continuará para o próximo loop da espiral. Caso seja decidida a continuidade, o custo, o cronograma e o planejamento da quantidade de iterações planejadas anteriormente serão refinados de acordo com os feedbacks recebidos do usuário.



Modelo espiral

- Sommerville (2014) “A importante distinção entre o modelo espiral e outros modelos de processo de software é a explícita consideração dos **riscos** no modelo em espiral”.
- Segundo o **PMBOK**: riscos são efeitos negativos nos objetivos do projeto, como escopo, qualidade, custo e prazo.
- Exemplo:
Durante a primeira espiral, é utilizado prototipação, para tirar dúvidas e realizar alinhamento com o cliente, visando à redução dos riscos, e, nos próximos loops da espiral, é utilizado o modelo cascata.



Montagem de Componente

- Faz reuso de software visando o ganho em agilidade e confiabilidade, utilizando tecnologias de orientação a objetos.
- Sendo que agilidade desse modelo se deve ao fato do **reuso dos componentes**, obtido por meio de adaptações e de combinações.
- *Pressman (2011)* define que “a engenharia de software baseada em componentes identifica, constrói, cataloga e dissemina um conjunto de componentes de software em determinado domínio de aplicação. Esses componentes são então qualificados, adaptados e integrados para uso em um novo sistema..”



Prof. Anderson Augusto Bosing

Montagem de Componente

➤ Exemplo:

Um software para controle de loja de materiais de construção.
Para ele, é criado o componente de login e o componente de
cadastro de usuários.

➤ Um software para controle de uma loja de moveis.

Pode ser reutilizado os dois componentes criados para o software de controle da loja de materiais de
construção para compor o software da loja de moveis, visto que já estão em uso, funcionando e testados.



Prof. Anderson Augusto Bosing

Técnicas de 4a Geração

- As técnicas de quarta geração englobam um **conjunto de ferramentas de softwares** que visa permitir ao desenvolvedor especificar as características do software em alto nível.
- A técnica de quarta geração tem o objetivo de que os requisitos descritos pelo cliente sejam capazes de ser introduzidos em uma ferramenta que automaticamente geraria a estratégia do projeto, as codificações e realizaria os testes.



Prof. Anderson Augusto Bosing