

FIA P GRADUAÇÃO

DISCIPLINA:
ENGENHARIA E DESIGN DE SOFTWARE

AULA:
2-CICLOS DE VIDA APLICADOS AOS PROJETOS DE SOFTWARE

PROFESSOR:
RENATO JARDIM PARDUCCI

PROFRENATO.PARDUCCI@FIAP.COM.BR

AGENDA DA AULA

CICLOS DE VIDA DE SOFTWARE:

- ✓ Modelos e aplicação

A ATIVIDADE DE DESENHO DA SOLUÇÃO:

- ✓ Dentro do ciclo de vida de software
- ✓ Dentro do processo de software
- ✓ Dentro da arquitetura empresarial

**CICLO DE VIDA DA
PRODUÇÃO DE SOFTWARE**

Agora que **já é conhecida** a característica do **ciclo de vida** de um software **APÓS** o início da sua utilização, vamos conhecer como é o ciclo de vida **ANTES** do software ser lançado!

APRENDENDO NA PRÁTICA



Você precisa agora, começar a planejar como vai desenvolver o seu software – precisa definir o chamado Ciclo de Vida de Desenvolvimento e escolher um processo de produção adequado.

O ciclo de vida de desenvolvimento vai do momento da concepção de ideias até a entrega do produto para uso e entrada no período de manutenção.

O ciclo de vida que você escolher vai determinar fases do projeto, formas de conduzir as atividades produtivas e ritmo de entregas de produtos.

Os Ciclos de Vida retratam uma forma de pensar e analisar a evolução das coisas ao longo do tempo da sua utilização.

Ciclos de Vida são formas de pensar!
São paradigmas!

Esse modelos...

Definem etapas ou fases a serem cumpridas para que o desenvolvimento de software ocorra de forma disciplinada, organizada, progressiva, compreensível, gerenciável e que alcance as expectativas de seus idealizadores.







Toda produção de software envolve as seguintes etapas que precisam ser administradas:

- ★ **Determinação de **escopo**:** especificação da **necessidade a ser atendida** pelo sistema.
- ★ **Desenho **lógico**:** explicação **descritiva e/ou ilustrada das funções** que o software deve ter e dados que deve guardar;
- ★ **Desenho **físico**:** explicação **descritiva e/ou ilustrada** para apresentar as escolhas de **plataforma** de desenvolvimento, arquitetura dos ambientes de hardware e software onde o sistema executará, padrões de **interface com o usuário**, padrões técnicos de construção e integração de **componentes** do software, **protocolos** e arquitetura da comunicação de dados local e remota, regras de **segurança** que serão aplicadas sobre redes de computadores e no software.
- ★ **Implementação:** produção do software em uma **linguagem de programação**, respeitando as diretrizes de desenho lógico e físico.

Na Engenharia de Software, desenvolveu-se diversos modelos conceituais de Ciclo de Vida para administrar as etapas de produção do software e sua manutenção pós-liberação para uso:

- Modelo de processo **Cascata**
- Modelo de processo **Incremental**
- Modelo de **Prototipação**
- Modelo de processo **Espiral**

CICLOS DE VIDA DE SOFTWARE**MODELO CASCATA**

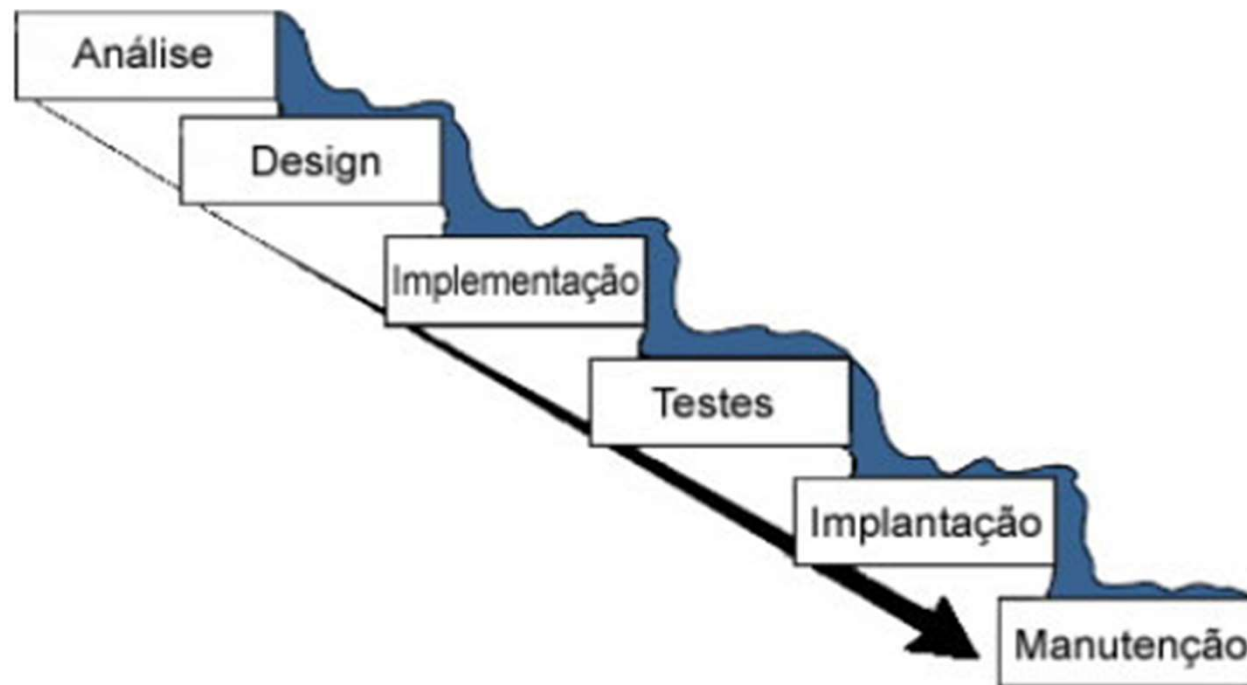
O projeto inteiro é tratado como uma frente de desenvolvimento única!

Seus passos são claros e sequenciais!

A evolução do projeto acontece como um todo: se estiver modelando o sistema, não existirá nada sendo construído. Se estiver construindo, nada poderá ser implantado no mesmo momento!

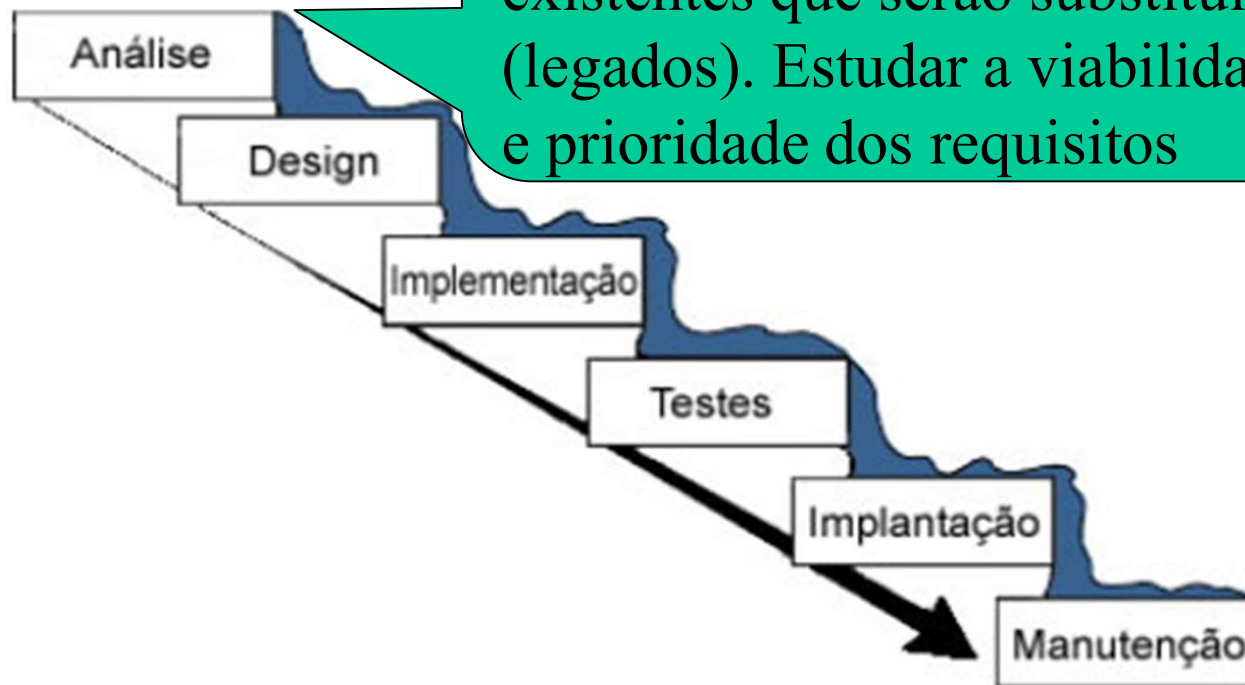
CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA



