

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

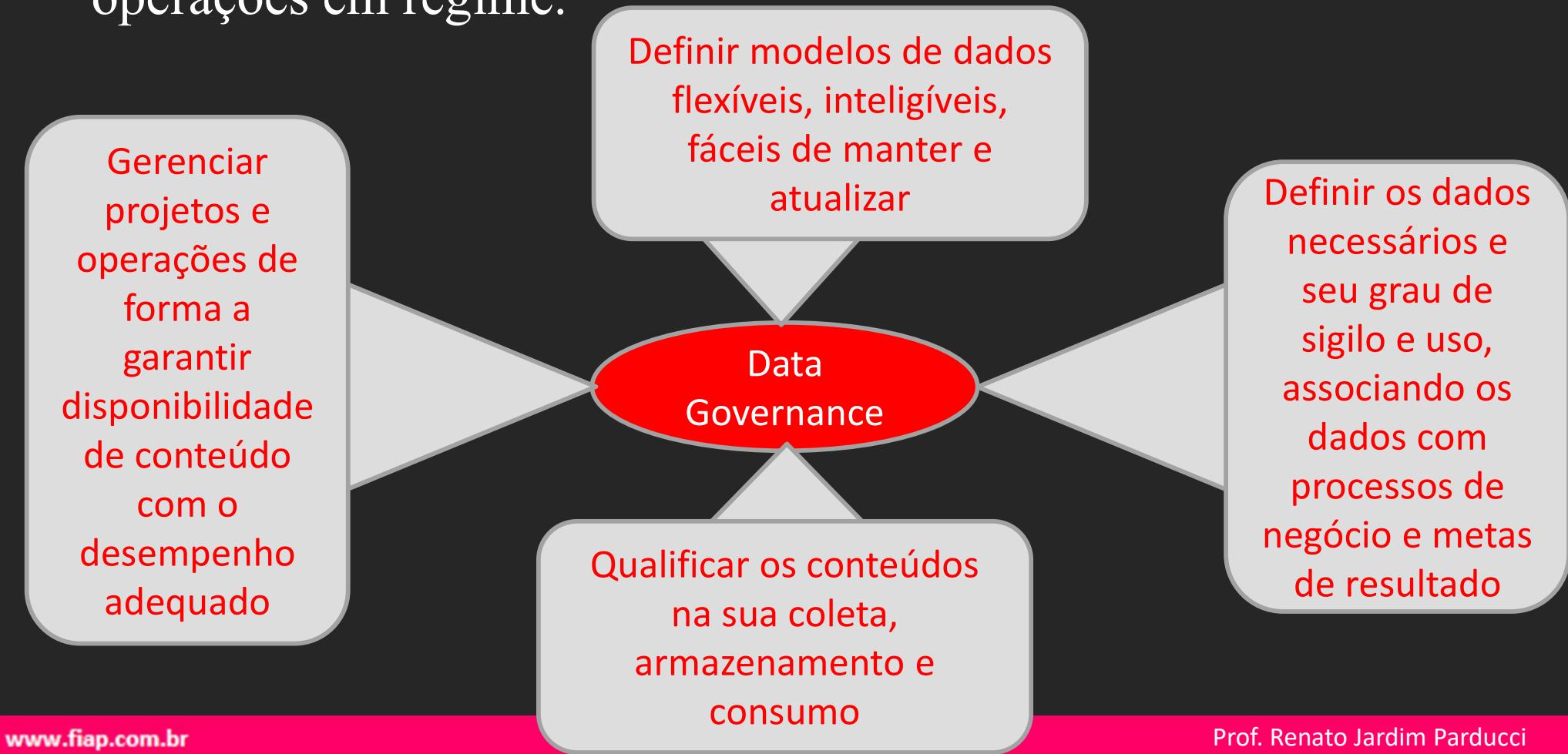
AGENDA DA AULA

- ✓ Buscas da Governança de Dados
- ✓ Ciclo de Vida de um Banco de Dados
- ✓ Ciclo de Vida de Projetos de Bancos de Dados

Busca da Governança de Dados

Afinal, ... O que quer a Governança de Dados?

- Entregar o conteúdo certo (completo, correto e válido), na hora certa, para a pessoa certa avaliar um cenário e tomar uma decisão que influencie positivamente o negócio ou mantenha suas operações em regime.



Os **primeiros passos** na Governança de Dados passam por compreender o **ciclo de projetos de bancos de dados e mecanismos de gerenciamento** desses projetos!

Vale considerar que em Data Governance, não existem projetos “de banco de dados” mas sim, projetos de aplicações feitas em software que usam bancos de dados.

O que se quer é um sistema que processe a folha de pagamento de uma empresa com o devido sigilo, desempenho e qualidade, sendo que esse sistema de aplicação precisa de bases de dados com dados cadastrais de funcionários, informações de cargos, salários e horas trabalhadas em um período, por exemplo!

Para prover as soluções de data e analytics, são necessários:

Dados originados de diversas fontes de interesse

Dados internos da empresa em diversos formatos

Dados externos à empresa em diversos formatos

Profissionais que desenvolvem o tratamento dos dados

Analistas de dados

Arquitetos de solução

Engenheiros de dados e sistemas

Estrategistas na aplicação de TI aos negócios

Usuários e interessados

Sistemas de armazenamento e distribuição em massa dos dados

Storage

Servidores

Redes

Serviços em núvem

Configuração escalável

Sistemas de processamento dos dados em informações e conhecimentos

Qualificação de dados

Ingestão de dados

Transformação de dados

Geração de informação e inteligência

Sistemas de auxílio ao consumo de dados e informações

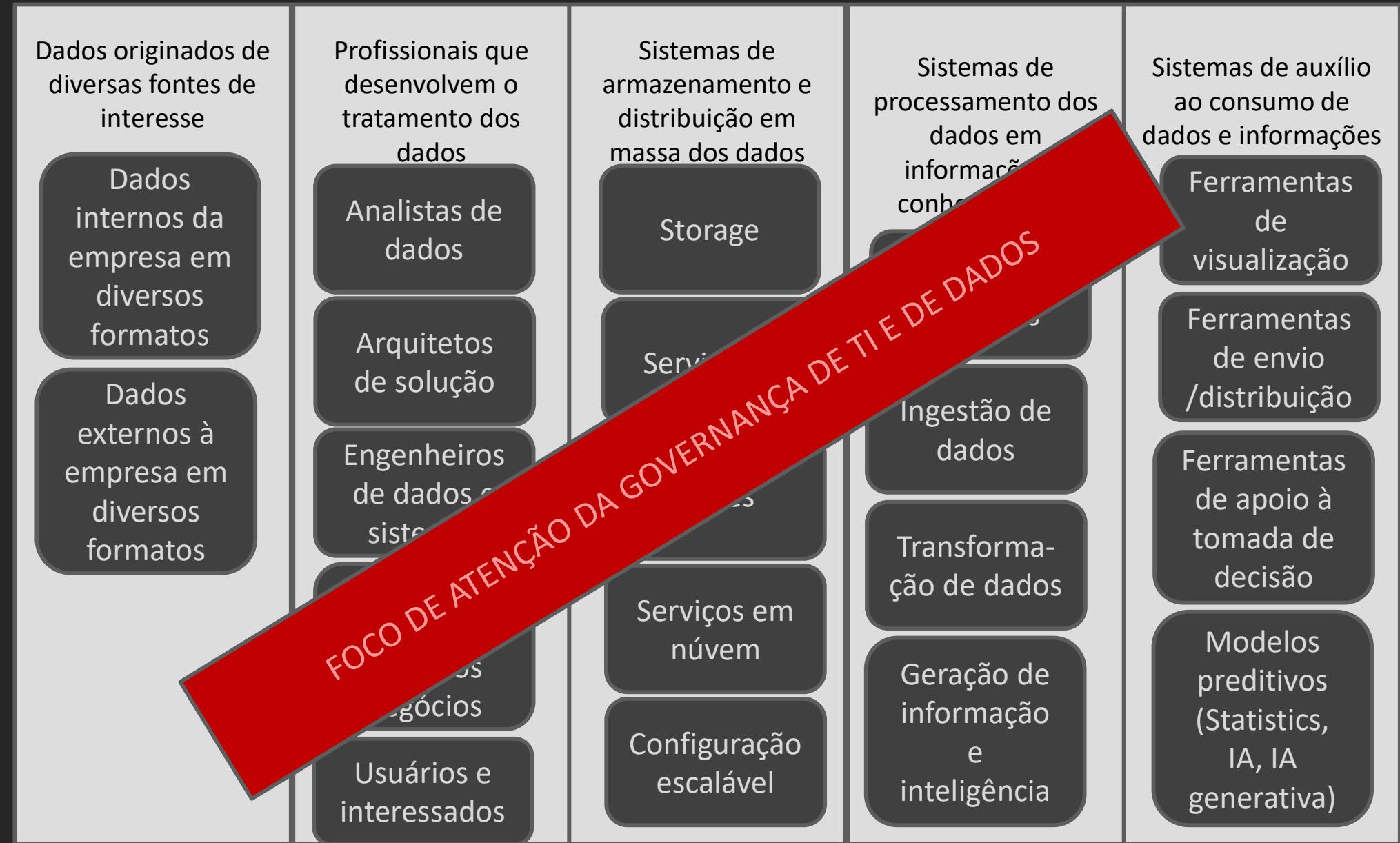
Ferramentas de visualização

Ferramentas de envio /distribuição

Ferramentas de apoio à tomada de decisão

Modelos preditivos (Statistics, IA, IA generativa)

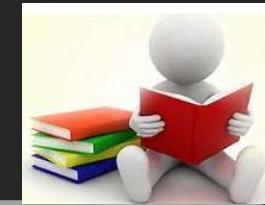
Para prover as soluções de data e analytics, são necessários:



VAMOS COMEÇAR A JORNADA!

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.

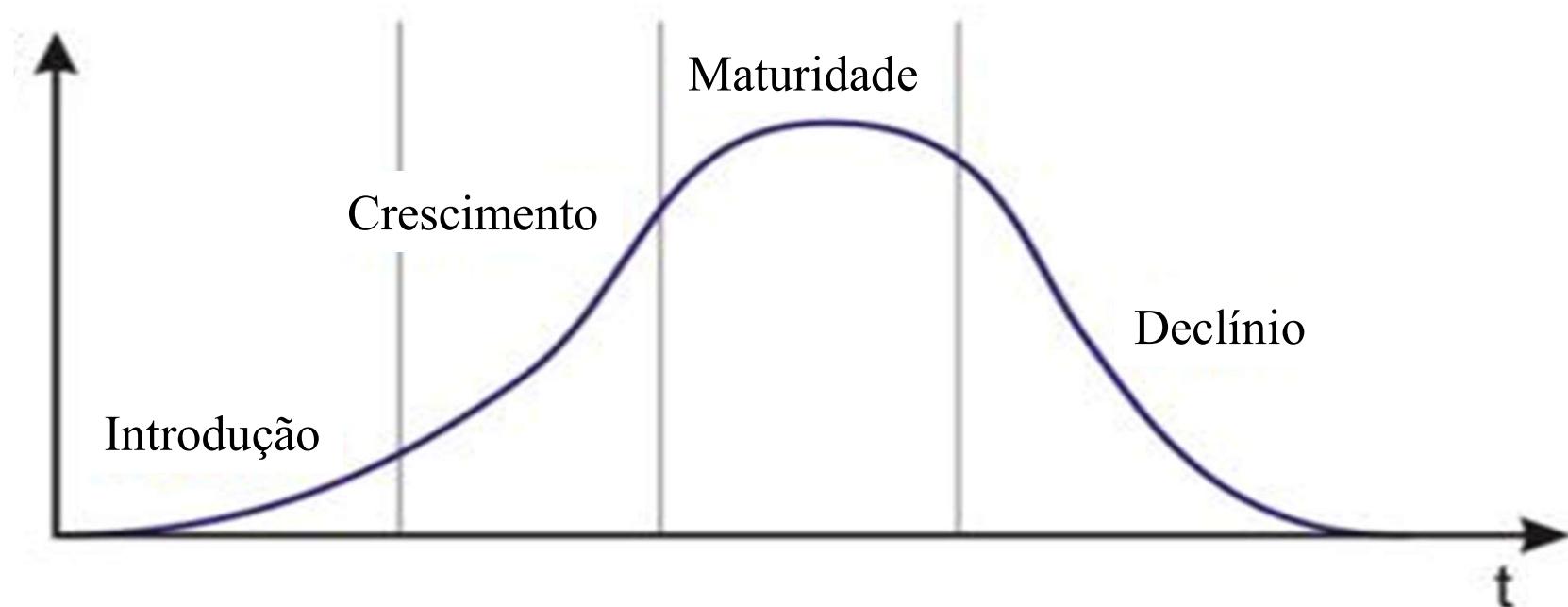
Você, da SuperSoluções precisa explicar se essa expectativa será atendida e justificar por que sim ou por que não.

