

FIAP GRADUAÇÃO

DISCIPLINA: DATA GOVERNANCE

AULA:

2 – CICLO DE VIDA DE PROJETOS DE BANCOS DE DADOS

PROFESSOR:

RENATO JARDIM PARDOCCI

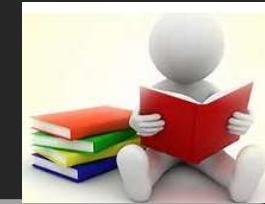
PROFRENATO.PARDOCCI@FIAP.COM.BR

AGENDA DA AULA

- ✓ Ciclo de Vida de um Banco de Dados
- ✓ Ciclo de Vida de Projetos de Bancos de Dados

Ciclo de Vida de um Banco de Dados

ESTUDO DE CASO SIMULADO



A LuxoDoLixo ouviu falar que os projetos de sistemas de informação com bancos de dados são dispendiosos e quer saber se, uma vez feito esse investimento e entregue a solução, ela nunca mais precisará gastar dinheiro com tecnologia.

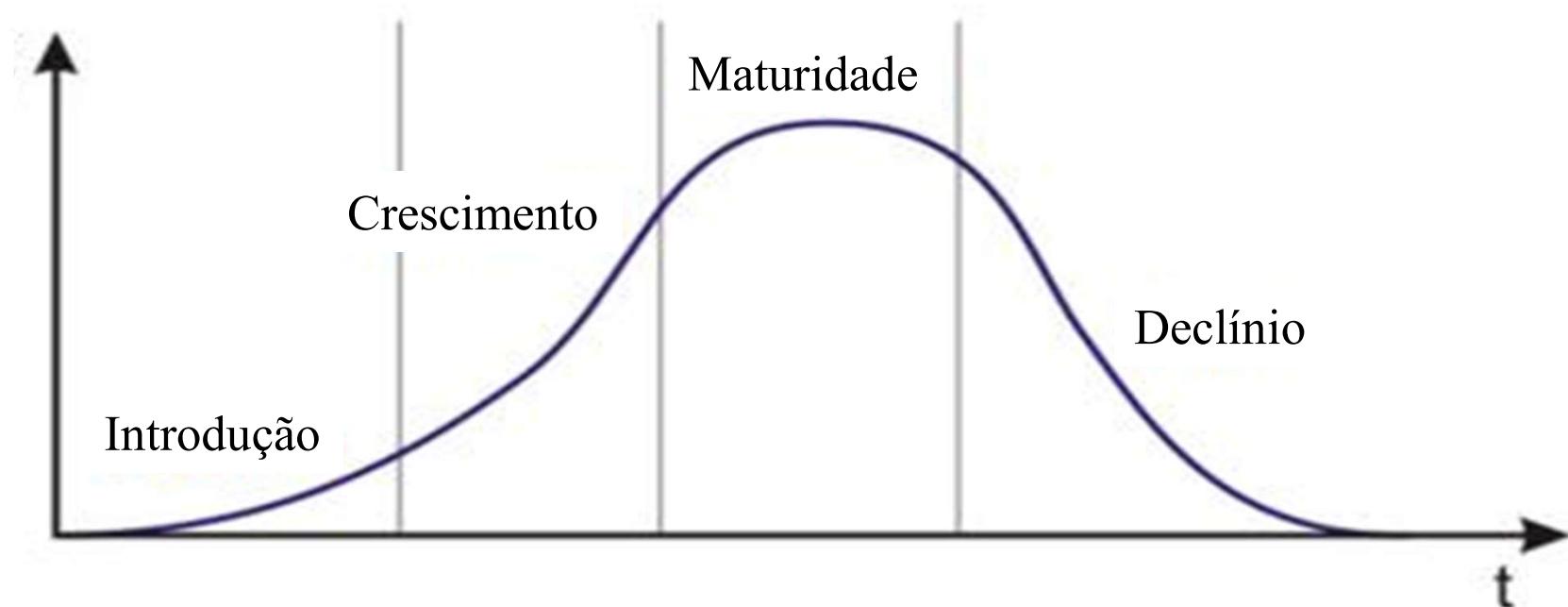
Você, da SuperSoluções precisa explicar se essa expectativa será atendida e justificar por que sim ou por que não.

Para isso, estude Ciclo de Vida de um software de aplicação e seu banco de dados, pós entrega/lançamento.

A base para a Engenharia de Banco de Dados e Software associado

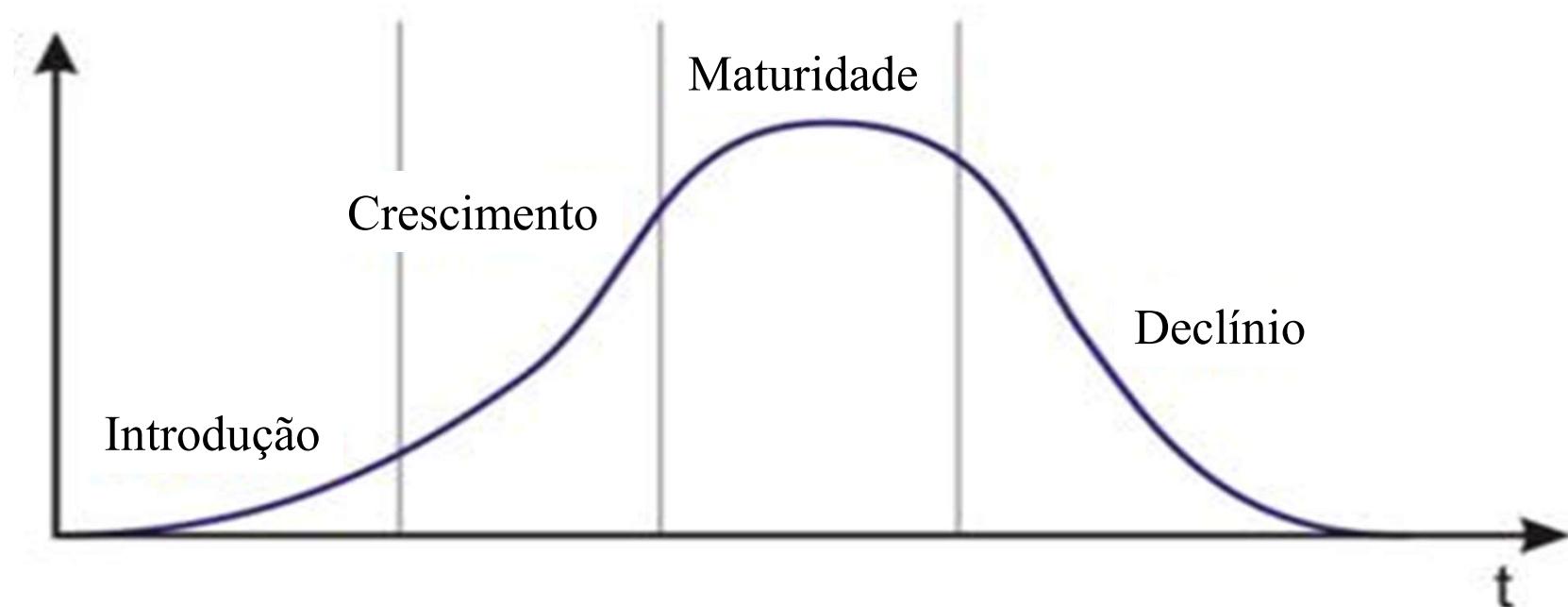
Todo produto tem uma CURVA DE VIDA

Nível de uso e
Remuneração



Essa CURVA DE VIDA é acompanhada por um CICLO DE VIDA que são as Fases pelas quais passa a solução

Nível de uso e
Remuneração



O Ciclo da Vida de QUALQUER PRODUTO

- Define as **fases pelas quais se passa**, da introdução à aposentadoria
- Define o que se pode esperar **como resultado** do cumprimento de cada fase

O ciclo de vida pode ajudar no planejamento e obtenção do retorno financeiro esperado.

Seja qual for o modelo de consumo, quanto mais o produto estiver vivo na praça, maior será o retorno que o produto trará!



Como dar sobrevida?

Manutenções em produtos prolongam a sua vida.

É o que acontece no versionamento de bens duráveis e software!

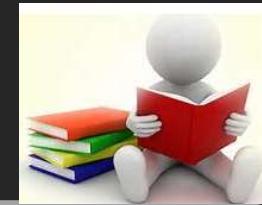


Manutenções podem ser do tipo:

- **Corretivas:** eliminam defeitos do Banco de Dados e programas associados
- **Adaptativas:** ajustam o Banco de Dados e programas associados para acomodar mudanças em regras de negócio que exigem novas funções e informações
- **Evolutivas:** incluem novas estruturas em Bancos de Dados e funcionalidades em programas associados aos Bancos, para criar novas oportunidades de aplicação prática para os usuários, aumentando a atratividade do produto
- **Perfectivas:** buscam ajustes de desempenho (Performance Tuning), otimização de ocupação de espaços de armazenamento de dados e consumo de recursos infraestruturais (processamento, RAM, telecomunicações).

Ciclo de Vida de Projetos de Bancos de Dados

ESTUDO DE CASO SIMULADO



Agora que o senhor LexLuxor entendeu como funciona a questão de manutenção do software pós entrega, ele quer entender como funcionará o projeto em si. Quer saber como um projeto é executado e controlado, quais são as principais atividades envolvidas e como acontecem.

A preocupação é saber se a SuperSoluções tem um método claro para administrar o projeto de forma adequada.

É hora de escolher um Ciclo de Vida e um Processo de projeto de sistema!

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!

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!

Na Engenharia de Software que inclui a Engenharia de Bancos de Dados, desenvolveu-se diversos modelos conceituais de Ciclo de Vida para projetos:

-**Modelo de processo Cascata**

-**Modelo de processo Incremental**

-**Modelo de Prototipação**

-**Modelo de processo Espiral**

Esse modelos...