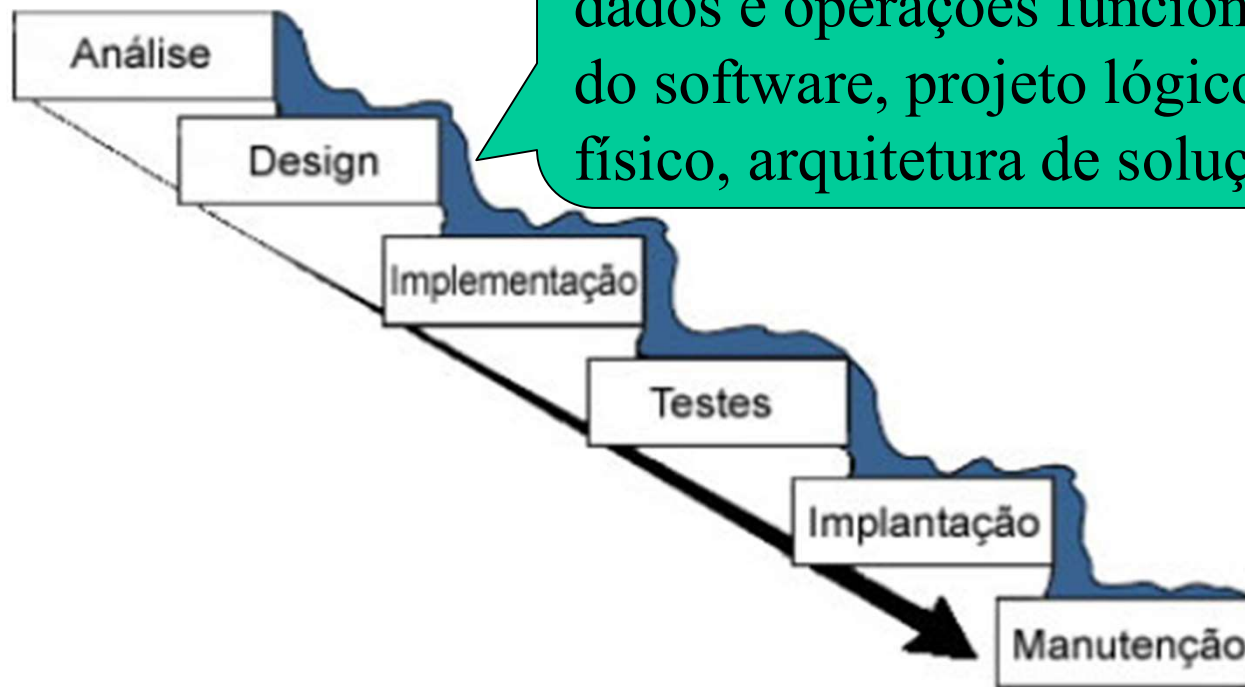
CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA



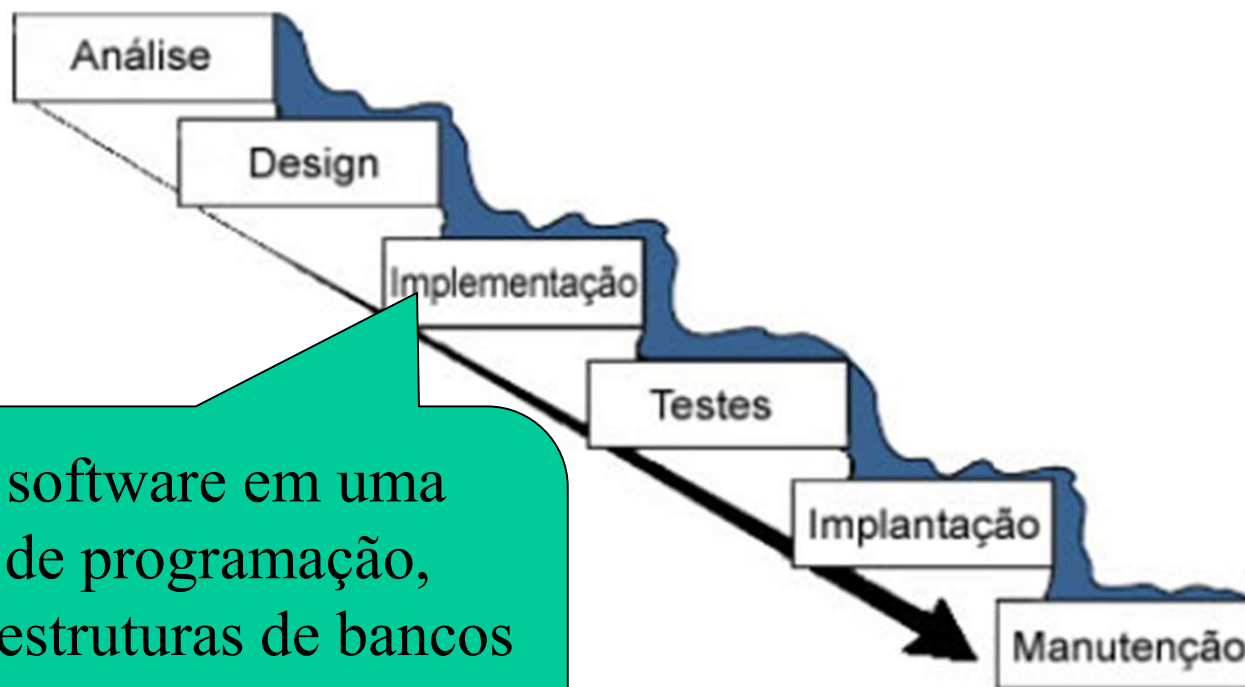
CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA



CICLOS DE VIDA DE SOFTWARE

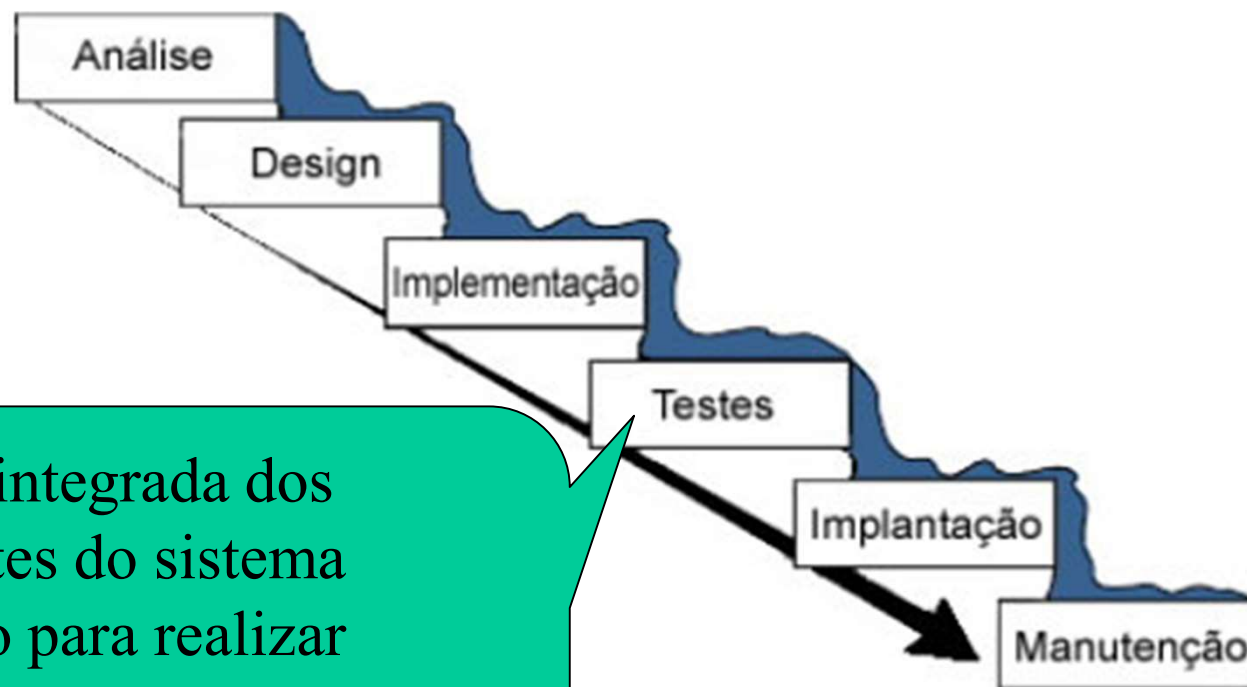
MODELO CASCATA



Escrever o software em uma linguagem de programação, criação de estruturas de bancos de dados, teste individual (unitário) dos programas. Criação de helps e manuais.