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

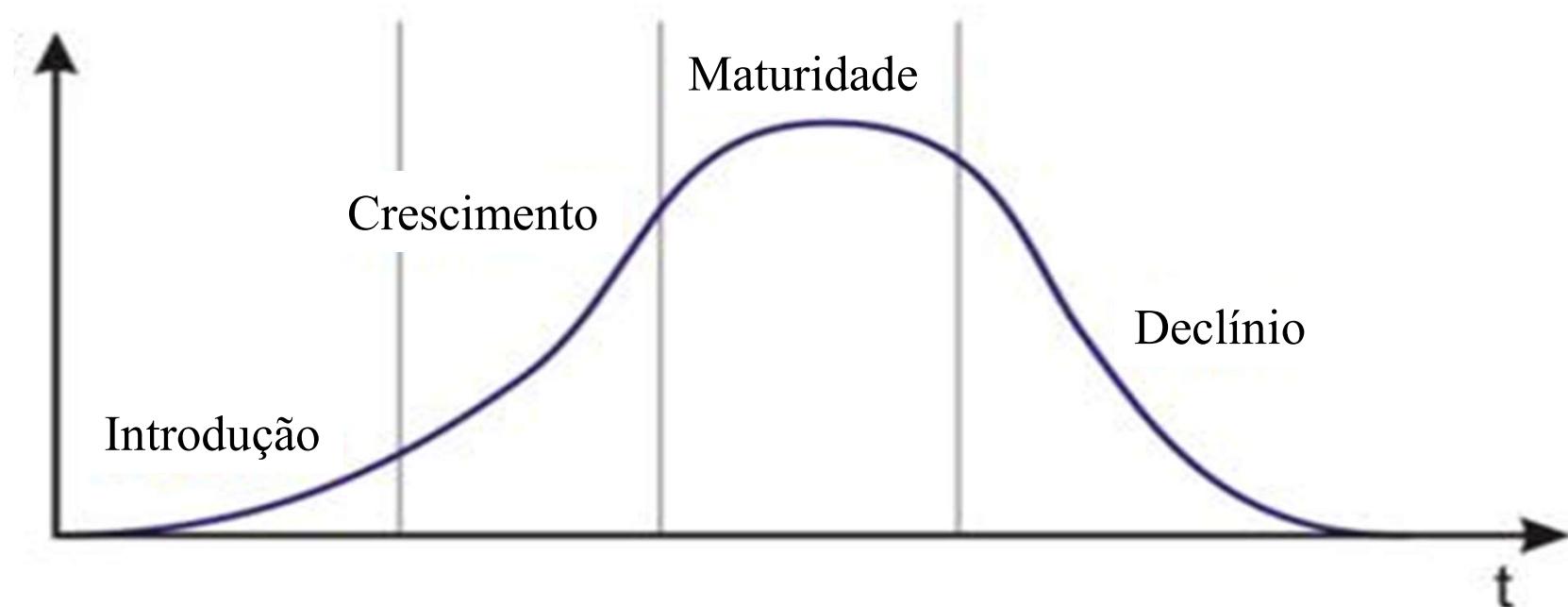
Todo produto tem uma CURVA DE VIDA

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!

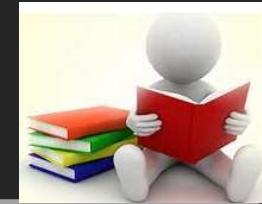


Manutenções podem ser do tipo:

- **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.

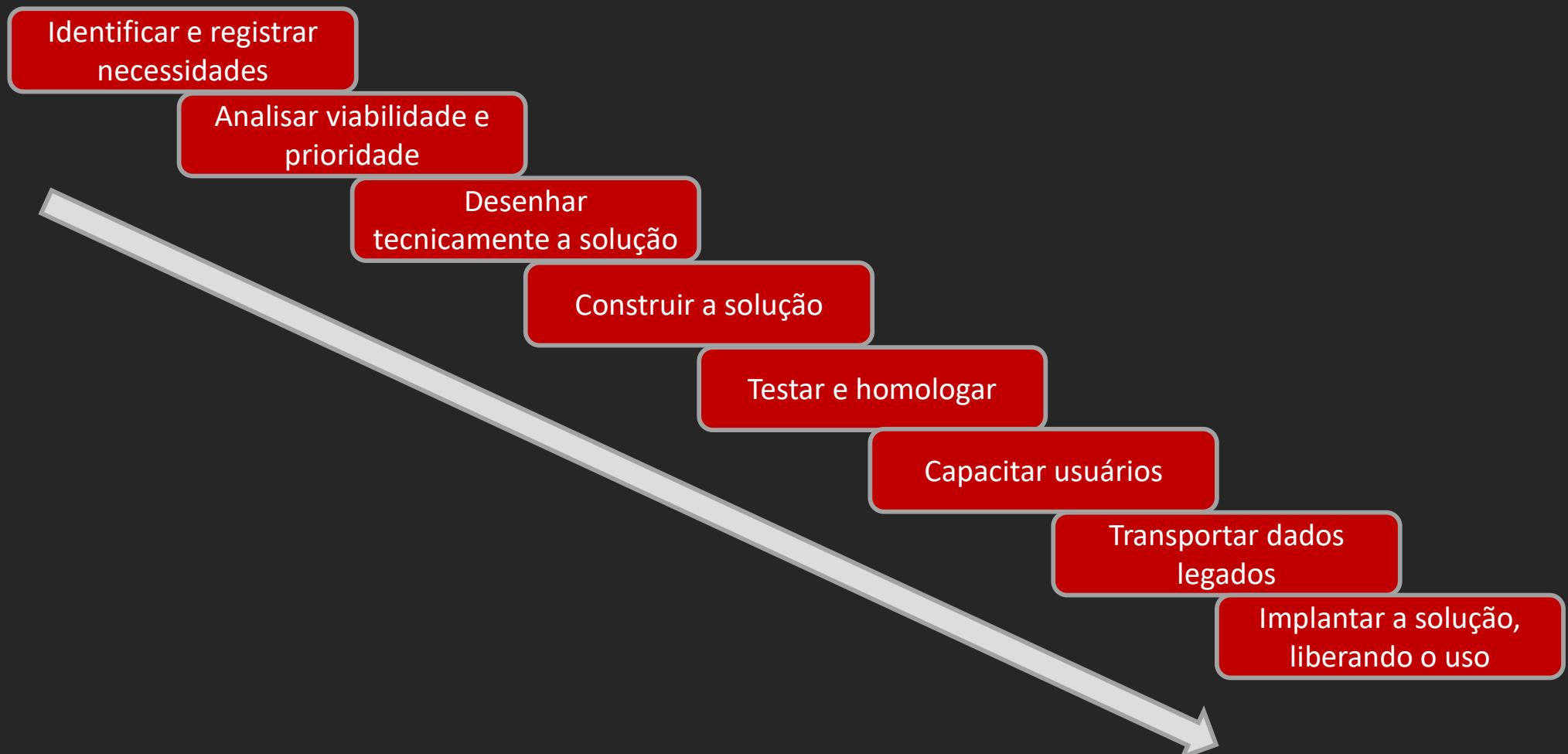
Ciclos de Vida são formas de pensar!
São paradigmas!