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

CICLOS DE VIDA DE SOFTWARE

Modelos que podem ser seguidos para gerenciar a produção de software:

- Cascata
- Incremental
- Prototipação Evolucionária
- Espiral

CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA

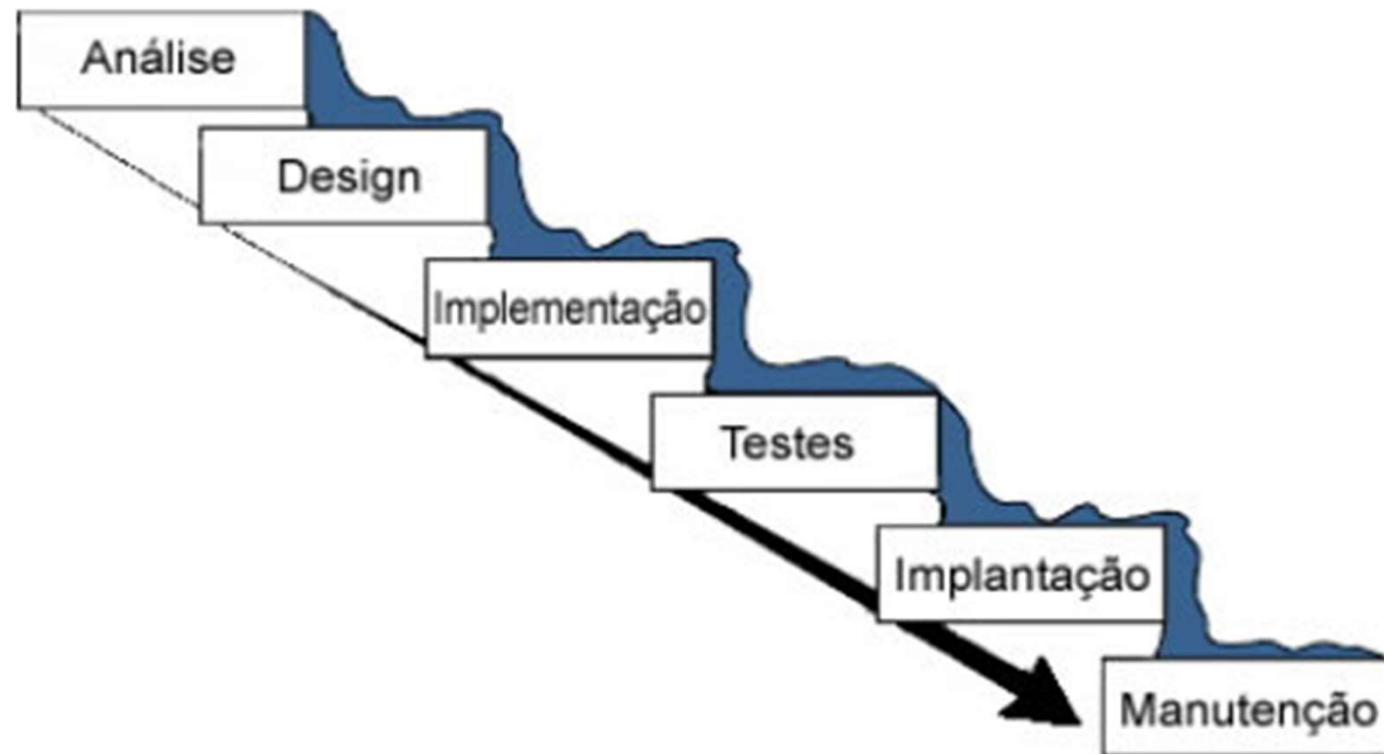
O projeto inteiro é tratado como uma frente de desenvolvimento única!

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

Requisitos

Estudo de viabilidade

Modelagem de Projeto

Documentação de engenharia

Codificação de software, teste

Criação de manuais de uso e operação

Teste integrado de solução e homologação

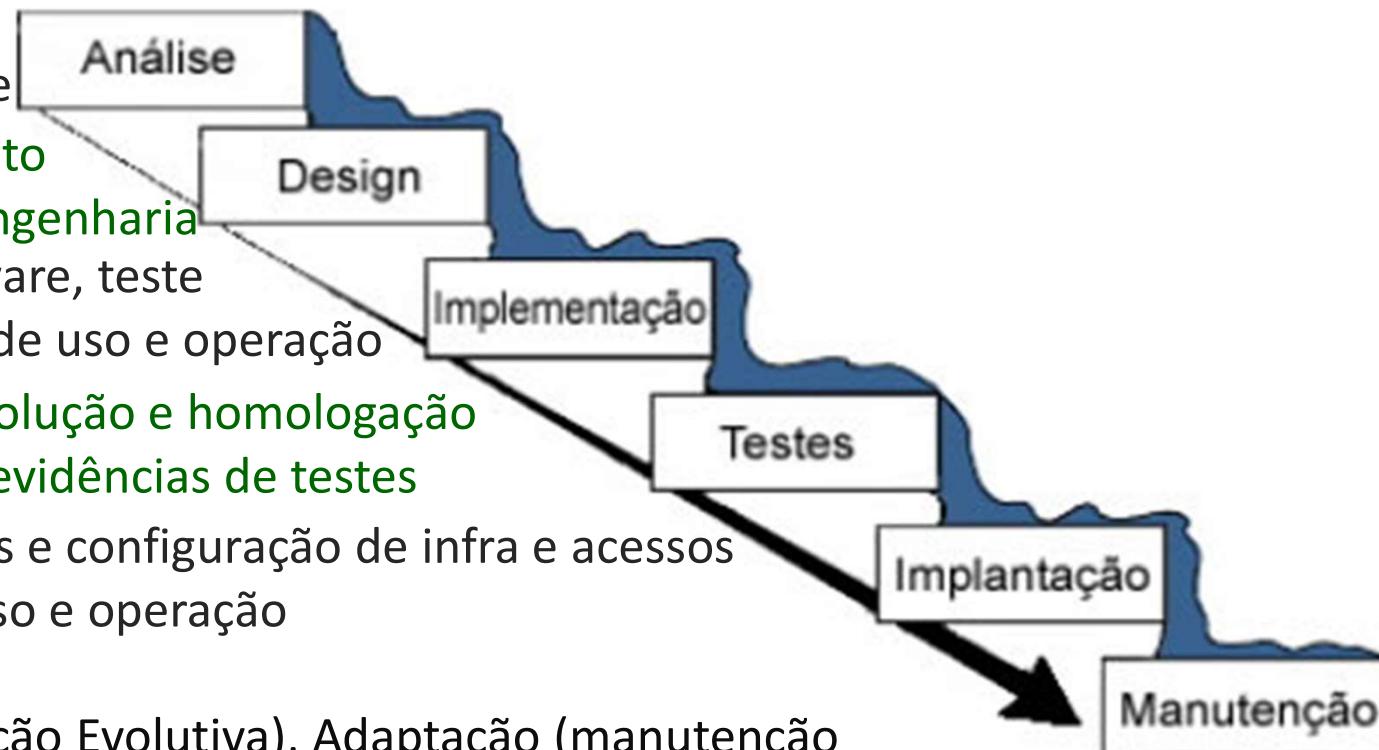
Documentação das evidências de testes

Gestão de mudanças e configuração de infra e acessos

Treinamento para uso e operação

Evolução (manutenção Evolutiva), Adaptação (manutenção

Adaptativa), Correção (manutenção Corretiva), Previsão (ajustes prevendo possíveis necessidades futuras), Busca da perfeição no desempenho e uso (manutenção Perfectiva)

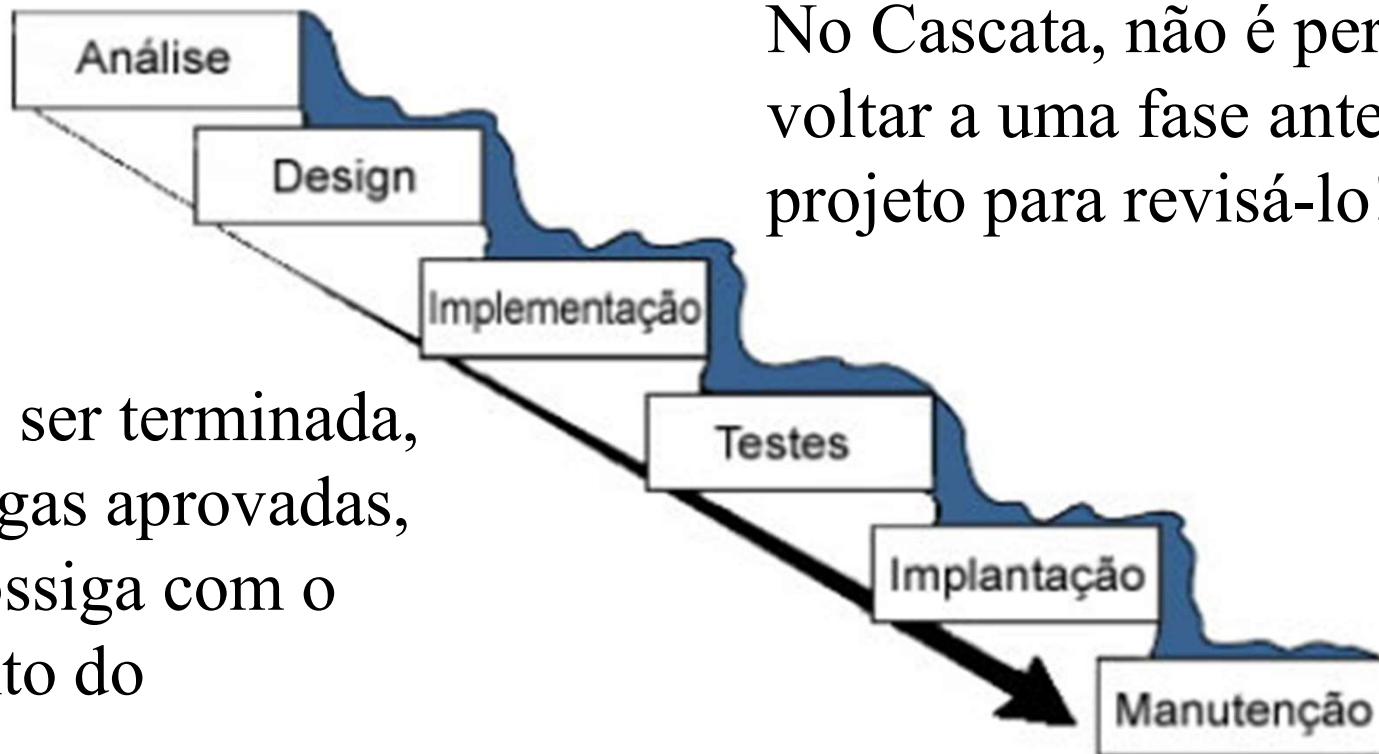


CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA

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!