CICLOS DE VIDA DE SOFTWARE

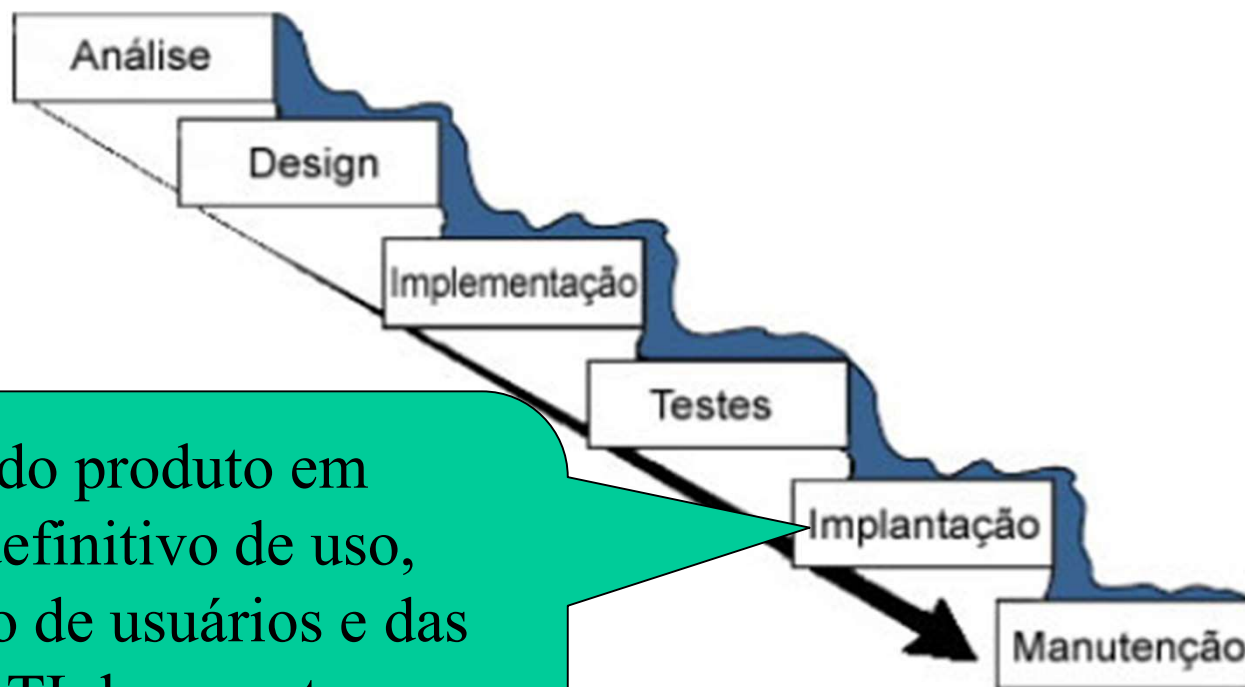
MODELO CASCATA



Avaliação integrada dos componentes do sistema interagindo para realizar funções completas de negócio, homologação com o usuário em ambiente controlado

CICLOS DE VIDA DE SOFTWARE

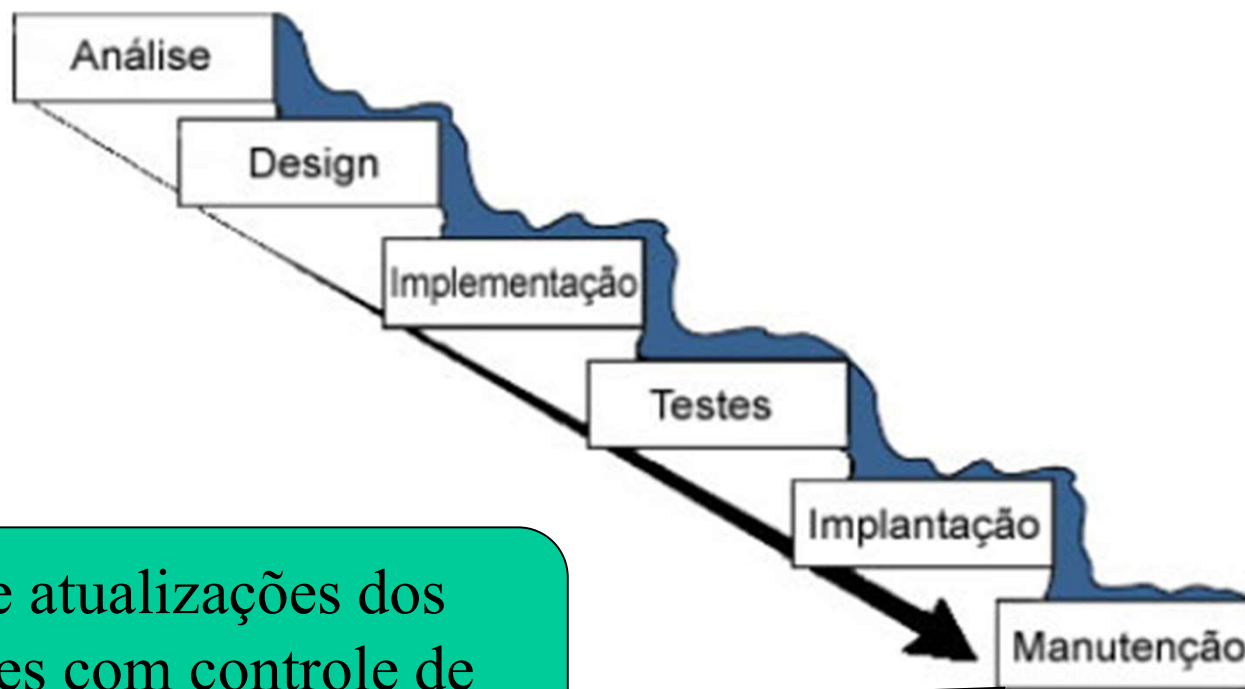
MODELO CASCATA



Instalação do produto em ambiente definitivo de uso, capacitação de usuários e das equipes de TI de suporte e operações. Configuração de acessos e segurança de dados.

CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA

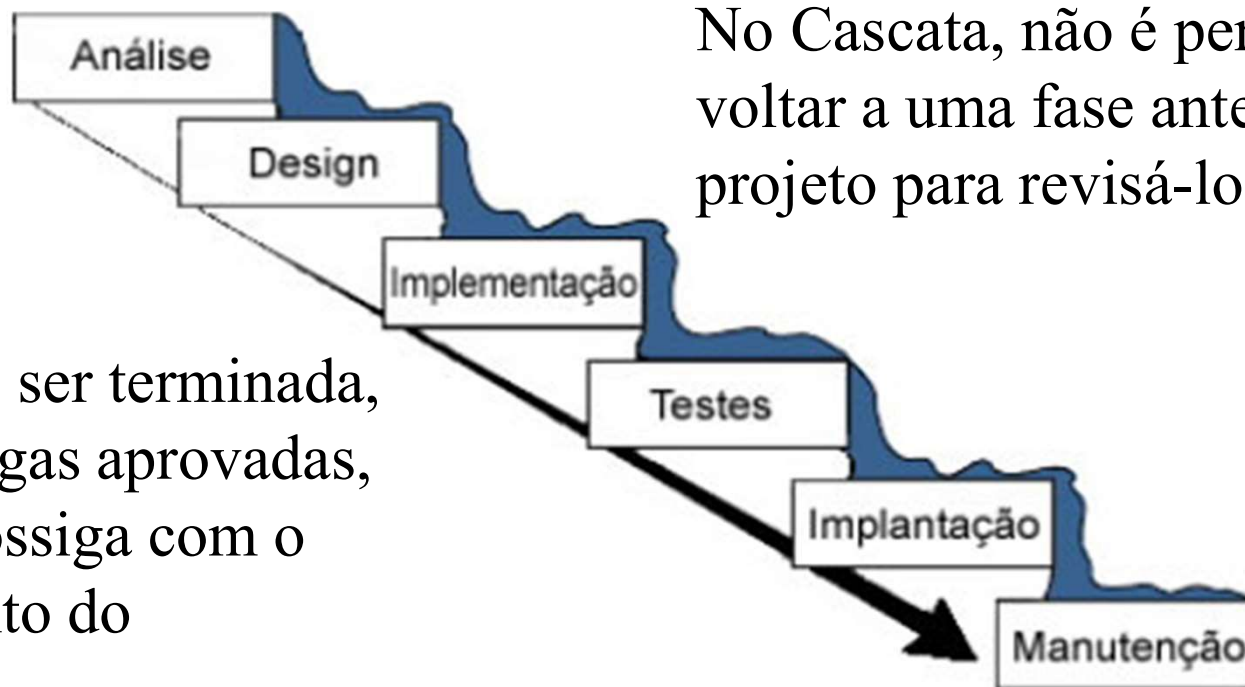


Correções e atualizações dos componentes com controle de versionamento

CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA

A água que desce não volta!
No Cascata, não é permitido voltar a uma fase anterior do projeto para revisá-lo!