Na Engenharia de Software que inclui a Engenharia de Bancos de Dados, define um conjunto de passos executados em uma sequência ideal para desenhar, construir e entregar a solução tecnológica necessária.

Definem etapas 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

ETAPAS DE UM PROJETO DE BANCO



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

Identificar e registrar necessidades

Analisar viabilidade e prioridade

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,

Avaliar os impactos para o negócio e desafios técnicos de cada requisito identificado

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

Aproveitamento de dados de sistemas já existentes na empresa

Desenhar tecnicamente a solução

Construir a solução

Testar e homologar

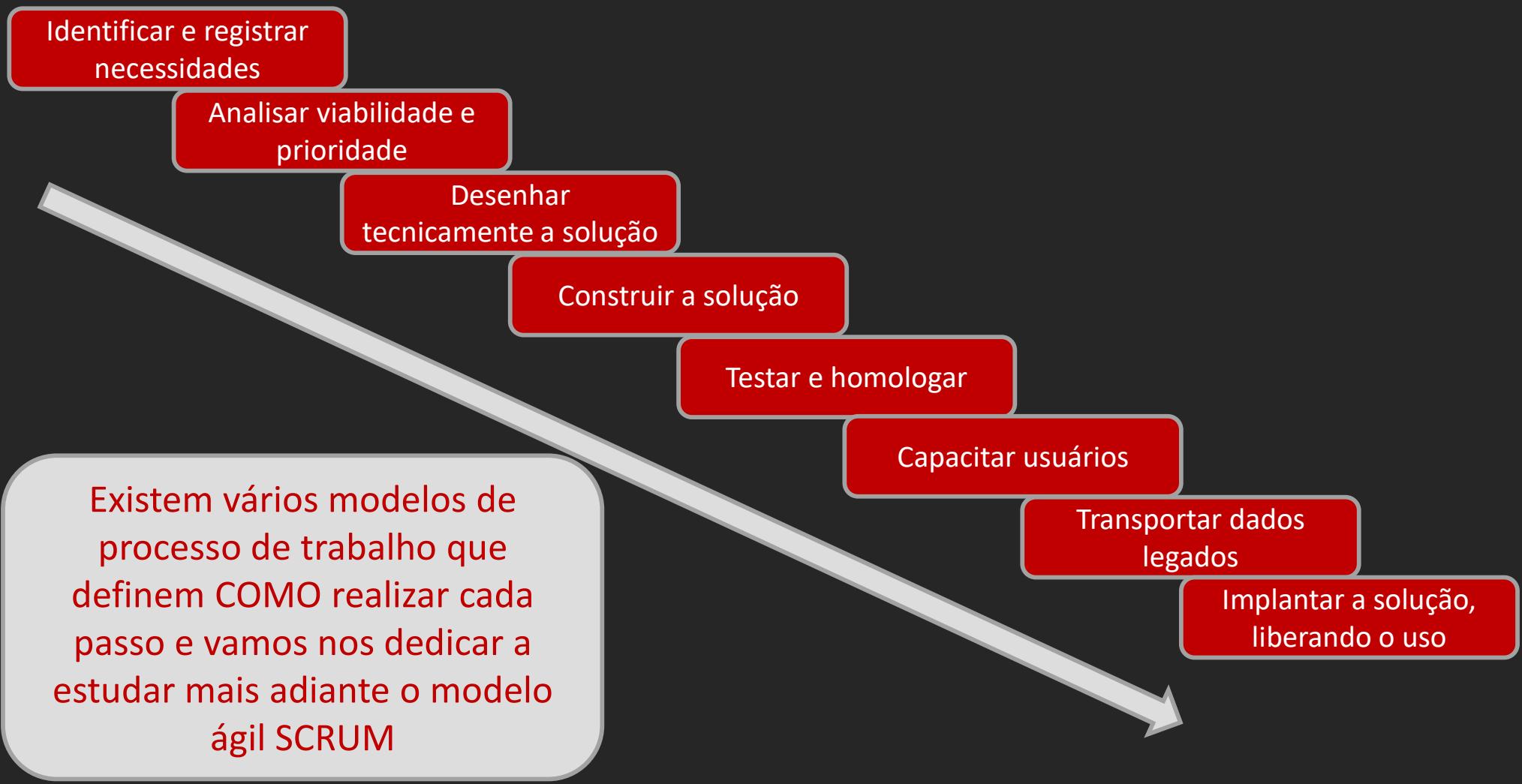
Capacitar usuários

Transportar dados legados

Implantar a solução, liberando o uso

CICLOS DE VIDA DE SOFTWARE

ETAPAS DE UM PROJETO DE BANCO



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?

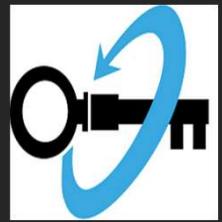
TURN KEY (VIRADA DE CHAVE)

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

Após entregar uma solução para uso, inicia-se o período de manutenção dessa solução.

Implantar a solução,
liberando o uso

Manutenção da
solução liberada

Fazer cópias de segurança (backup) dos dados, limpar dados desnecessários, reorganizar as estruturas do banco de dados, acompanhar o uso/consumo de recursos, ajustar parâmetros de desempenho do banco de dados, atualizar o sistema gerenciador do banco, exportar e importar dados sob demanda.

Na Engenharia de Software (que engloba bancos de dados), 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 Evolucionária**

-Modelo de processo **Espiral**

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**

Modelos preditivos, que se baseiam em um escopo bem determinado e procuram eliminar ou minimizar mudanças no curso do projeto

-Modelo de processo **Incremental**

-Modelo de **Protot.Evolucionária**

-Modelo de processo **Espiral**

Modelos adaptativos, que se baseiam no aprimoramento de conhecimentos e da compreensão de necessidades conforme o projeto evolui, sendo abertos à aceitação de mudanças

Abordagem híbrida: durante um projeto ou programa que engloba vários projetos dentro de um mesmo assunto/objetivo de negócio, adota-se dinâmica preditiva em alguns momentos ou parte das entregas e modelo adaptativo em outras.