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

CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA – EXEMPLO DA APLICAÇÃO PARA BD

Descobrir o que o banco de dados deve guardar e as operações que serão feitas sobre

Desenhar as tabelas do banco e explicar seus conteúdos e programas de manipulação

Criar os repositórios digitais em um computador / dispositivo de armazenamento e criar programas de acesso e atualização do banco,

Executar os programas rodando em conjunto e avaliar se funcionam

Levar as estruturas dos arquivos e os programas já aprovados para um novo computador e configurar usuários de acesso

Fazer cópias de segurança (backup) dos dados, limpar dados desnecessários, etc

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!

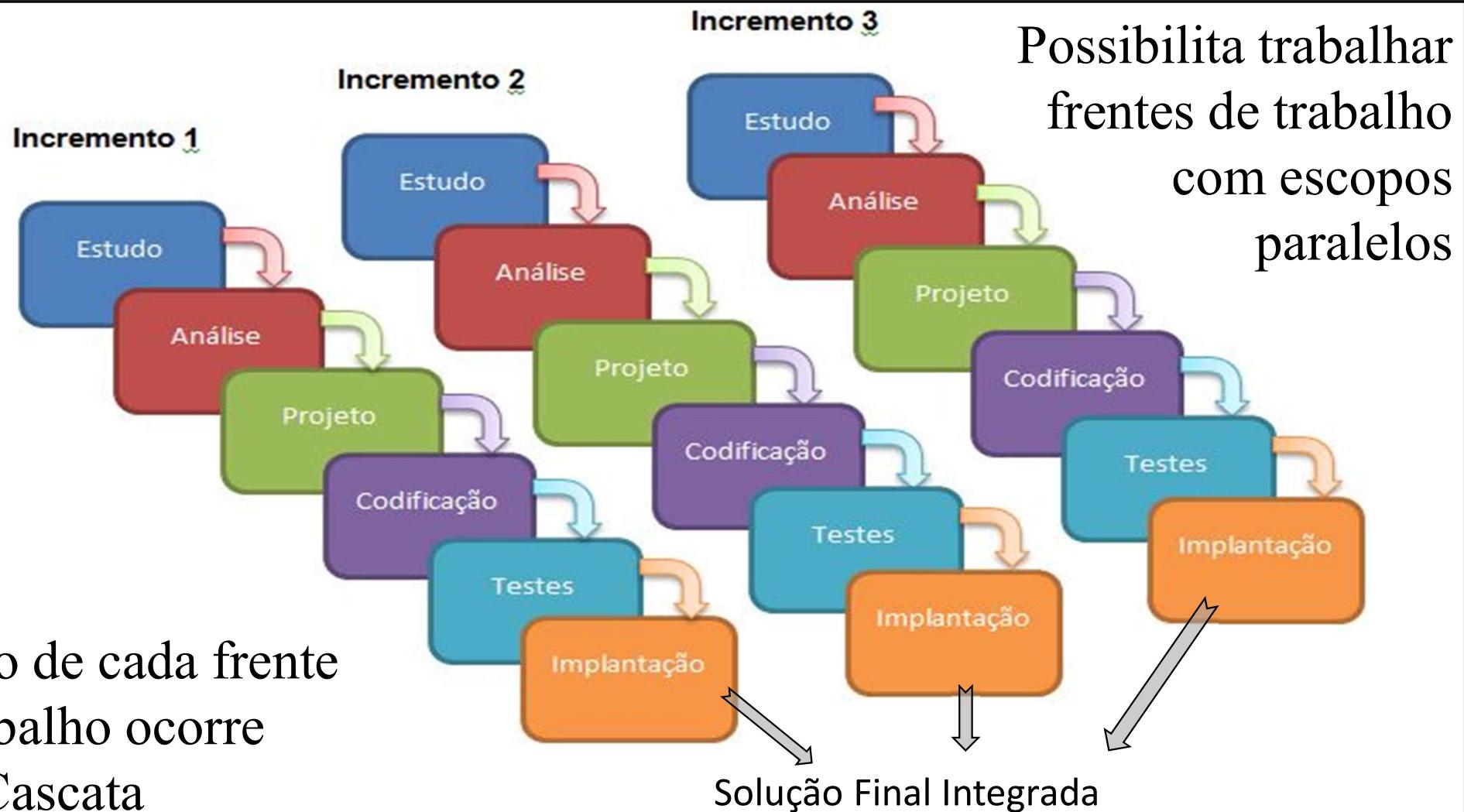
CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL



CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL



CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL – EXEMPLO DA APLICAÇÃO P/ BD

Módulo de projeto de desenvolvimento dos bancos de dados cadastrais de clientes da

Módulo dos bancos de dados de saldo e movimento de estoque

Módulo dos bancos de dados de vendas



Banco de dados completos do sistema de controle vendas de loja integrado com a administração de estoques e clientes

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

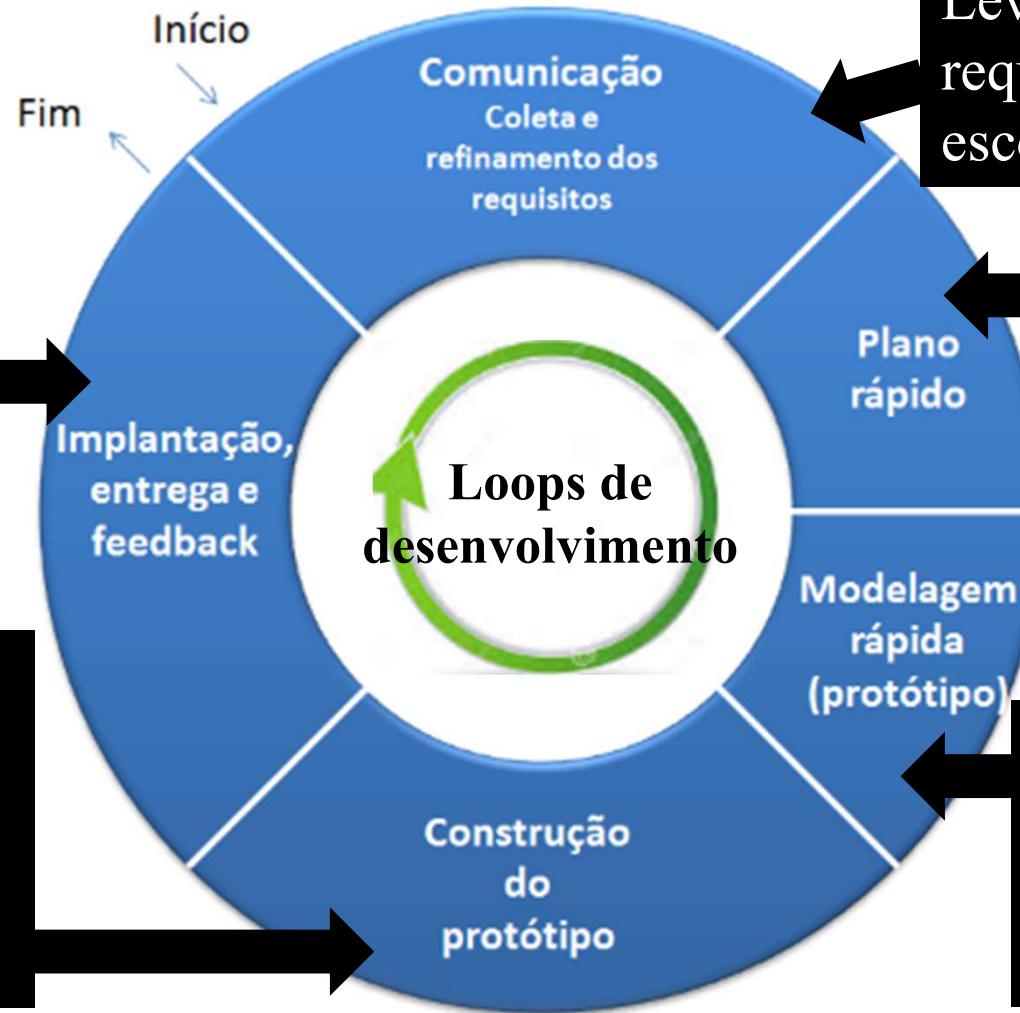


CICLOS DE VIDA DE SOFTWARE

MODELO PROTOTIPAÇÃO EVOLUCIONÁRIO

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



Os ciclos de desenvolvimento acontecem até que o produto esteja completo

Levantar e documentar requisitos. Definir escopo do produto

Definir atividades de trabalho, distribuir tarefas, definir prazos

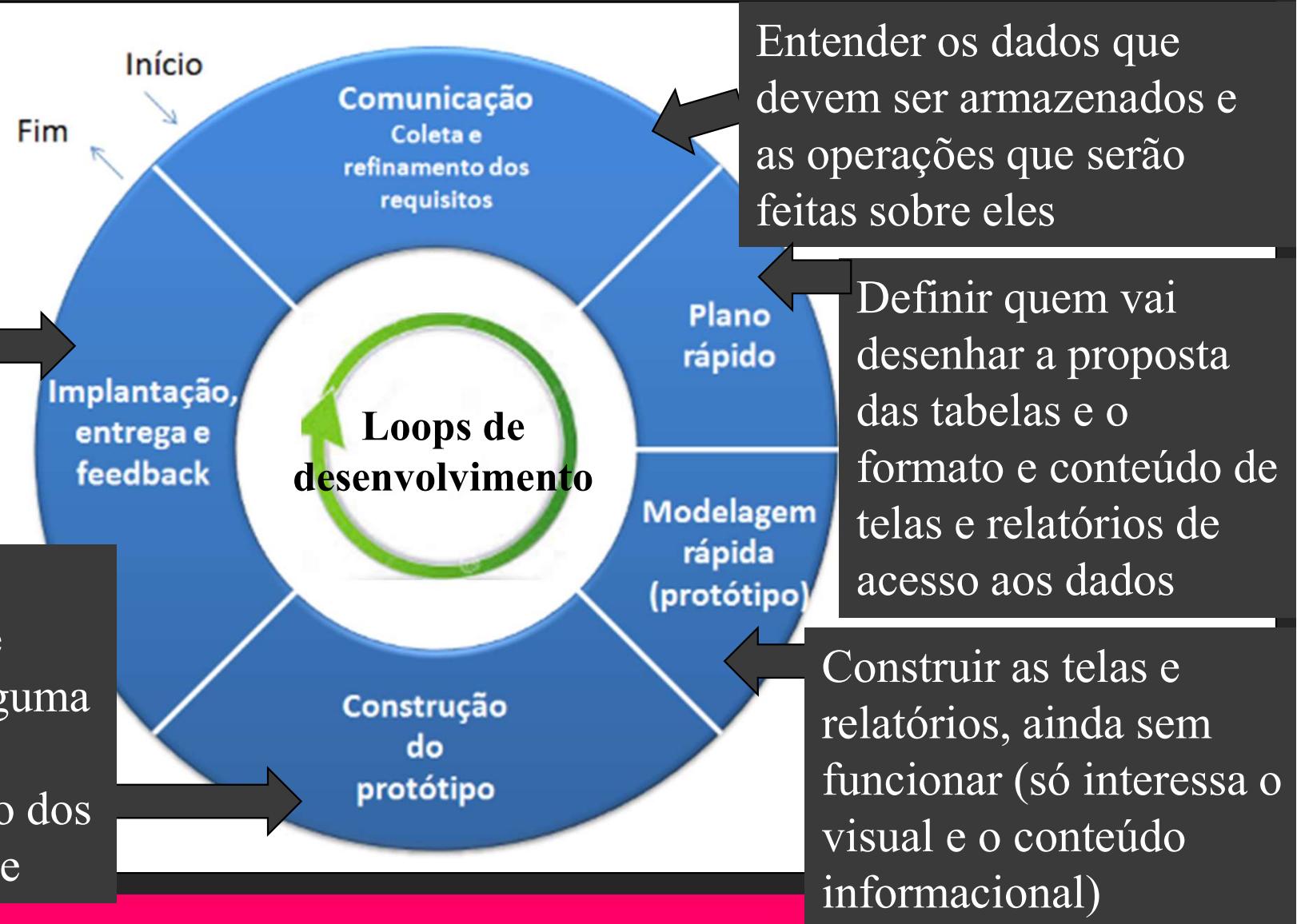
Elaborar prova de conceito (POC), e protótipo do produto (MOCUP)