Cada fase deve ser terminada, com suas entregas aprovadas, para que se prossiga com o desenvolvimento do software!

As entregas de uma fase subsidiam a seguinte!

ATIVIDADE PRÁTICA



Vamos estudar o cronograma de referência para projetos de software conduzidos no modelo Cascata, da área de apostilas!

Atividade	Duração	Início	Término
PROJETO DE SOFTWARE - Condução Clássica	185 days	Mon 13/02/17	Fri 27/10/17
Fase de Iniciação	10 days	Mon 13/02/17	Fri 24/02/17
Definição do Ciclo de Vida do Software (Modelo Cascata, Incremental, Prototipação ou Espiral)	5 days	Mon 13/02/17	Fri 17/02/17
Atribuição da equipe inicial responsável pelo projeto	1 day	Mon 20/02/17	Mon 20/02/17
Identificação dos demandantes do projeto (Stakeholders)	1 day	Tue 21/02/17	Tue 21/02/17
Concepção da demanda	1 day	Wed 22/02/17	Wed 22/02/17
Estimativa de esforço, tempo e custo com base em histórico	1 day	Thu 23/02/17	Thu 23/02/17
Documentação do termo/acordo de abertura de projeto (TAP)	1 day	Fri 24/02/17	Fri 24/02/17
Documento de Visão de escopo inicial, premissas e restrições de projeto, definido	0 days	Fri 24/02/17	Fri 24/02/17

Trata-se de um exemplo de um plano de projeto, baseado em um ciclo de vida Cascata.

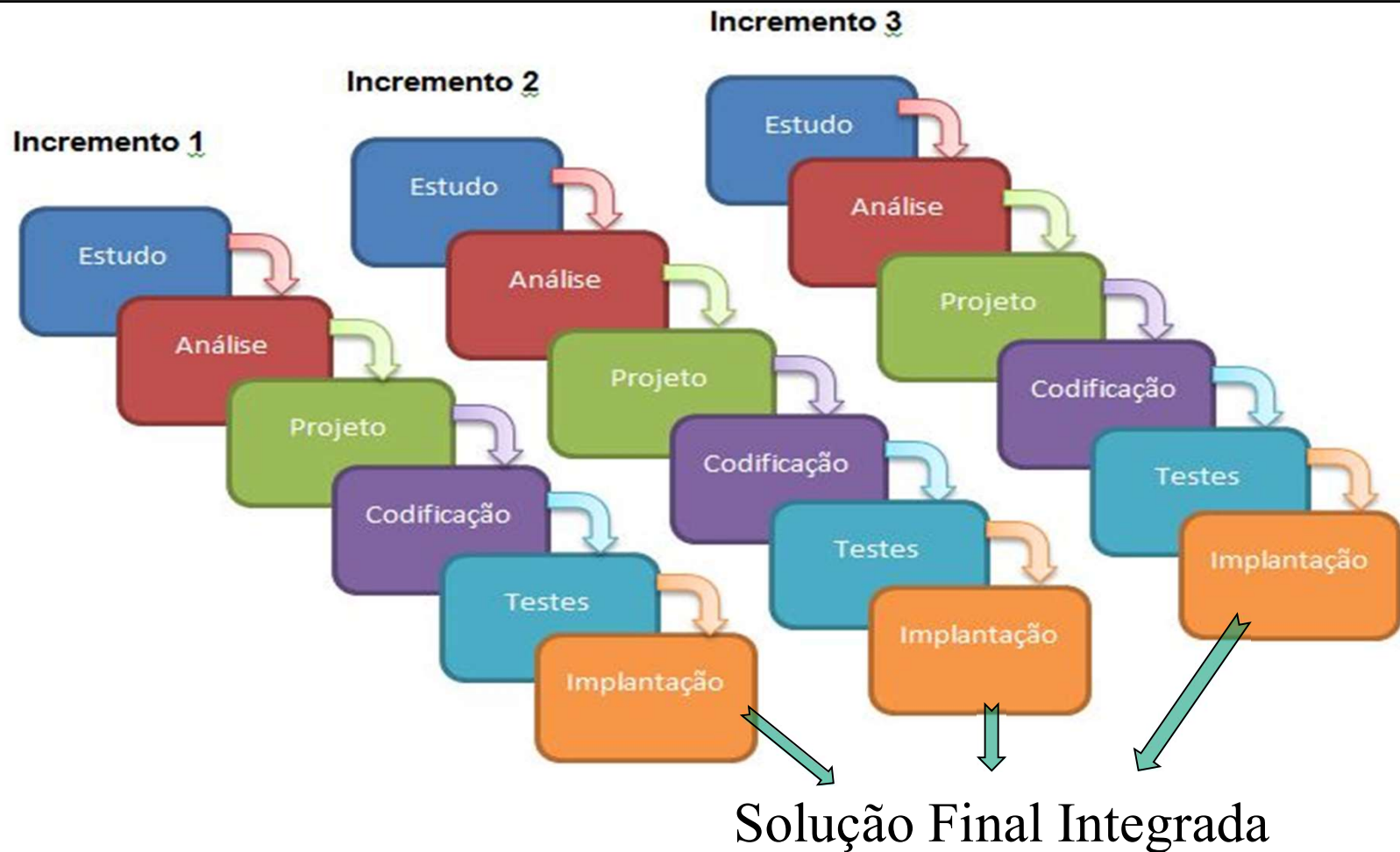
CICLOS DE VIDA DE SOFTWARE**MODELO INCREMENTAL**

O projeto é fatiado em módulos que podem ser produzidos com certa independência e em paralelo!

Os módulos devem ser integrados para compor o sistema/solução final para o cliente!

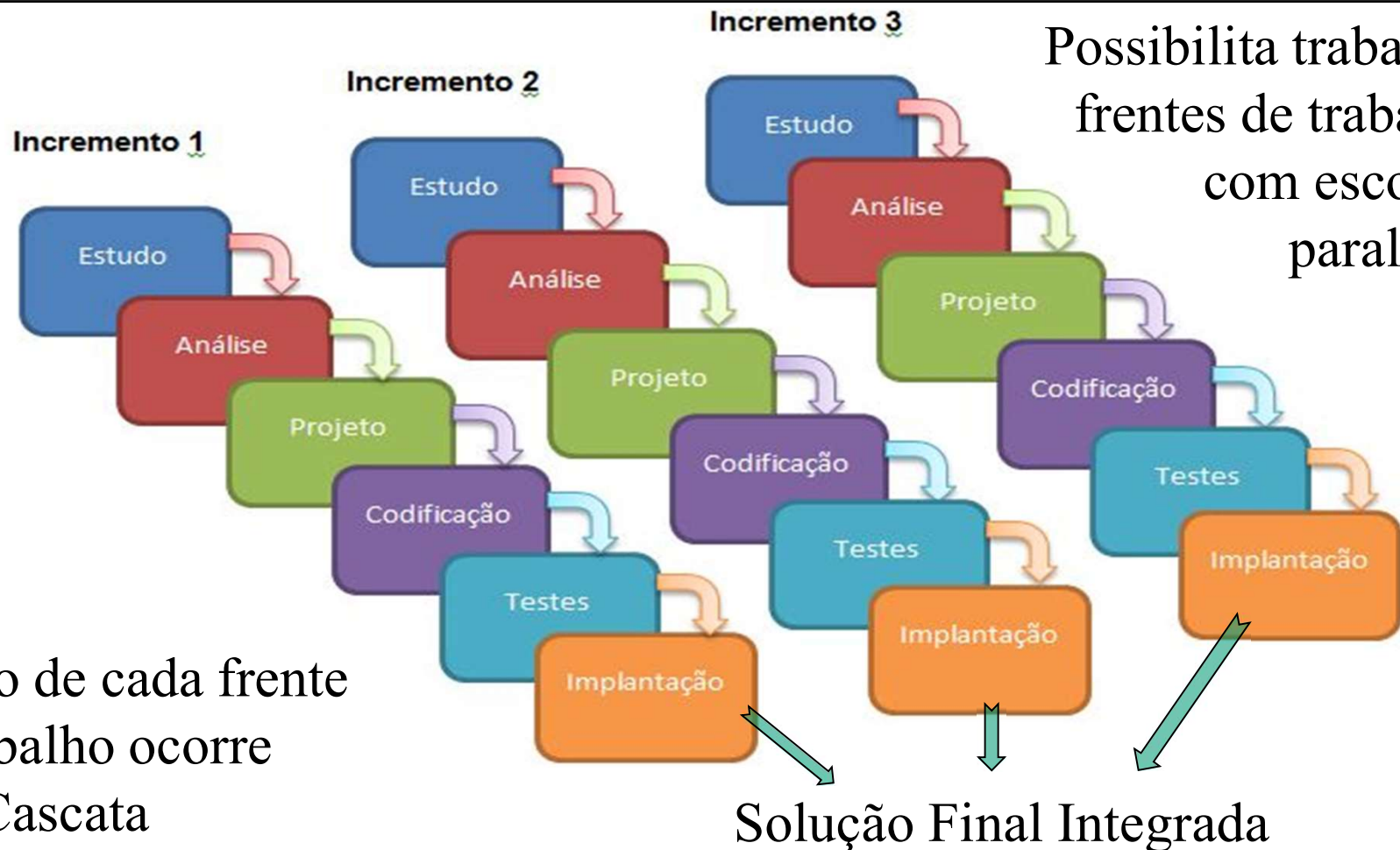
CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL



CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL



CICLOS DE VIDA DE SOFTWARE**MODELO PROTOTIPAÇÃO EVOLUCIONÁRIA**

O projeto evolui a partir de protótipos (mockup)!