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**

Criam “ondas” de entregas de projetos e suas complementações, procurando assim, eliminar a acomodação de mudanças que vejam impactar planos

-Modelo de processo **Incremental**

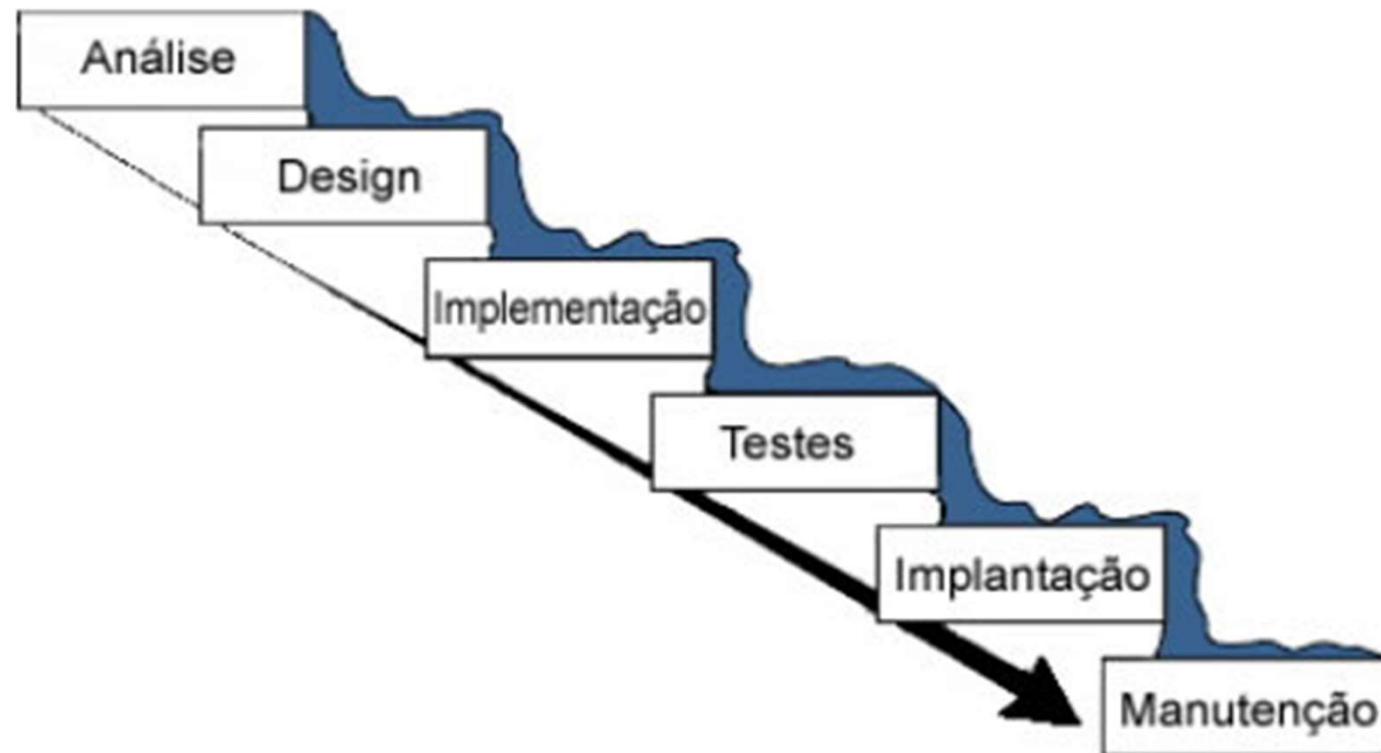
-Modelo de **Prototipação**

Trabalham com ciclos curtos de entendimentos de demandas e entregas, ajustando expectativas e planos sempre que necessário