CICLOS DE VIDA DE SOFTWARE

MODELO PROTOTIPAÇÃO EVOLUCIONÁRIA – EXEMPLO P/BD

Levar as estruturas dos arquivos e os programas já aprovados para um novo computador e configurar usuários de acesso

Construir a versão definitiva das telas e relatórios usando alguma ferramenta, após ter ocorrido a aprovação dos desenhos pelo cliente



CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL

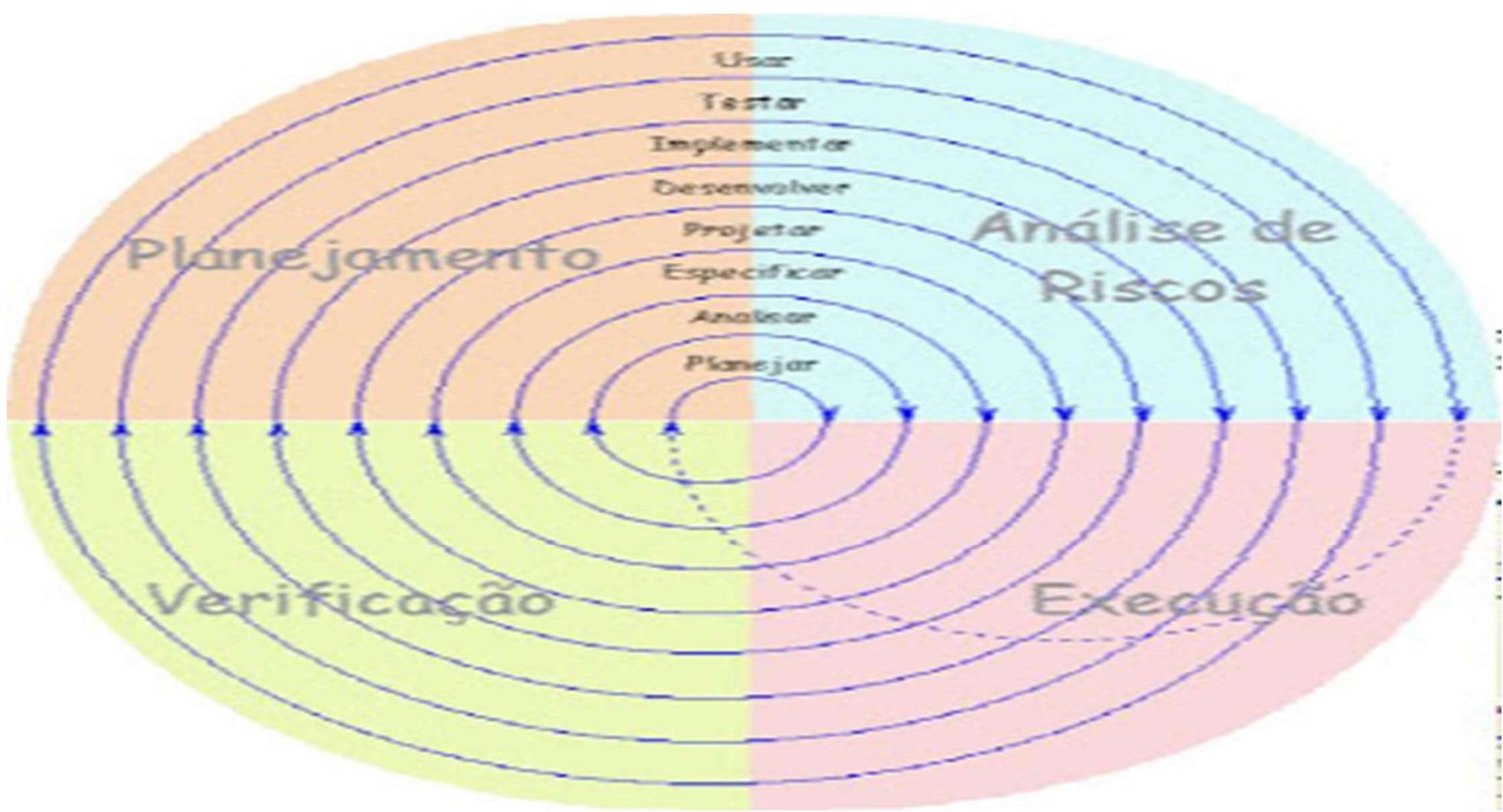
O projeto pode ser desenvolvido com flexibilidade de adaptação, onde o software pode ser repartido e ter módulos e até componentes individuais sendo evoluídos em ritmo distinto dos demais.

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

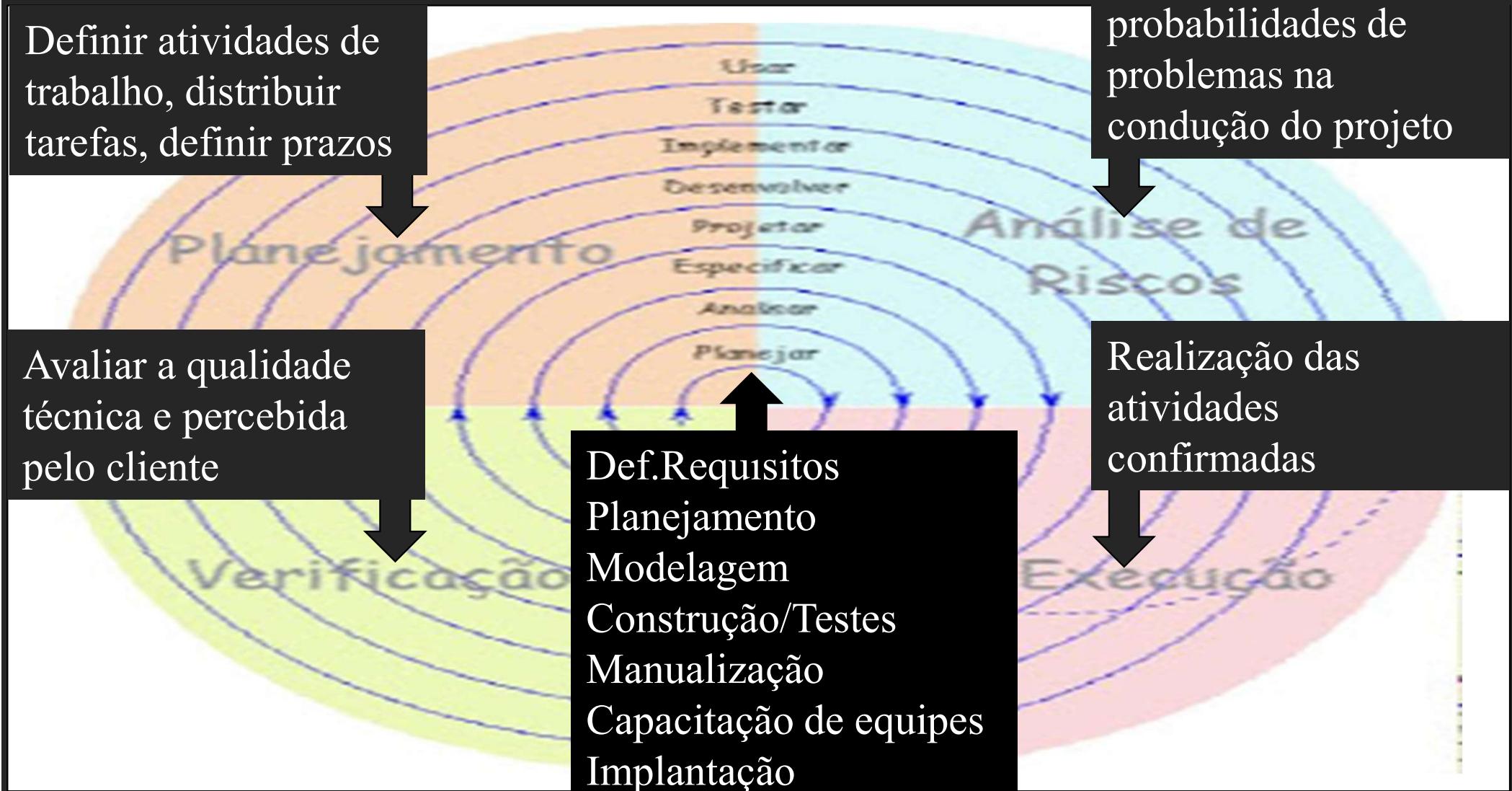
Definir atividades de trabalho, distribuir tarefas, definir prazos

Avaliar a qualidade técnica e percebida pelo cliente

Def.Requisitos
Planejamento
Modelagem
Construção/Testes
Manualização
Capacitação de equipes
Implantação