Constrói-se o protótipo, valida-se o protótipo e depois, constrói-se a solução definitiva!

Um protótipo feito pode ser descartado, revisado ou aproveitado na construção definitiva, possibilitando revisão e adaptação evolutiva!

O desenvolvimento ocorre em ciclos e partes do produto de software podem ser desenvolvidas em separado e depois integradas. O cliente permanece em contato constante durante o projeto que evolui em seus requisitos e construção a cada ciclo de produção.

CICLOS DE VIDA DE SOFTWARE

MODELO PROTOTIPAÇÃO EVOLUCIONÁRIA



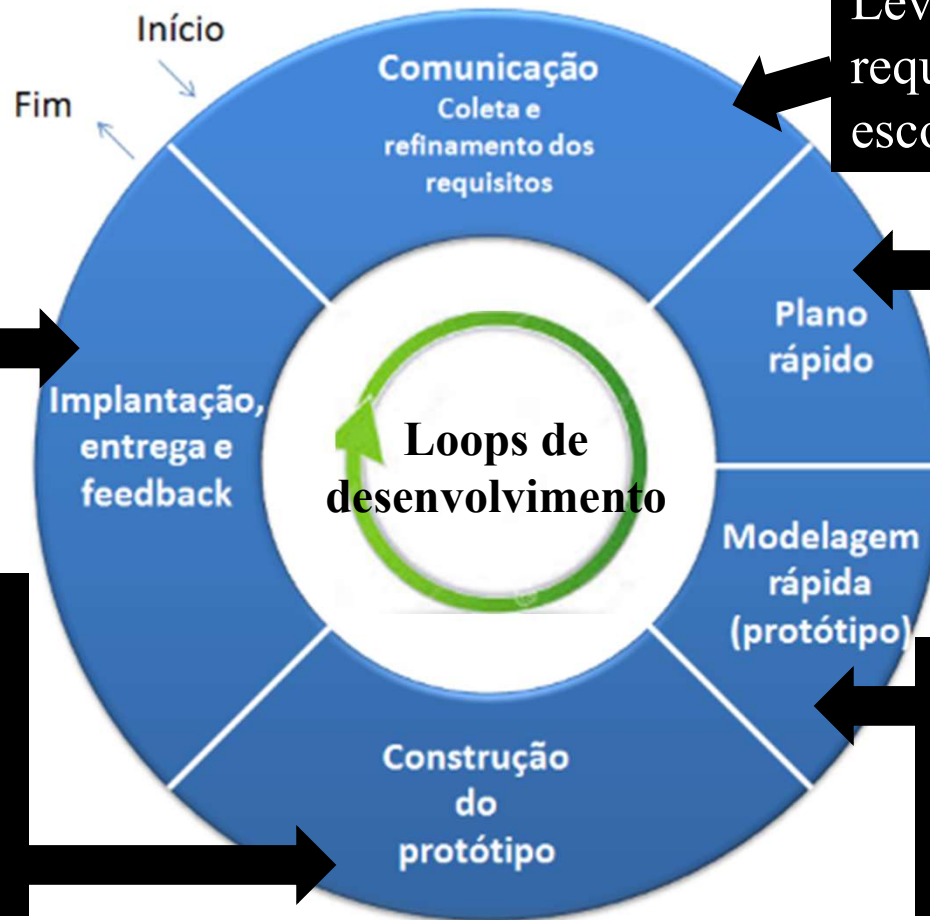
CICLOS DE VIDA DE SOFTWARE

MODELO PROTOTIPAÇÃO EVOLUCIONÁR

Os ciclos de desenvolvimento acontecem até que o produto esteja completo

Liberar o software p/uso, instalar o software, treinar usuários e técnicos de operação e suporte

Desenvolver código fonte, aplicar testes, liberar componente, integrar componentes



Levantar e documentar requisitos. Definir escopo do produto

Definir atividades de trabalho, distribuir tarefas, definir prazos

Elaborar prova de conceito (POC), e protótipo do produto (MOCUP)

CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL

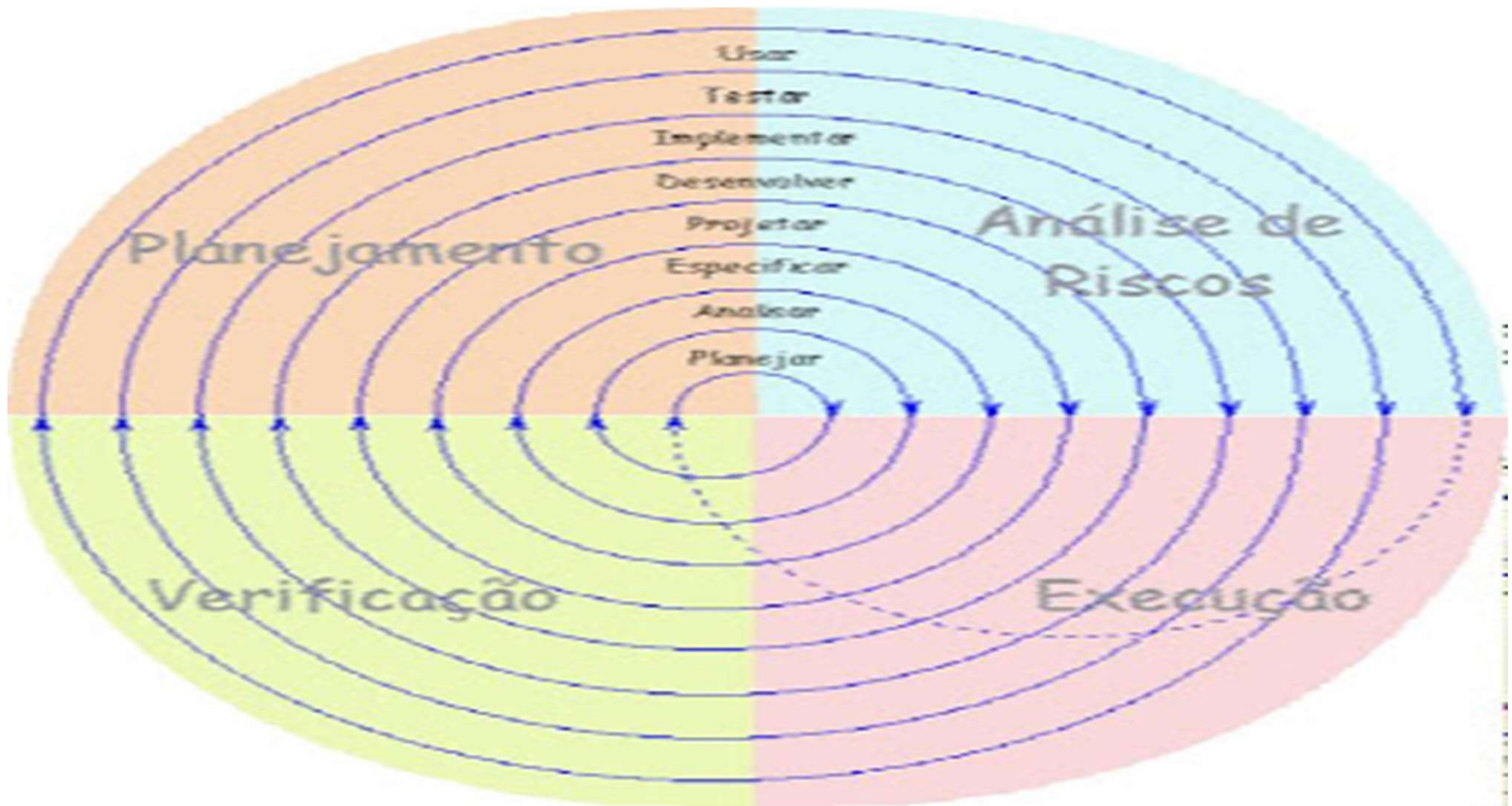
O projeto pode ser desenvolvido com flexibilidade de adaptação, onde o software pode ser repartido e ter módulos e até componentes individuais sendo evoluídos em ritmo distinto dos demais.