-Modelo de processo **Espiral**

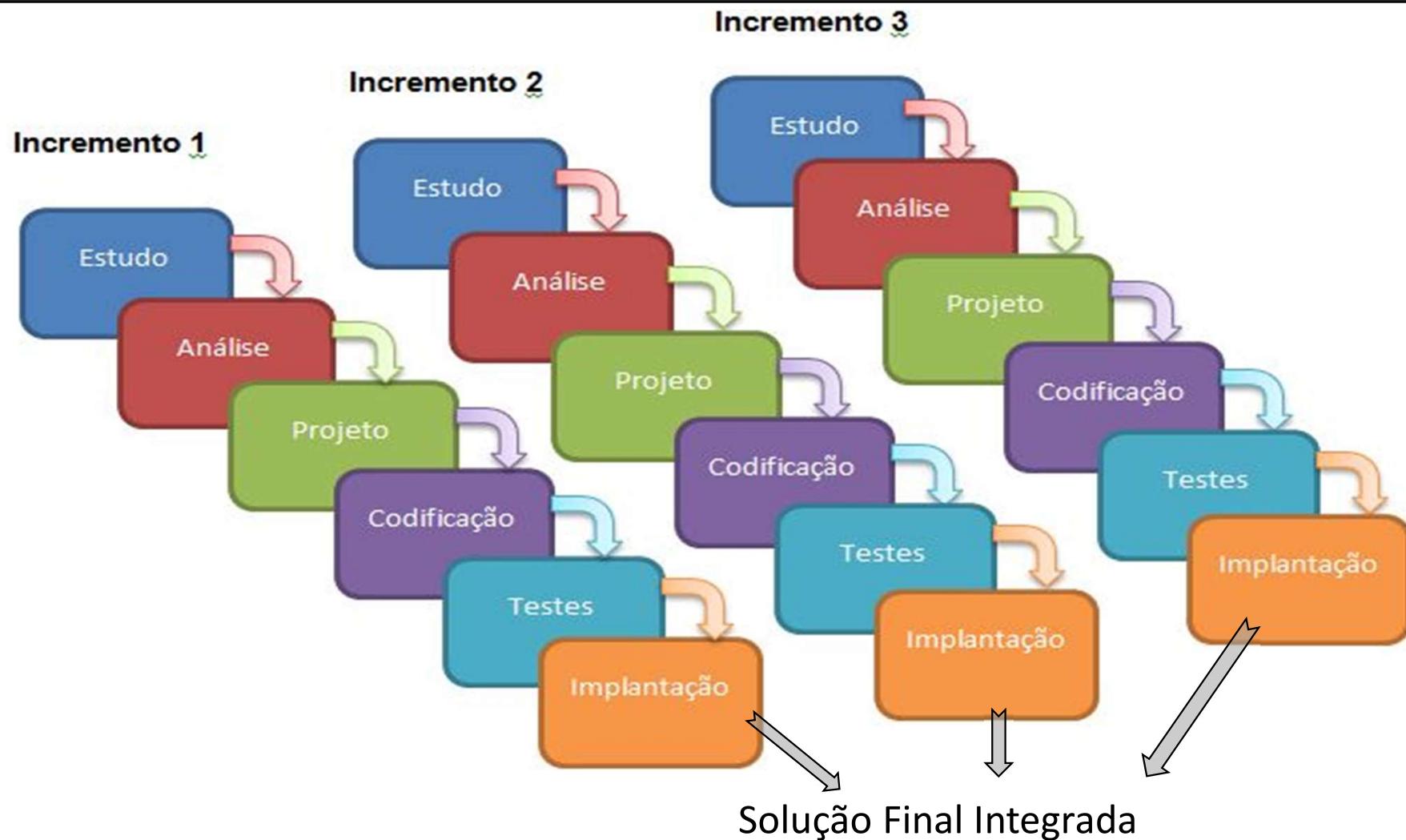
CICLOS DE VIDA DE SOFTWARE

MODELO CASCATA



CICLOS DE VIDA DE SOFTWARE

MODELO INCREMENTAL



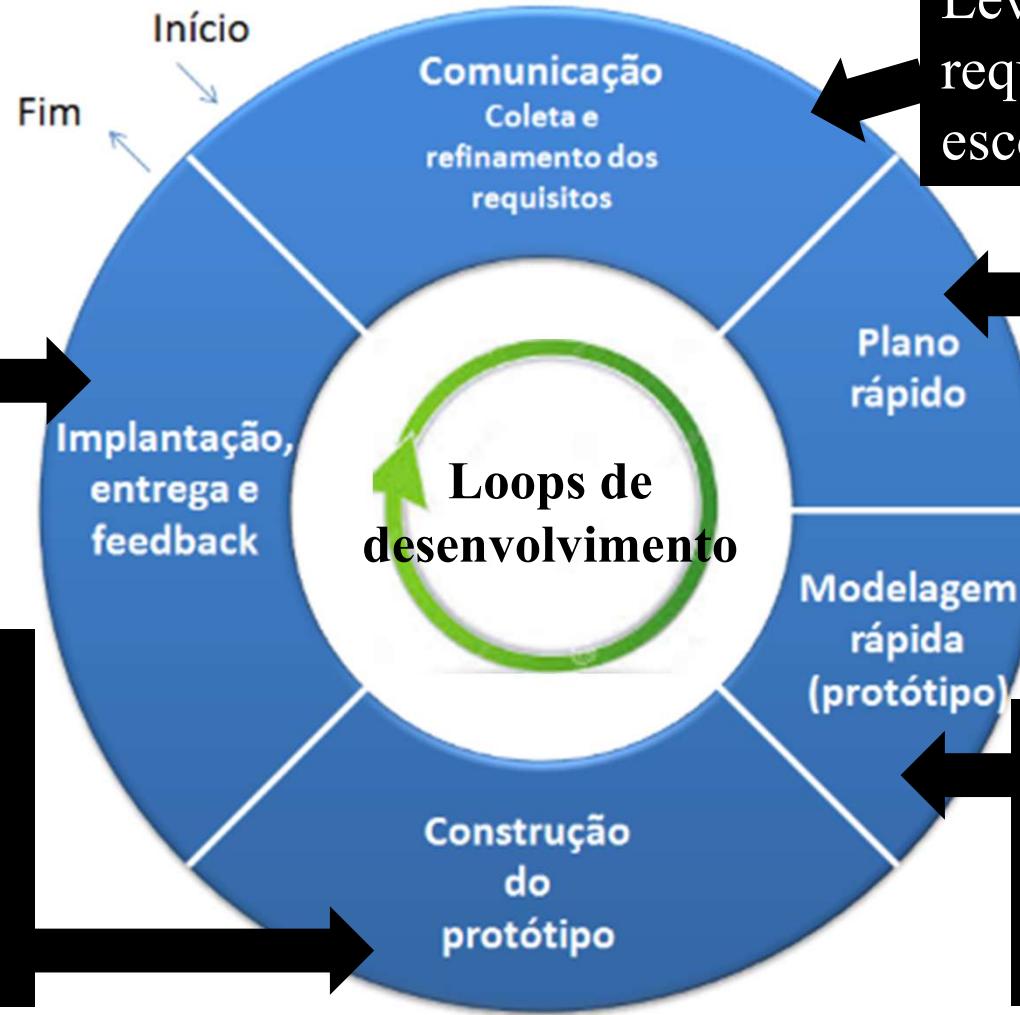
CICLOS DE VIDA DE SOFTWARE

Os ciclos de desenvolvimento acontecem até que o produto esteja completo

MODELO PROTOTIPAÇÃO EVOLUCIONÁRIA

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 SCRUM (abordado adiante) explica esse processo em detalhes



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

QUAL DESSAS LINHAS DE TRABALHO TRAZ MELHOR RESULTADO?



-Modelo de processo **Cascata**

?

-Modelo de processo **Incremental**

?

-Modelo de **Prototipação**

-Modelo de processo **Espiral**

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).

CICLOS DE VIDA DE SOFTWARE

PRÁTICA



Vamos usar o MS-Project para aprender a criar cronogramas de projetos:

Cascata

Incremental

Evolucionário (Prototipação)

Espiral

O PMI (Project Management Institute) é a maior referência mundial em práticas de gerenciamento de projetos e estabelece domínios de conhecimento e princípios de conduta, no seu livro PMBoK (Project Management Body of Knowledge).

Guia PMBOK® – Sétima Edição

Padrão de Gerenciamento de Projetos:

- Introdução
 - Sistema de entrega de valor
 - Princípios de gerenciamento de projetos
 - Intendência
 - Equipe
 - Partes interessadas
 - Valor
 - Visão sistêmica
 - Liderança
 - Tailoring
 - Qualidade
 - Complexidade
 - Risco
 - Adaptabilidade e resiliência
 - Change

Guia do Conhecimento em Gerenciamento de Projetos:

- Domínios de desempenho de projetos:
 - Partes interessadas
 - Equipe
 - Abordagem de desenvolvimento e ciclo de vida
 - Planejamento
 - Trabalho do projeto
 - Entrega
 - Medição
 - Incerteza
 - *Tailoring*
 - Modelos, métodos e artefatos

O guia do PMI não estabelece quais técnicas e ferramentas aplicar; ele apenas determina o que precisa ser pensado e providenciado, cabendo a cada empresa, escolher seus métodos e ferramentas para realizar projetos.