Avaliar impactos e probabilidades de problemas na condução do projeto

Realização das atividades confirmadas



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

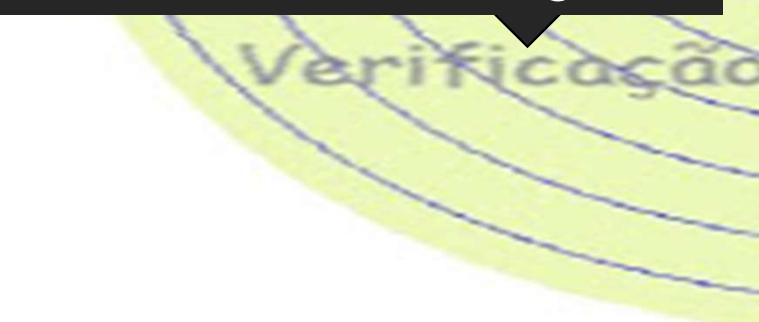
CICLOS DE VIDA DE SOFTWARE

MODELO ESPIRAL – EXEMPLO PARA BD

Definir quais padrões serão seguidos para modelar, construir, testar, entregar

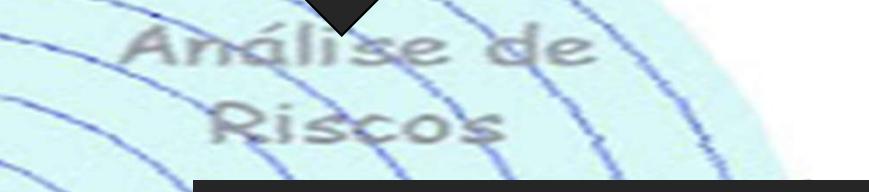


Avaliar a qualidade técnica e percebida pelo cliente do que foi desenhado ou entregue



Partes do projeto podem estar em fases diferentes: uma tabela foi construída com os programas de acesso e liberada para uso, outra está em construção e outra em desenho

Avaliar possibilidade e impacto de desconhecimentos técnicos, prazos e custos para realizar atividades



Desenhar esquemas do banco, dicionários de dados, construir programas e repositórios digitais, etc.



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 mantém até o final das implantações.

CICLOS DE VIDA DE SOFTWARE

DEBATE



Quando devemos usar cada uma dessas estratégias de implanação?

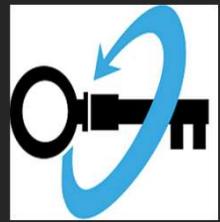
TURN KEY (VIRADA DE CHAVE)

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 pertençam 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.

VAMOS TRABALHAR DAQUI POR DIANTE
COM O CICLO DE VIDA ESPIRAL

Ciclos de Vida de Projeto – considerações importantes:

- Estabelecem fases onde são esperadas determinadas atividades e resultados
- Não detalham como realizar atividades, não criam padrões de trabalho e reporte e nem define responsabilidades por atividades
- – isso precisa ser feito com um detalhamento de processo de trabalho
- O tipo de Ciclo de Vida que será adotado pode ser escolhido em função das características do projeto.

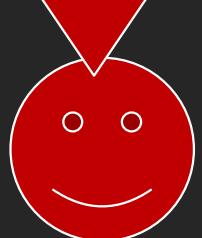
Os **ciclos de vida** de software determinam:

- 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.

UM DOS
MODELOS É O
SCRUM!
VOCÊ TERÁ
UMA VISÃO
GERAL DELE A
SEGUIR E MAIS
DETALHES NO
PRÓXIMOANO!

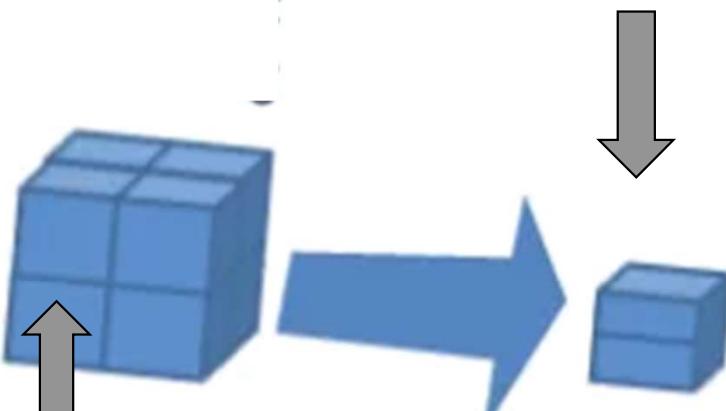


SCRUM – APLICANDO O CICLO DE VIDA ESPIRAL

CICLO DE VIDA E O PROCESSO DE PRODUÇÃO DE SOFTWARE

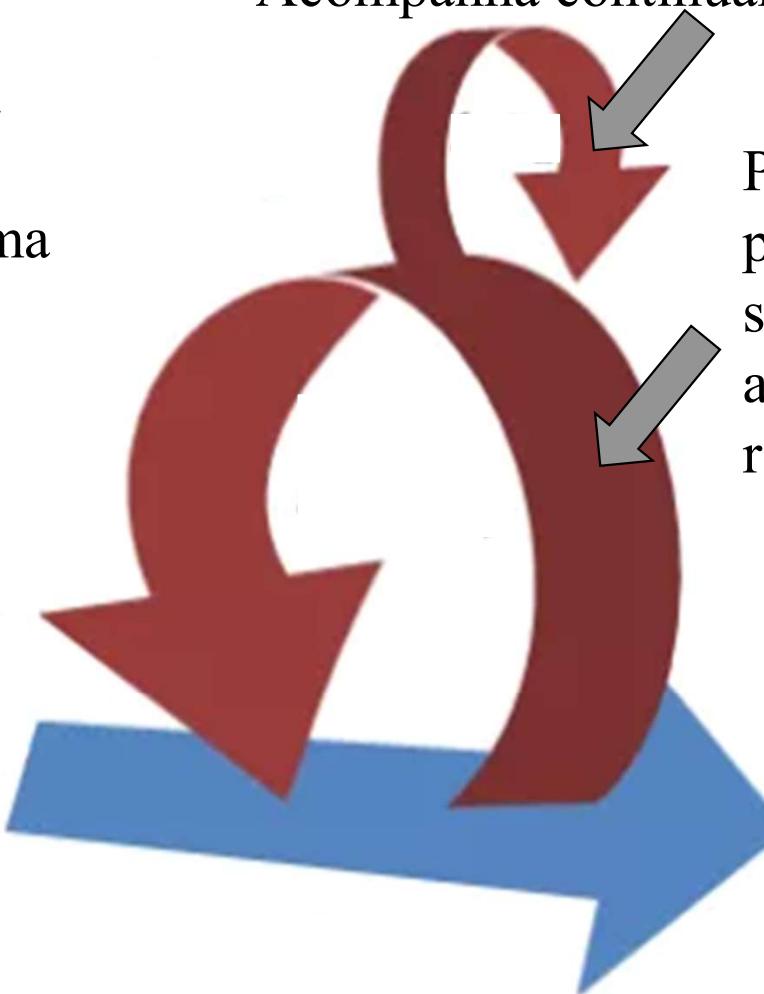
SCRUM

Define um/alguns requisito(s) de uma lista a ser(em) produzido(s) em uma corrida de desenvolvimento



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

Acompanha continuamente os resultados



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



Pacote entregue para uso (pedaço útil do produto geral)

SCRUM – APLICANDO O CICLO DE VIDA ESPIRAL

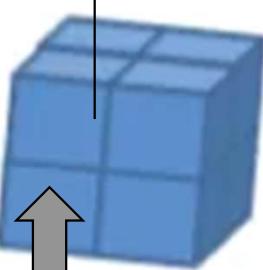
CICLO DE VIDA E CICLO DE VIDA ESPIRAL

SCRUM

Define um/algum

O que o BD precisa atender:

- Administrar clientes da loja
- Acompanhar vendas
- Administrar vendedores
- Pagar comissões



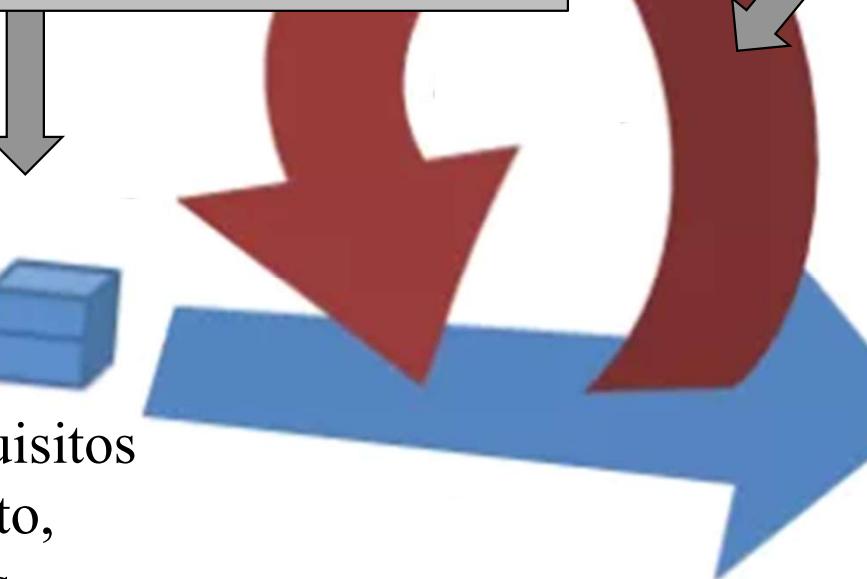
Lista e documenta os requisitos identificados para o projeto, agrupando temas/assuntos

O que fazer nos próximos 10 dias de trabalho:

- Administrar clientes da loja
- Modelar as tabelas (MER/DER)
- Dicionário de dados
- Criar tabela no Oracle
- Criar chaves de registros
- Criar SQL de inclusão
- Criar SQL de consulta

Quais são as tarefas:

- Modelar as tabelas (MER/DER)
- Dicionário de dados
- Criar tabela no Oracle
- Criar chaves de registros
- Criar SQL de inclusão
- Criar SQL de consulta



RODUÇÃO DE SOFTWARE

anha conti

Reunião diária para verificar evoluções das tarefas

REALIZAR tarefas:

- Modelar as tabelas (MER/DER)
- Dicionário de dados
- Criar tabela no Oracle
- Criar chaves de registros
- Criar SQL de inclusão
- Criar SQL de consulta



Produto:

- Tabela de Clientes
- Programas de cadastro e consulta

geral)

SCRUM Funções da equipe e responsabilidades

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



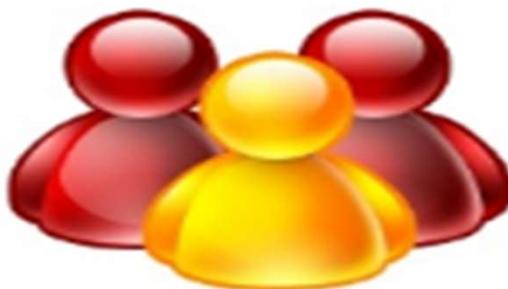
Product Owner (PO)

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)



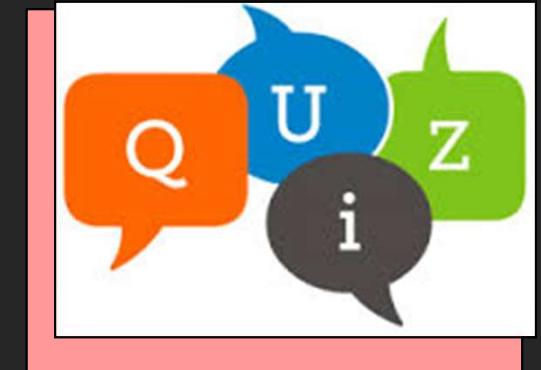
Time (SQUAD)

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

CICLOS DE VIDA DE SOFTWARE

Vamos jogar!