Contém os princípios:

- O projeto deve passar a todo tempo por Planejamento, Avaliação de Riscos observados mediante o plano, Execução do plano, Monitoração e Controle de resultados, de forma a garantir melhoria contínua no projeto;
- O desenvolvimento não é linear, ou seja, **é possível ir e voltar nas etapas do desenvolvimento** como modelagem, construção, teste sendo mais importante garantir a aderência do software aos requisitos do que o cumprimento de um plano traçado preliminarmente
- O projeto envolve negociação constante em busca do ganho mútuo entre os desenvolvedores e o cliente da solução

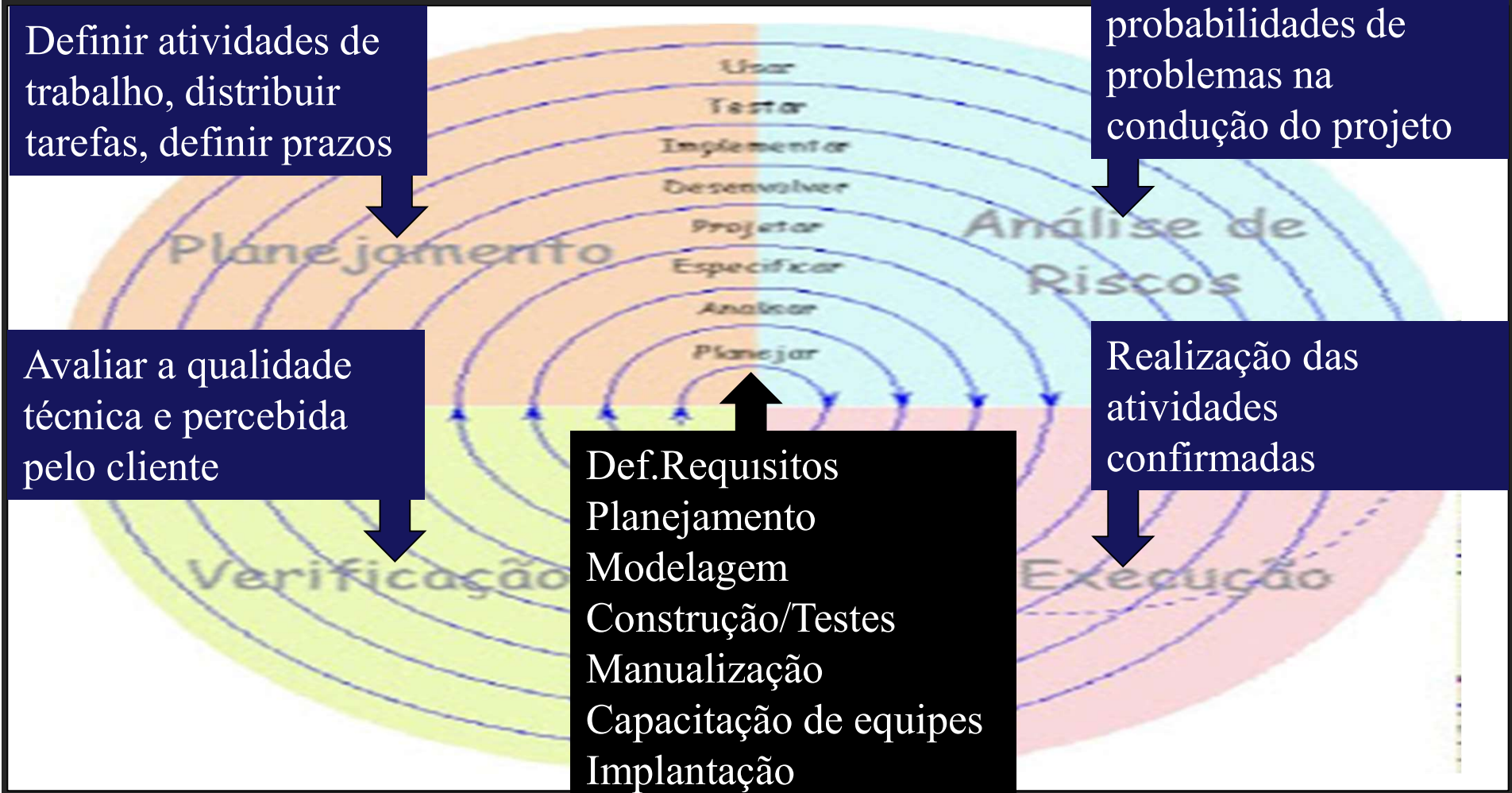
CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL



CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL



CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL

Partes do projeto (alguns módulos e componentes de software) podem estar **sendo construídos, enquanto outros estão sendo modelados e outros ainda estão tendo requisitos negociados**



ATIVIDADE PRÁTICA



No projeto que será desenvolvido para o estacionamento, o proprietário deseja que entreguemos o sistema o mais rapidamente possível.

Se o projeto for longo, ele prefere que entreguemos o sistema mesmo que parcialmente, o quanto antes.

Para o proprietário, controlar os pagamentos de forma digital é a primeira prioridade, depois o controle de entrada e saída de carros, em seguida o controle de ocupação de vagas e por fim (se ele ainda tiver dinheiro para investir), quer automatizar a leitura de placas e abertura de cancelas.

Qual ciclo de vida de projeto você propõe usar?

ATIVIDADE PRÁTICA



Decidimos que em nosso projeto para o estacionamento, vamos usar o modelo de ciclo de vida Espiral para podermos entregar rapidamente algumas partes do software para uso, de forma a cativar o cliente que está relutante quanto aos investimentos que fará.

Para empregar o ciclo Espiral, vamos adotar um processo Ágil de produção.

Algumas opções são o RAD, o DSDM, o XP, o SCRUM.

Optamos pelo SCRUM, que é o framework mais adotado internacionalmente em projetos ágeis de software.

CICLOS DE VIDA DE SOFTWARE

CICLO DE VIDA E O PROCESSO DE PRODUÇÃO DE SOFTWARE

Os **processos de produção de software** envolvem a **aplicação de técnicas** para gerar o produto final (o software) e **administrar as atividades** de trabalho da concepção até a entrega.

Existem processos **Clássicos ou Tradicionais** e processos Ágeis.

Os **Processos** aplicam um **Ciclo de Vida** como referência.

Os processos Clássicos alinham com perfeição com CVS (Ciclo de Vida de Software) Cascata ou Incremental; já os Ágeis combinam mais com abordagens Evolucionária e principalmente Espiral.

CICLOS DE VIDA DE SOFTWARE

CICLO DE VIDA E O PROCESSO DE PRODUÇÃO DE SOFTWARE

Todas as práticas que estudaremos nesta disciplina podem ser aplicadas a qualquer processo de software.

Em nosso projeto de disciplina e no desafio do ano (Challenge), aplicaremos a abordagem de condução Ágil, usando o Ciclo de Vida Espiral.

Sem entrar em detalhes neste momento, estaremos guiados pelo **SCRUM** nesses projetos (o modelo SCRUM será estudado em todos os seus detalhes no próximo ano, na disciplina que trata boas práticas, qualidade de software e governança em projetos).

SCRUM Funções da equipe e responsabilidades

A equipe de um projeto SCRUM tem a seguinte distribuição de papéis e responsabilidades:



Product Owner (PO)

Responsável por garantir o ROI (Retorno de Investimento)
Responsável por conhecer as necessidades do(s) cliente(s)
(único por produto a entregar)



ScrumMaster (SM)

Responsável por remover os impedimentos do time
Responsável por garantir o uso de Scrum
Protege o time de interferências externas
(único por time Scrum, podendo ser compartilhado com outros times)



Time

Definir metas das iterações
Auto-gerenciamento
Produzir produto com qualidade e valor para o cliente

SCRUM Funções da equipe e responsabilidades

A equipe de um projeto SCRUM tem a seguinte distribuição de papéis e responsabilidades (segundo o SBOK):



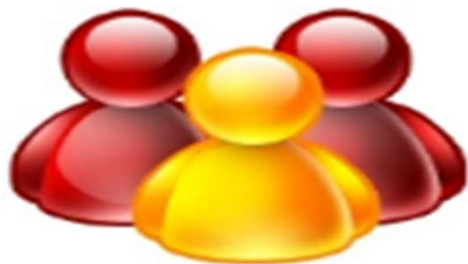
PO: Único por produto a entregar

Se se tratar de um projeto complexo ou um programa com vários projetos, existindo vários produtos/soluções de negócio a entregar, existirá um PO por frente de solução



MASTER: Único por time Scrum

Dependendo da característica de demanda dos projetos, o Scrum Master pode ser compartilhado entre mais de um time/frente de desenvolvimento de solução.