Garanta, durante o projeto:

- 1-Gerenciamento de **integração**: *fundamental nos atuais projetos que têm grandes equipes de desenvolvimento, produzindo partes de uma solução completa. Essa área do conhecimento pede que sejam planejados e controlados os pontos de intersecção de trabalhos.*
- 2-Gerenciamento do **escopo**: *focado na definição das entregas (características do produto) e tarefas a serem realizadas.*
- 3-Gerenciamento do tempo: *estabelecer uma proposta de ciclos de entregas que constituem um cronograma ou plano de entregas.*
- 4-Gerenciamento do custo: *estimar o valor a ser investido no desenvolvimento do projeto e gerenciar os gastos, mantendo foco no orçamento negociado no projeto. O custo dependerá do esforço ou seja, da quantidade de pessoas colocadas para trabalhar em uma janela de tempo.*

5-Gerenciamento de **recursos**: *recursos são pessoas e infraestrutura (equipamentos, softwares de apoio, por exemplo) que precisam ser previstos e controlados na sua aplicação ao longo do projeto.*

6-Gerenciamento de aquisição/fornecimentos: *foca em planejar compras a serem feitas para suprir as necessidades de equipamentos, softwares e mão-de-obra terceirizada para o projeto..*

7-Gerenciamento de **comunicação**: *objetiva manter o alinhamento entre membros da equipe de projeto e todas as frentes/times de trabalho. Também objetiva manter o alinhamento com stakeholders (partes interessadas no projeto), notificando sobre planos e realizações.*

8-Gerenciamento de **partes interessadas**: *foco em registrar quem são as pessoas interessadas no projeto (tanto da área usuária quanto técnicos) e suas expectativas, identificando os afetados pela solução que será desenvolvida, os influenciadores das decisões de projeto e os financiadores.*

9-Gerenciamento do risco: *identificar quais são os fatores que geram risco ao projeto (risco de perder recursos, indefinição de escopo, por exemplo), estimar o grau de risco (probabilidade de ocorrência e impacto no prazo, custo e escopo, por exemplo) e planejar e executar ações de mitigação (eliminar o fator de risco, transferir o risco, criar alternativas de contingenciamento a serem acionadas se surgir problema).*

10-Gerenciamento de qualidade: *define indicadores de sucesso (métricas de desempenho) do projeto que devem incluir medições tangíveis de resultados de custos, cumprimento de metas de prazos para entregas, escopo atendido, distribuição de carga de trabalho entre as pessoas da equipe de projeto, cobertura de requisitos entregues versus solicitados, percentual de valor agregado das entregas, produtividade por desenvolvedor, entre outras métricas da escolha de cada empresa.*

São considerados como principais indicadores do PMBoK:

- Valor agregado (VA)
- Índice de Desempenho de Prazo (IDP)
- Índice de Desempenho de Custo (IDC)
- Taxa de tarefas realizadas
- Desvios de esforço

O PMBoK determina como **TRIPLA RESTRIÇÃO** de projeto as determinações de **escopo**, **tempo**, **custo** de projeto são os maiores desafios do planejamento e controle, sendo também fatores que afetam todos os demais aspectos da gestão:

- Se mudar escopo, isso afeta custo, risco, qualidade, integração, custo, prazo;
- Se mudar prazo, isso reduz a disponibilidade de agendamento de recursos, podendo implicar em custos adicionais para dar conta do trabalho, gerando novos riscos, modificando controles de qualidade e criando novas situações de comunicação e integração;
- Se mudar o orçamento (custo) do projeto, isso pode reduzir ou ampliar recursos que impactarão prazos, podendo gerar novos pontos de integração, comunicação e controle de qualidade.

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

A escolha de aplicar um modelo Clássico ou Ágil é da equipe de desenvolvimento, conforme as características do projeto.

As ferramentas de gerenciamento serão escolhidas em vista do modelo de ciclo de vida de projeto e processo escolhido.

PRODUÇÃO DE SOFTWARE

O software é dinâmico!

Tecnologias disponíveis mudam rapidamente e suas aplicações aos negócios igualmente.

Travar o escopo de um projeto que demora anos para a sua conclusão é algo impensável nos dias atuais.

Modelos flexíveis de gerenciamento de projetos são mais adequados para a produção de software e hoje trabalha com princípios de desenvolvimento otimizado e acelerado (DEVOPS), previstos em frameworks (estruturas de trabalho como o



SCRUM e FLEKS

O modelo SCRUM é um dos mais difundidos para gerenciamento dentro do modelo Adaptativo, e o mais aplicado na dinâmica indústria de software.

O Scrum se fundamenta em ciclo de iteração chamados Sprints, que contém cerimônias de planejamento, organização e controle que possibilitam flexibilizar as entregas e tarefas do projeto.



DETALHES DO MÉTODO ÁGIL

O SCRUM explica como realizar os passos do projeto de banco de dados e traz um conceito forte de trabalho em TIME para concretizar resultados rápidos, tendo clara definição de papéis e responsabilidades!

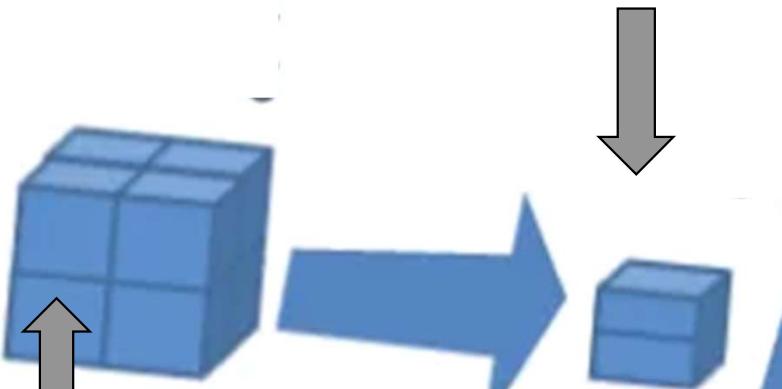
Ele é organizado em um processo definido em TIME BOXES, que são caixas de tempo pré-fixado, mantendo assim um ritmo de produção, tendo o compromisso de gerar frequentemente uma entrega útil para o cliente.

Ele inclui ainda, um conjunto de ARTEFATOS de gestão, que nada mais são que documentações gerenciais padronizadas a serem produzidas ao longo do ciclo de processo produtivo.