Valendo um brinde!



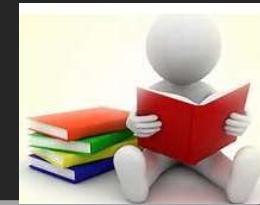
Kah??t!

Jogo de aprendizado de CVS

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

Gestão de Conteúdo

ESTUDO DE CASO SIMULADO



O projeto foi contratado e vai começar!

A partir de agora, todos os documentos e arquivos de programa de computador que forem produzidos precisam ser guardados com segurança e versionados.

Usar diretórios do computador (pastas) e renomear arquivos a cada nova versão é um procedimento perigoso – pessoas podem editar arquivos e salvar “por cima” sem criar um novo nome, perdendo o histórico das diversas versões – alguém pode apagar um conteúdo indevidamente e não conseguirmos recuperar mais.

Existem soluções de gestão de conteúdo que são mais adequadas ao controle de artefatos de projeto.

Conheça agora!

Ao longo do ciclo de vida de produção do software/banco de dados, diversos artefatos serão coletados ou produzidos, e armazenados para uso ao longo do projeto.

- Cópias de documentos físicos que o demandante do sistema usa e que serão convertidos em sistema digital;
- Desenhos de estruturas de bancos de dados;
- Fontes de programas SQL;
- ...



ASSISTA OS VÍDEOS NO CANAL DO PROFESSOR



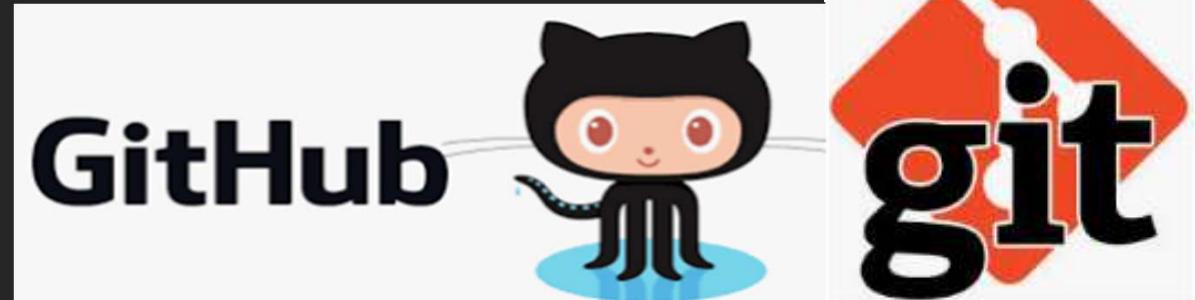
<https://youtu.be/t1E-cbB4gFY>

A screenshot of a Windows File Explorer window. The left sidebar shows a navigation tree with 'Google Drive' selected. The main pane displays three folders: 'Metodologia', 'Docencia (I)', and 'Recibos 2016', all listed as 'Pasta de arquivos'. To the right of the file list is a video player interface showing a man wearing a headset and speaking. The video player has a play button and other control icons.

Conteúdo didático complementar - Controle de Versão de Documentos



Para administrar esse material de projeto, versionando automaticamente o conteúdo, vamos empregar o



Sistema de gestão de conteúdo em nuvem

Existem outras soluções, hoje menos populares que o GIT mas que funcionam bem na gestão de fontes e versões...



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Através da ferramenta de controle de versões de códigos e/ou documentos de software, é possível evitar os problemas de se trabalhar com a fonte errada em um determinado ponto do projeto, facilitando a colaboração no projeto.



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Trabalhando com GIT, os arquivos fontes serão organizados em:

Cópia Master

Contém os arquivos na versão estável, que podem ser usados por outros desenvolvedores na integração de componentes ou com outros sistemas, ou podem ser usados para gerar um pacote de versão final do produto.

Cópia Branch

Contém os arquivos em uma versão de manutenção/atualização que não estão estáveis e não podem ser usados para gerar uma versão final do produto.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

O funcionamento da Branch.

Versão Master (estável - usada para gerar versões finais do produto)

Pode retratar os fontes de uma versão já em uso/instalada para usuários

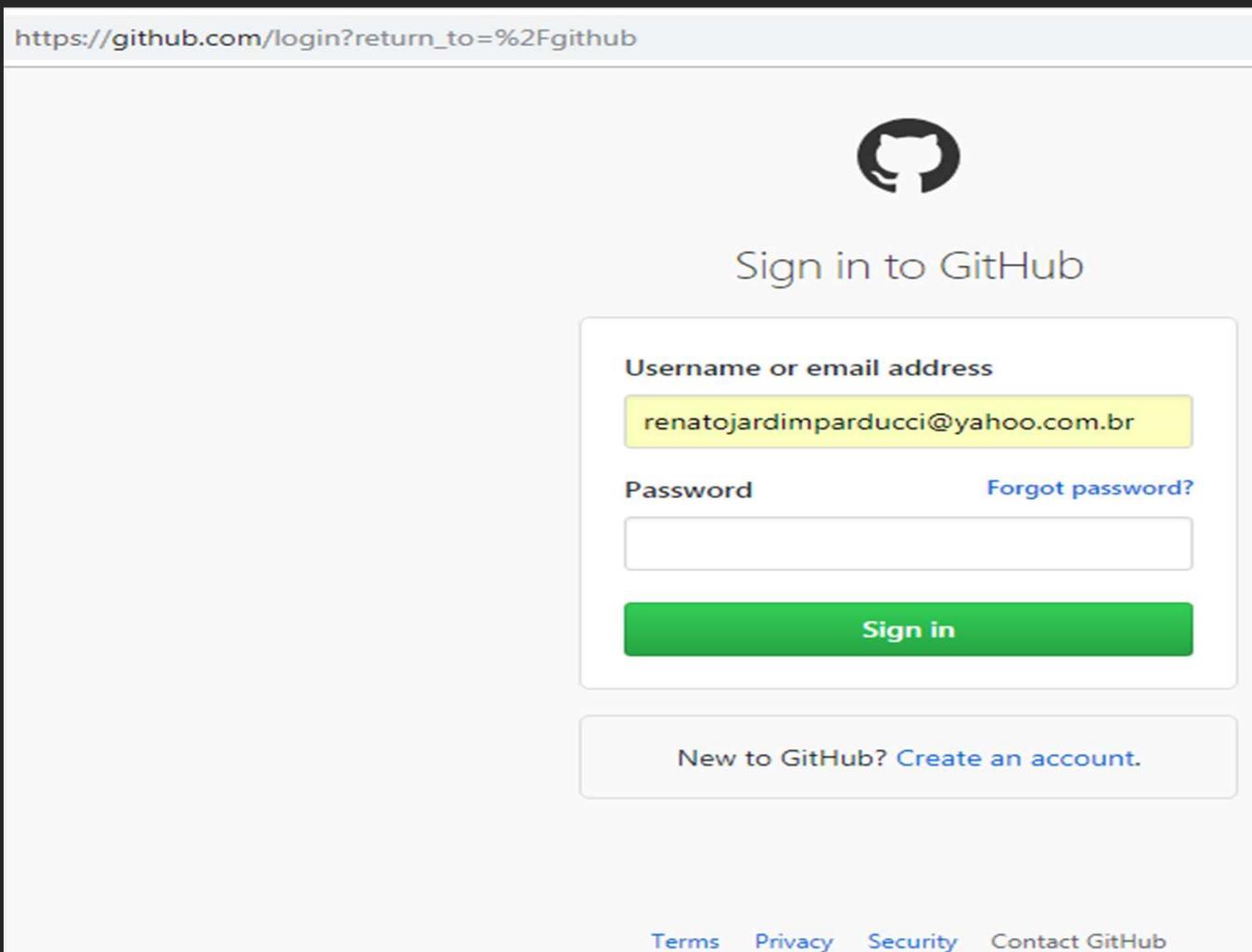


Ponto de necessidade de manutenção para:

- Corrigir o software/eliminar BUGs (manutenção CORRETIVA);
- Adaptar o software para novas regras de negócio (manutenção ADAPTATIVA);;
- Prevenir contra possíveis problemas futuros (manutenção PREVENTIVA);;
- Alcançar a perfeição na experiência do usuário (manutenção PREFECTIVA)

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Crie sua conta no GIT HUB.



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Faça o login para ver se está tudo Ok.

The screenshot shows a GitHub profile page with the following details:

- Profile Picture:** A black silhouette of a cat.
- Name:** GitHub
- Slogan:** How people build software.
- Location:** San Francisco, CA
- Website:** <https://github.com/about>
- Email:** support@github.com
- Verified:** Verified badge

Navigation Bar: Pull requests, Issues, Marketplace, Explore

Pinned repositories:

- fetch**: A window.fetch JavaScript polyfill.
JavaScript, 20.8k stars, 1.9k forks.
- hub**: hub helps you win at git.
Go, 14.1k stars, 1.4k forks.
- training-kit**: Open source on demand courses and cheat sheets for Git and GitHub.
HTML, 1.9k stars, 1.9k forks.
- choosealicense.com**: A site to provide non-judgmental guidance on choosing a license for your open source project.
Ruby, 1.4k stars, 457 forks.
- scientist**: A Ruby library for carefully refactoring critical paths.
Ruby, 4.7k stars, 200 forks.
- gh-ost**: GitHub's Online Schema Migrations for MySQL.
Go, 5.9k stars, 453 forks.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Logado no GITHUB, acesse a área de repositórios.