SQUAD (time): Composto por várias pessoas

Os profissionais do Time devem ter múltiplas e complementares competências para lidar com todas as tarefas de desenvolvimento (gestão de projeto, modelagem e sistema, construção, teste, etc.)

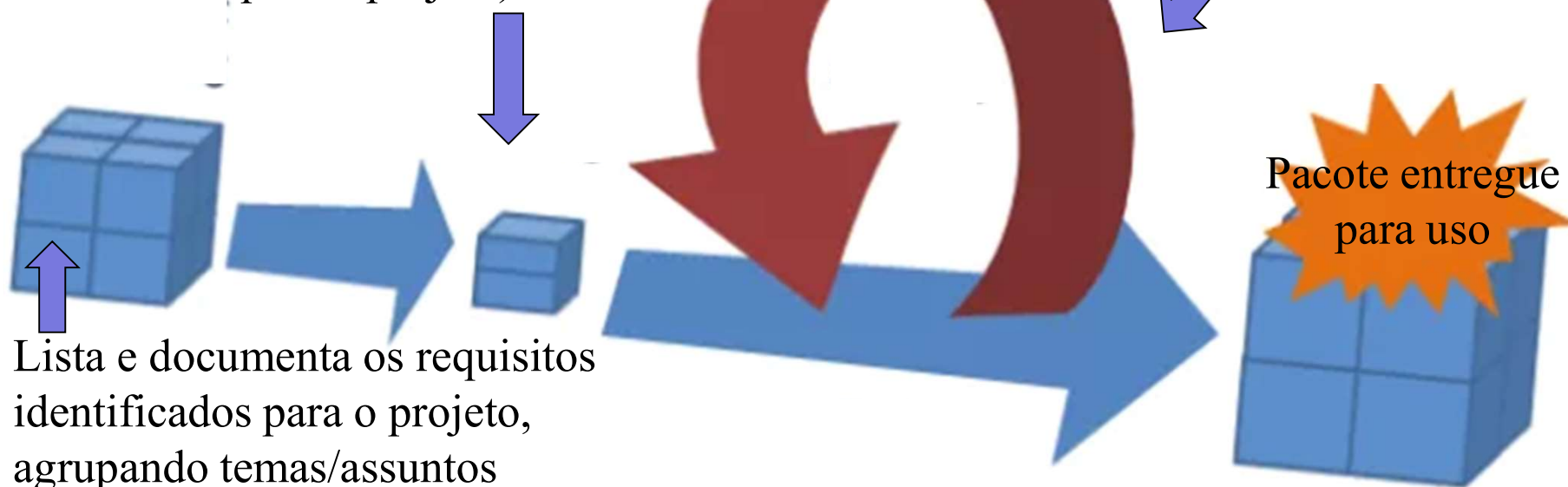
CICLOS DE VIDA DE SOFTWARE

CICLO DE VIDA E O PROCESSO DE PRODUÇÃO DE SOFTWARE

SCRUM

Define um requisito de uma lista a ser produzido no projeto (a lista de requisitos não precisa estar completa para o projeto)

Produz o pacote/peça do software, acompanhando e repostando resultados

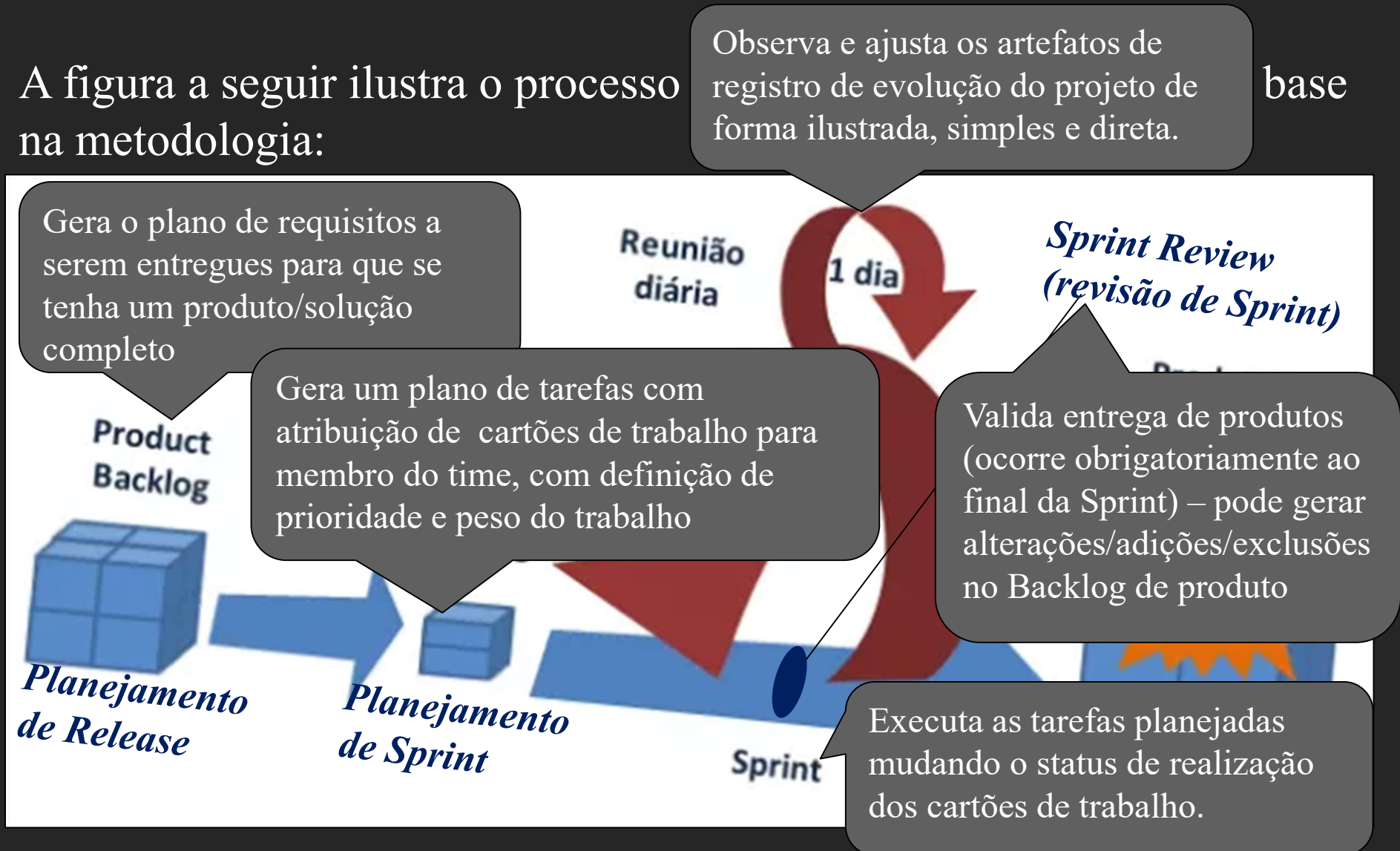


Lista e documenta os requisitos identificados para o projeto, agrupando temas/assuntos

Pacote entregue para uso

SCRUM

A figura a seguir ilustra o processo na metodologia:



SCRUM

A figura a seguir ilustra o processo de produção de software com base na metodologia:

TIME BOXES



CICLOS DE VIDA DE SOFTWARE

No momento de **implantar o software** desenvolvido (momento do Go-Live) quando o produto ganha vida e utilidade real, podem ser adotadas as seguintes **estratégias** de entrega para os usuários:



TURN KEY: desliga-se o sistema antigo e liga-se o novo no dia da “virada” para a produção.



PILOT & ROLL OUT o sistema antigo vai sendo substituído pelo novo aos poucos. Em geral, escolhe-se uma unidade da empresa ou atividade para começar a virada e depois rola para as demais, seguindo um cronograma que dê segurança ao processo de mudança. Tem custo e esforço de manter os ambientes antigo e novo operando juntos mas esse custo decai conforme as implantações vão acontecendo.



PARALLEL: o sistema antigo continua operando junto com o novo. Implica em dupla digitação de dados, processamento e operação dos dois sistemas com as rotinas de segurança e backup acontecendo juntas. Possui custo elevado em função do trabalho dobrado da equipe técnica e usuários e da infraestrutura que deve atender aos dois ambientes. Os custos elevados se mantêm até o final das implantações.

CICLOS DE VIDA DE SOFTWARE**DEBATE**

Quando devemos usar cada uma dessas estratégias de implantação?

TURN KEY (VIRADA DE CHAVE)

PILOTO E ROLAGEM (PILOT & ROLL OUT)

PARALLEL (PARALELO)

CICLOS DE VIDA DE SOFTWARE

Quando usar cada estratégia de implantação?



TURN KEY: os custos de manutenção dos dois ambientes (antigo e novo) são proibitivos ou os riscos operacionais de uma mudança completa são baixos (existe como recuperar dados da empresa e reestabelecer processos com certa rapidez em caso de falha no novo software).