The screenshot shows the GitHub user profile page for RenatoJardimParducci. The top navigation bar includes links for Pull requests, Issues, Marketplace, and Explore. On the right, a dropdown menu lists options like Signed in as, Your profile, Your repositories (which is highlighted with a red box and a red arrow pointing to it), Your stars, Your gists, Help, Settings, and Sign out. Below the dropdown, the main content area displays the GitHub logo, the tagline "How people build software.", and the user's location as San Francisco, CA. It also shows contact information: https://github.com/about, support@github.com, and a Verified badge. At the bottom, there are sections for Pinned repositories, showing three projects: fetch, hub, and training-kit.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Seus repositórios de projetos de software aparecerão, caso você já tenha catalogado algum.

The screenshot shows a GitHub profile interface. At the top, there is a blue banner with the text "ProTip! Updating your profile with your name, location, and a profile picture helps other GitHub users get to know you." and a green "Edit profile" button. Below the banner, the profile navigation bar includes "Overview", "Repositories 12", "Stars 0", "Followers 0", and "Following 0". The "Repositories" tab is currently selected, indicated by an orange underline. Below the navigation bar is a search bar with the placeholder "Find a repository..." and filters for "Type: All" and "Language: All", along with a green "New" button. The main content area displays three repository cards: "1TDSJ" (updated on 26 Apr 2017), "RepExemplo" (updated on 17 Jan 2017), and "ExemploTDSS". Each card has a horizontal green progress bar underneath it.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Seus repositórios de projetos de software aparecerão, caso você já tenha catalogado algum.

The screenshot shows a GitHub profile interface. At the top, there is a blue banner with the text "ProTip! Updating your profile with your name, location, and a profile picture helps other GitHub users get to know you." and a green "Edit profile" button. Below the banner, the profile navigation bar includes "Overview", "Repositories 12", "Stars 0", "Followers 0", and "Following 0". The "Repositories" tab is currently selected, indicated by an orange underline. Below the navigation bar is a search bar with placeholder text "Find a repository..." and filters for "Type: All" and "Language: All", along with a green "New" button. The main content area displays three repository cards: "1TDSJ" (updated on 26 Apr 2017), "RepExemplo" (updated on 17 Jan 2017), and "ExemploTDSS". Each card has a horizontal green progress bar underneath it.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Crie um repositório.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner RenatoJardimParducci / Repository name TesteGITHUB ✓

Great repository names are short and memorable. Need inspiration? How about sturdy-spoon.

Description (optional)

 Public Anyone can see this repository. You choose who can commit.

 Private You choose who can see and commit to this repository.

Initialize this repository with a README This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None Add a license: None ⓘ

Create repository

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Entenda a tela do GitHub.

The screenshot shows a GitHub repository page for 'RenatoJardimParducci / TesteGITHUB'. The page includes navigation tabs for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. A description section states 'No description, website, or topics provided.' with an 'Edit' button. Below this, there's a 'Manage topics' section and a summary bar showing 1 commit, 1 branch, 0 releases, and 1 contributor. The main content area displays a commit from 'RenatoJardimParducci' titled 'Initial commit' made a minute ago. A file named 'README.md' is listed. At the bottom, the repository name 'TesteGITHUB' is shown. Three red numbered arrows point to specific UI elements:

- 1: Points to the 'Branch: master ▾' dropdown menu.
- 2: Points to the 'README.md' file entry in the commit list.
- 3: Points to the 'Upload files' button in the top right toolbar.

Below the annotations, three numbered descriptions explain the purpose of each highlighted element:

- 1 Seleciona a área/cópia de fontes para trabalho
- 2 Nomes dos arquivos que constam na área
- 3 Usado para carregar arquivos fonte no GIT

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Você pode criar pastas para separar tipos de arquivos

The screenshot shows a GitHub repository page for 'RenatoJardimParducci / TesteGITHUB'. The page includes navigation tabs for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. A red callout box with a downward arrow points to the 'Create new file' button in the main repository actions area. The callout contains the text: 'Crie um nome para a pasta/nome de um arquivo Readme que será criado'.

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided.

Manage topics

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

RenatoJardimParducci Initial commit Latest commit 7296d6c a minute ago

README.md Initial commit a minute ago

README.md

TesteGITHUB

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

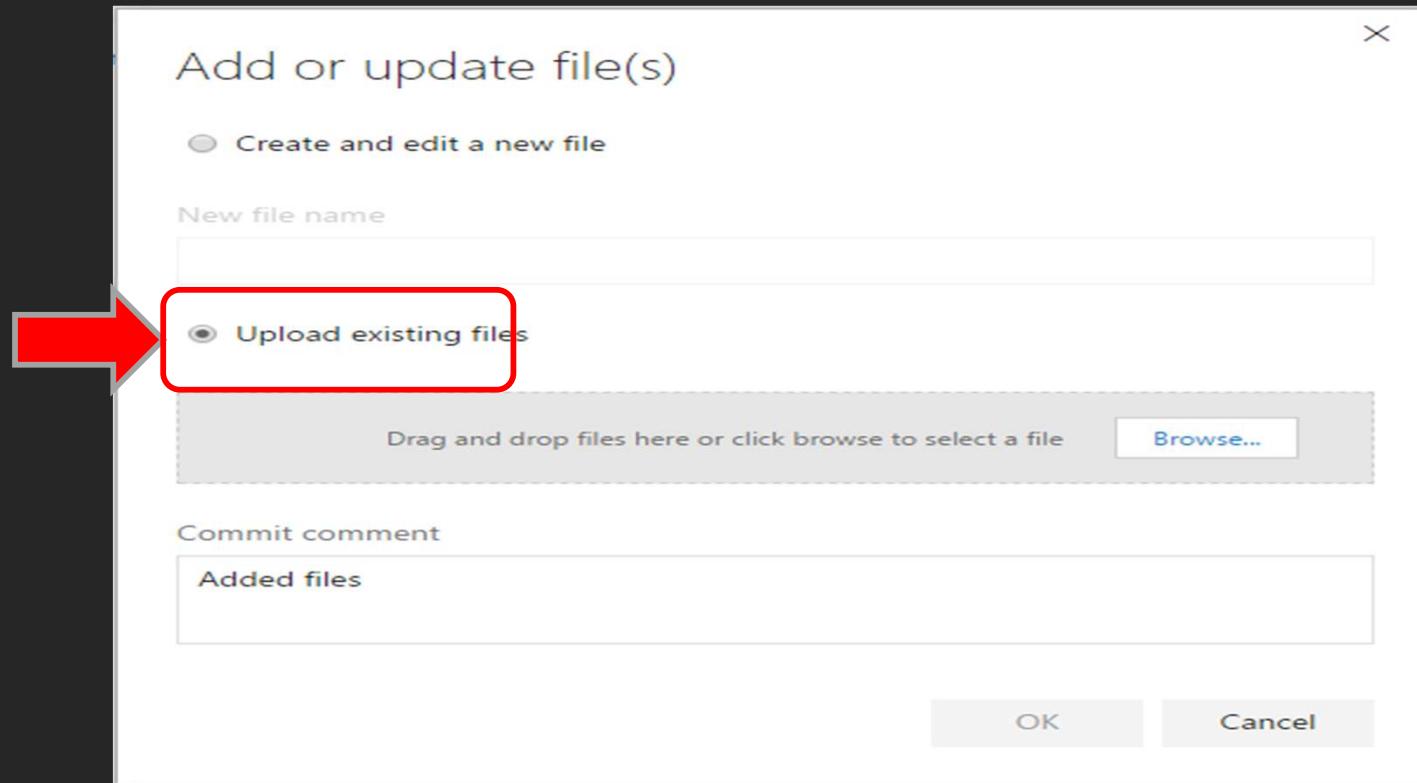
Neste ano, vamos trabalhar diretamente na **Branch MASTER**.

O GIT permite criar outras Branches que você conhecerá no próximo ano, quando estudar Database Projects & Operations!

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Acesse o link de Upload para subir para o GitHub um arquivo do seu computador.

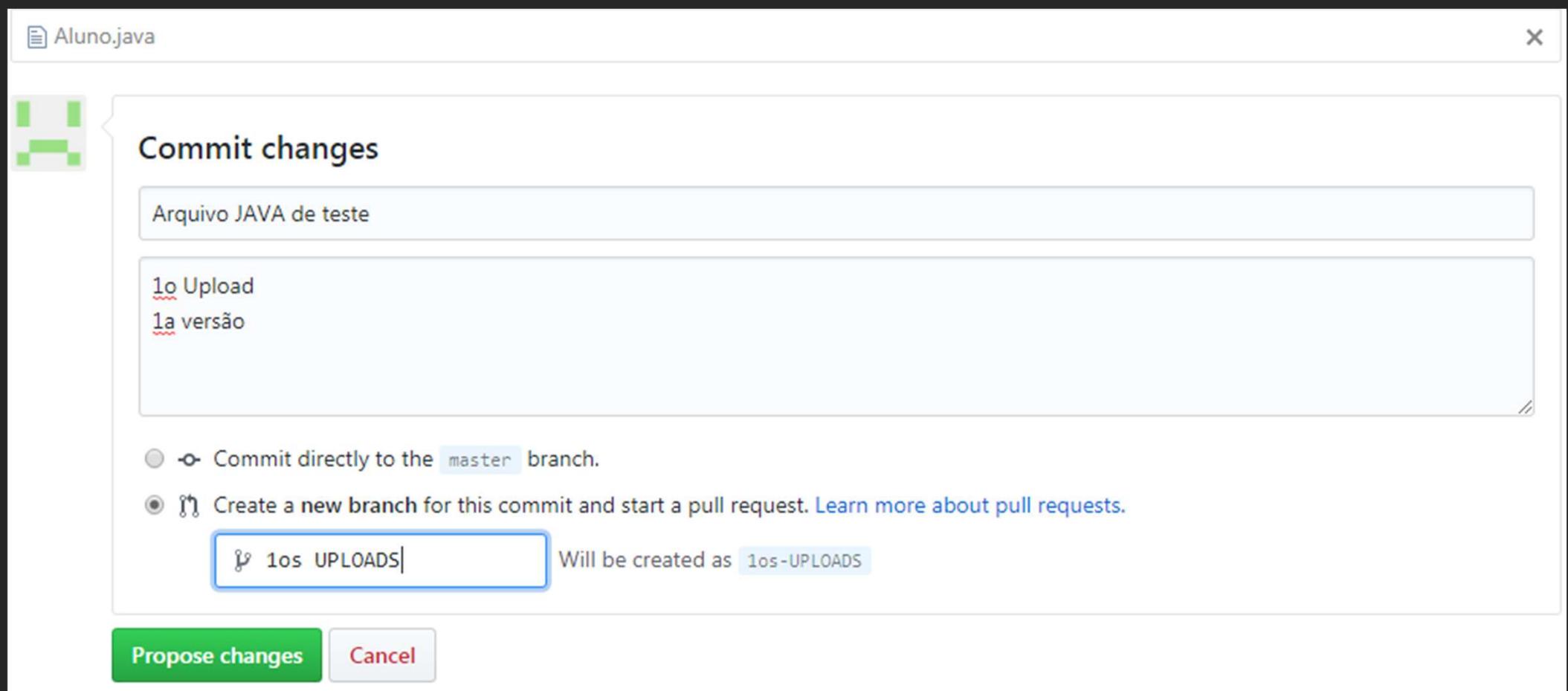
Suba um arquivo .JAVA ou .SQL para experimentar!



Como alternativa, você pode abrir a pasta com o seu arquivo no Windows Explorer e arrastá-lo para a página do GITHUB.

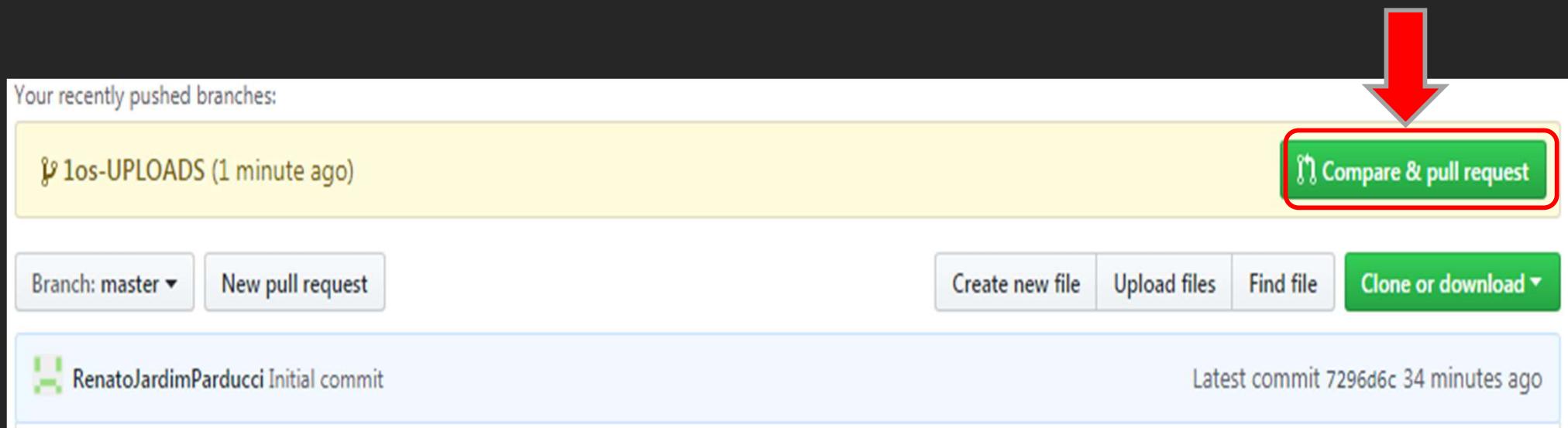
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Cada inclusão ou alteração de arquivo pode ser comentada ao ser salva, facilitando a interpretação das versões.!



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Quando algo é modificado na área de trabalho/cópia, fica habilitada a possibilidade de criar uma Pull request para atualizar a Master a partir da Branch.!



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Depois que estiver certo que a mudança está completa e correta, publique a modificação na cópia Master, tornando-a disponível para todos os desenvolvedores usarem!

The screenshot shows a GitHub interface for a repository named 'TesteUsoGitVS'. The 'Pull Requests' tab is selected. A pull request titled 'Updated CalcHoras.js' is displayed. The 'Description' section contains the note '- Updated CalcHoras.js'. Below the description, it says 'Markdown supported. Use # to mention a work item or @ to mention a person.' and lists '• Updated CalcHoras.js'. The 'Reviewers' section shows '[ProjetoExemplo-2TBA-2016]\ProjetoExemplo-2TBA-2016 Team' and a search bar. The 'Work Items' section has a search bar with the placeholder 'Search work items by ID or title'. At the bottom, there is a blue button labeled 'New pull request' with a red arrow pointing to it, and a link 'fewer options'.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

A Master terá o histórico de todas as modificações feitas. Basta acessar o nome do arquivo e clicar no histórico.

The screenshot shows a GitHub repository interface. At the top, there are navigation links: Code, Issues 0, Pull requests 0, Projects 0, Wiki, Insights, and Settings. Below this, the branch is set to master, and the file path is TesteGITHUB / Aluno.java. There are buttons for Find file and Copy path. The main content area displays the file's history. A commit by RenatoJardimParducci is shown, with the commit message "Update Aluno.java" and the hash e92ad94 from 9 minutes ago. A red arrow points to the History button in the toolbar below the commit list. The toolbar also includes Raw, Blame, History (which is highlighted with a red box), and other icons. The code listing shows the following Java code:

```
1  public class Aluno1 extends Pessoa {  
2  
3      private Matricula matricula;  
4  
5      public void SolicitarMatricula() {  
6          }  
7      }  
8  
9  }
```

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

A Master terá o histórico de todas as modificações feitas.

The screenshot shows a GitHub repository page for 'RenatoJardimParducci / TesteGITHUB'. The repository has 1 unwatched star, 0 forks, and 0 issues/pull requests/projects/wiki/insights/settings. The current branch is 'master'. The commit history for November 30, 2018, is displayed:

- Merge pull request #1 from RenatoJardimParducci/1os-UPLOADS ...
RenatoJardimParducci committed 10 minutes ago. Verified, ba7d3c0, copy link.
- Update Aluno.java
RenatoJardimParducci committed 14 minutes ago. Verified, e92ad94, copy link.
- Arquivo JAVA de teste ...
RenatoJardimParducci committed 23 minutes ago. Verified, 55ecc9c, copy link.
- Initial commit
RenatoJardimParducci committed an hour ago. Verified, 7296d6c, copy link.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Caso você precise recuperar uma versão anterior, basta selecioná-la.

The screenshot shows a GitHub repository page for 'RenatoJardimParducci / TesteGITHUB'. The 'Code' tab is selected. A dropdown menu shows the branch is 'master'. Below the dropdown, it says 'Commits on Nov 30, 2018'. There are four commits listed:

- Merge pull request #1 from RenatoJardimParducci/1os-UPLOADS ...
RenatoJardimParducci committed 9 minutes ago
- Update Aluno.java
RenatoJardimParducci committed 12 minutes ago
- Arquivo JAVA de teste ...
RenatoJardimParducci committed 21 minutes ago
- Initial commit
RenatoJardimParducci committed 44 minutes ago

Each commit has a 'Verified' button, a copy icon, a commit hash (ba7d3c0, e92ad94, 55ecc9c, 7296d6c), and a 'diff' icon. A tooltip 'Browse the repository at this point in the history' points to the commit list. At the bottom, there are 'Newer' and 'Older' buttons, and a link to the repository's tree: <https://github.com/RenatoJardimParducci/TesteGITHUB/tree/55ecc9c3ecf99d5fc8bf61652825d9ca3a2da336>. The taskbar at the bottom shows icons for SourceTreeSetup, Git, and a browser, along with system status icons.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Quando selecionada uma versão do fonte, o GIT mostra o que foi alterado para você ter certeza de qual versão está vendo.

The screenshot shows a GitHub commit history for the file `Aluno.java`. The commit was made by RenatoJardimParducci 12 minutes ago and is verified. It has one parent commit `55ecc9c` and a commit hash `e92ad942ca6b8d544ad4282d95dfcde8d1121567`. The commit message is not visible. The interface shows 1 changed file with 1 addition and 1 deletion. The diff view highlights the change where the class name was updated from `Aluno` to `Aluno1`.

```
diff --git a/Aluno.java b/Aluno.java
@@ -1,4 +1,4 @@
- public class Aluno extends Pessoa {
+ public class Aluno1 extends Pessoa {
```

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Você acabou de estudar um processo com atividades, definição de responsabilidades e ferramentas para gerenciar fontes de programas de aplicação em suas versões.

Seu 1º passo para garantir Qualidade em projetos de software e a Governança, através da garantia da continuidade dos negócios.

ASSISTA OS VÍDEOS NO CANAL DO PROFESSOR



<https://youtu.be/MYhIM0bk9aQ>

A screenshot of a video player interface. The video content shows a man speaking about GitHub. The video player has a progress bar at 0:30 / 13:49. On the left, there is a sidebar with a "Discover interesting projects and people" section and a "Visualize your project's community" notification. The URL in the browser bar is github.com. The video player has standard controls like play/pause, volume, and full-screen.



Conteúdo didático complementar - GitHub

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Existem algumas outras formas de você acessar e operar o conteúdo do Git Hub a partir do seu PC:

- GIT Gui (por menus em um software cliente);
- GIT CMD (por linha de comando).

Não exploraremos essas modalidades nas aulas de Data Governance.
Vamos focar em compreender e usar bem o GIT HUB.

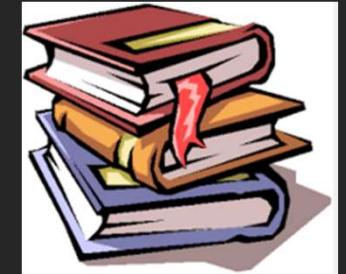


D Ú V I D A S

Referência bibliográficas

BIBLIOGRAFIA:

- PRESSMAN, R. S. **Engenharia de software**. São Paulo: Editora McGraw-Hill, 2002.
- SOMERVILLE, I.. **Engenharia de software**. São Paulo: Editora Pearson, 2010.



BONS ESTUDOS!