PILOT & ROLL OUT: existirão resistências à implantação por parte dos usuários e a implantação faseada, iniciando pelos usuários menos resistentes irá gerar exemplificação de sucesso e convencimento dos demais usuários, facilitando a introdução da nova cultura de trabalho que o sistema impõe.



PARALLEL: os processos da rotina empresarial comprometidos no software são muito sensíveis e pertencem à cadeia de valor ao cliente (gera produtos e serviços percebidos diretamente pelo cliente), e existe risco de uma falha no software comprometer a imagem e os resultados financeiros e operacionais finais.

OBS: O momento imediatamente após o Go-Live deve ser acompanhado de perto pela equipe de TI com o pronto apoio dos desenvolvedores, se for necessário. Esse período conhecido como estabilização pode durar de alguns dias até alguns meses, dependendo da complexidade do software e a quantidade de usuários e suas localizações de trabalho.

CICLOS DE VIDA DE SOFTWARE

Você deve optar por um ciclo de vida de software em seus projetos!

Uma empresa pode adotar mais de um tipo de ciclo de vida, dependendo das características do projeto de software.

CICLOS DE VIDA DE SOFTWARE

Vamos jogar!

Valendo um brinde!



Kah??t!

Jogo de aprendizado de CVS

<https://kahoot.it/#/>

Os **ciclos de vida** de software determinam:

- Fases da produção de software com objetivos bem definidos

Os **processos** de software aplicam um dos modelos de ciclo de vida e o complementam com:

- Detalhes de como executar as tarefas em cada fase;
- Papéis e responsabilidades em cada atividade;
- Entradas e saídas esperadas por atividade e fase;
- Artefatos a serem empregados;
- Indicadores e controles de desempenho e resultados.

Você conhecerá
mais sobre
processo de sw
na disciplina de
gestão de
qualidade!



**EXECUÇÃO DE UM PROJETO
DE SOFTWARE – VISÃO
GERAL DE UM
CRONOGRAMA DE
ATIVIDADES**

Os **ciclos de vida de software** são a base para a definição do processo de trabalho que contempla as **tarefas a serem feitas** pela equipe de projeto ao longo do tempo.

O planejamento de um projeto passa pelas etapas de:

1. **Definição do escopo** do software (requisitos/entregas esperadas)
2. **Escolha do ciclo de vida** ideal para o tipo de projeto
3. **Definição das tarefas** para desenhar e produzir os itens do escopo
4. **Atribuição de responsáveis** por cada tarefa
5. **Definição de tempo** de trabalho com base na competência de cada profissional alocado para cada tarefa e nas ferramentas que cada profissional possui para apoiá-lo
6. **Determinação dos custos** de mão-de-obra, aquisição de software e hardware que serão usados como plataforma de desenvolvimento (servidores, equipamentos de usuário, sistemas operacionais, sistemas gerenciadores de bancos de dados, sistemas de segurança, ferramentas de apoio à engenharia de software).

As **determinações de tempo e custo de projeto** são grandes desafios do planejamento.

Para auxiliar nesse trabalho de estimativa, **existem modelos de previsão e projeção baseados em métricas** (que serão estudados em outra disciplina).

Outras formas de estimar projetos são: por **analogia** a projetos anteriormente realizados e por **opinião de especialistas**.

Ao final, um cronograma com releases (entregas) previstas, recursos e custos associados é traçado, e depois acompanhado pela equipe de projeto.

As **determinações de tempo e custo de projeto** são grandes desafios do planejamento.

Para auxiliar nesse trabalho de estimativa, **existem modelos de previsão e projeção baseados em métricas** (que serão estudados em outra disciplina).

Outras formas de estimar projetos são: por **analogia** a projetos anteriormente realizados e por **opinião de especialistas**.

Ao final, um cronograma com releases (entregas) previstos e custos associados é traçado, e depois acompanhado pelo projeto.

Você conhecerá mais sobre processo de sw na disciplina de gestão de qualidade e governança!



CANAL DO PROFESSOR

YouTube



Vídeos de
complementares
de aulas

Se você quer aprender mais sobre o planejamento clássico, assista ao vídeo sobre Planejamento de Projetos com MS-Project no canal do professor!

<https://youtu.be/KEjI5CNdV88>

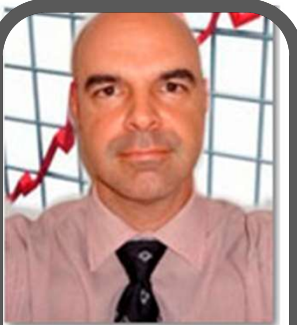
Gerenciamento da Documentação do Projeto

Toda a documentação do projeto deve ser administrada de forma a:

- Toda a **equipe** de desenvolvimento **saber onde está a fonte oficial** de informação;
- Ninguém **fazer alterações** diretamente na fonte oficial mas **em uma cópia**, sendo a fonte oficial atualizada após a confirmação de que as modificações na cópia estão corretas;
- Permitir **fácil e rápido acesso** dos projetistas aos documentos, **com regras de segurança** de acesso aplicadas;
- Todos **saberem quando um documento oficial** se encontra **em estado de revisão** com cópia sendo editada, sendo **conhecido o responsável** pela edição em curso.

CANAL DO PROFESSOR

YouTube



Vídeos de
complementares
de aulas

Assista ao vídeo sobre Administração de Conteúdo no canal do professor!

<https://youtu.be/Z76STqABbqs>

Observação: nesse momento, não estudaremos nenhum software de apoio ao gerenciamento de conteúdo!

Use as dicas sobre padronização de nomes de arquivos e diretórios propostos no vídeo para você administrar seus arquivos de projeto!

ATIVIDADE EXTRA

Defina grupos de 2 pessoas para trabalharem o projeto que desenvolveremos durante a disciplina e as atividades que serão feitas em sala como exercícios de fixação.

No seu grupo, defina um conjunto de convenções de nomes de arquivos e pastas que permitam organizar toda a documentação que vocês produzirão nos projetos que virão. Tome por base o modelo Cascata e os tipos de conteúdos de cada fase trabalha.

*Validem com
o professor
quando
concluírem*

ATIVIDADE EXTRA

Vamos trocar nossas ideias!

Objetivo: definirmos juntos um padrão para organizar os documentos e arquivos fonte de um projeto de software, usando uma estrutura de diretórios.

Cada grupo expõe a sua proposta, o professor anota de forma que todos vejam. **Ninguém debate durante essa exposição!**

Depois que todos falarem, negociaremos um padrão comum.

Padronização

ATIVIDADE EXTRA

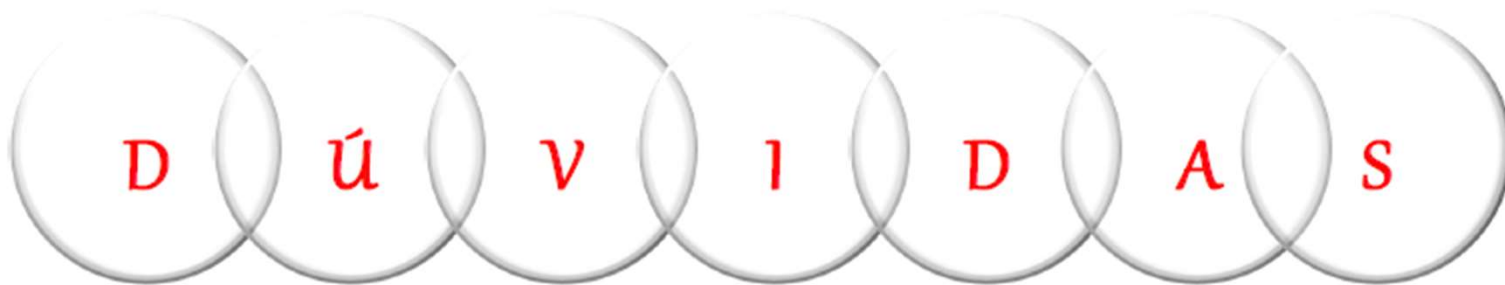
Agora que alinhamos um padrão para todas as equipes...

Crie a estrutura de pastas que servirá como repositório de arquivos em um Drive na nuvem.

Crie também uma estrutura de diretórios em outro drive na nuvem que funcionará como backup.

Defina a política para realizar as cópias de segurança (backup): periodicidade, responsável.

Chamem o professor ao terminarem



Material de aula estará no site após a aula.

BONS ESTUDOS!

Bibliografia

- PRESSMAN, R. S. Engenharia de software. São Paulo: Editora McGraw-Hill, 2002.
- SOMMERVILLE, IAN. Engenharia de software. Editora Pearson, 9.ed. - São Paulo, 2014.
- BEZERRA, EDUARDO. Princípios de análise e projeto de sistemas com UML. Alta Books, Rio de Janeiro, 2006.