CICLO DE VIDA E O PROCESSO ÁGIL DE PRODUÇÃO DE SOFTWARE

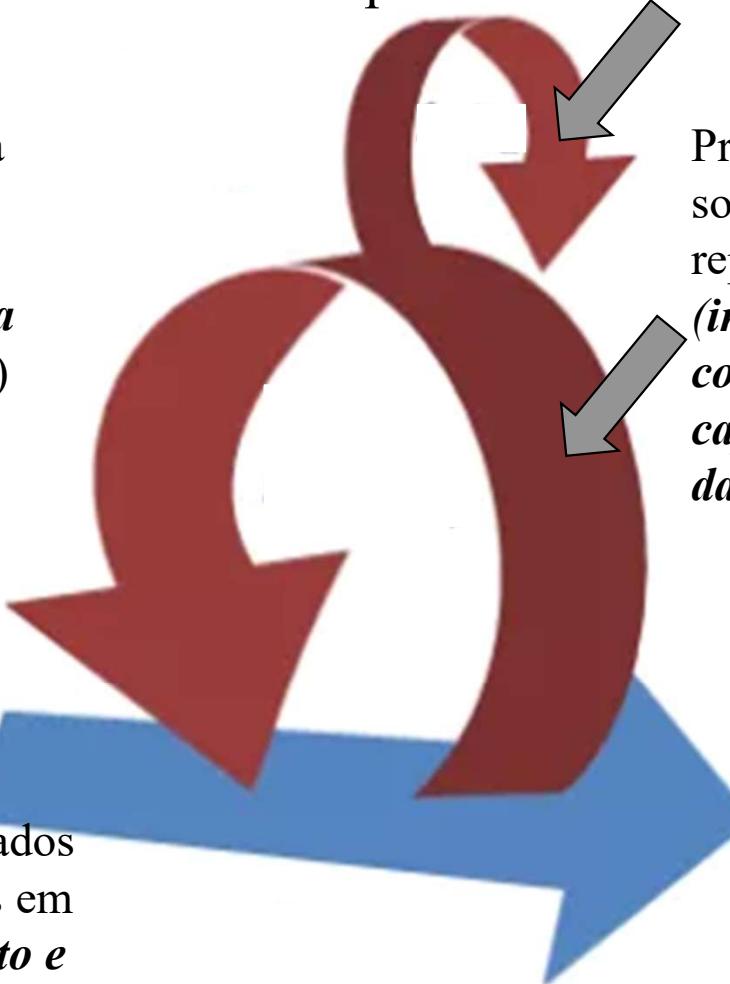
SCRUM

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



Lista e documenta os requisitos identificados para o projeto, agrupando temas/assuntos em um Plano de Release (*inclui levantamento e análise de viabilidade*)

Acompanha continuamente os resultados



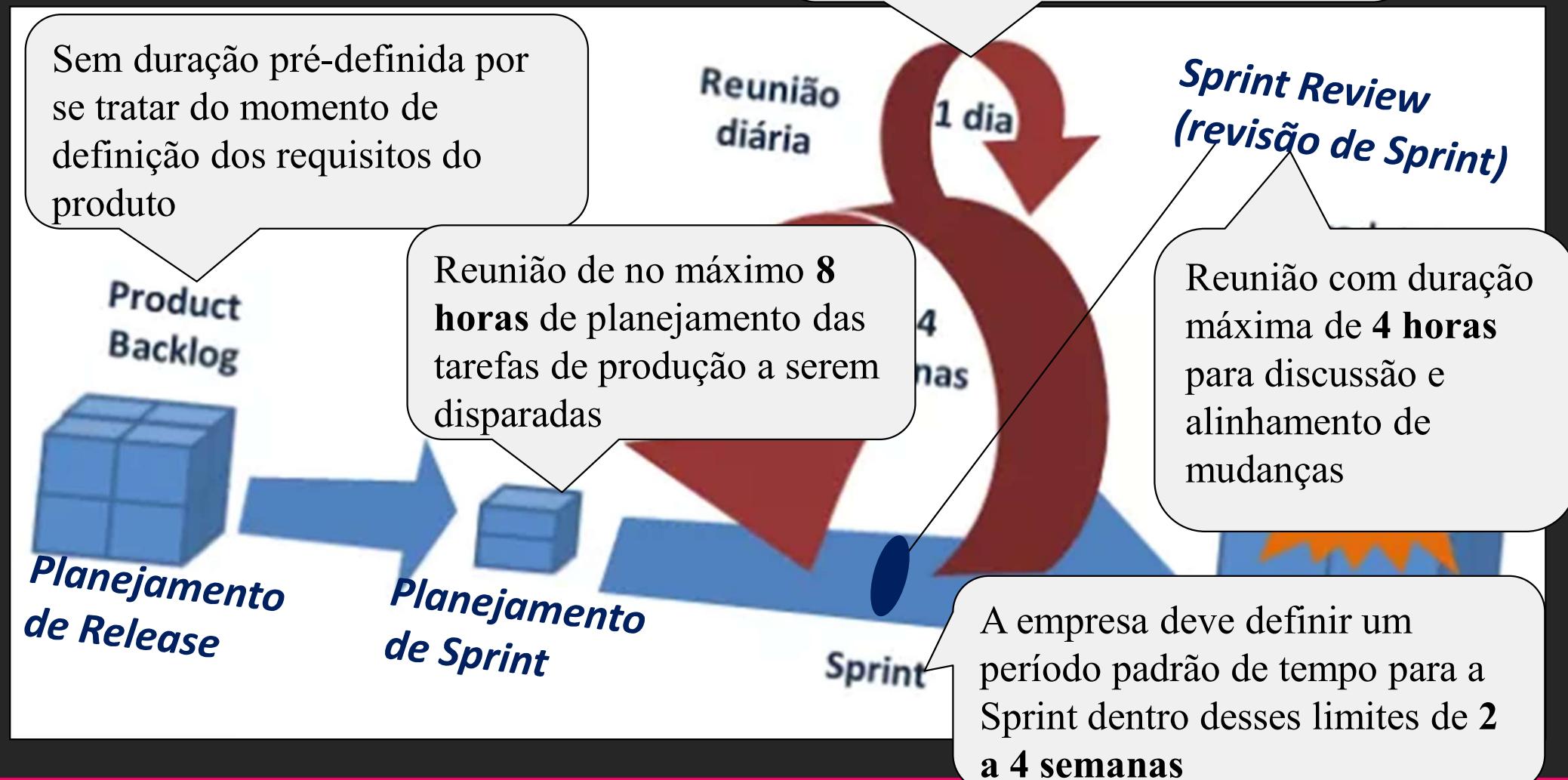
Produz o pacote/pedaço do software, acompanhando e reportando resultados
(*inclui o desenho, construção, teste, capacitação e cargas de dados*)



Pacote/pedaço da solução entregue para uso (*inclui a disponibilização para uso/implantação*)

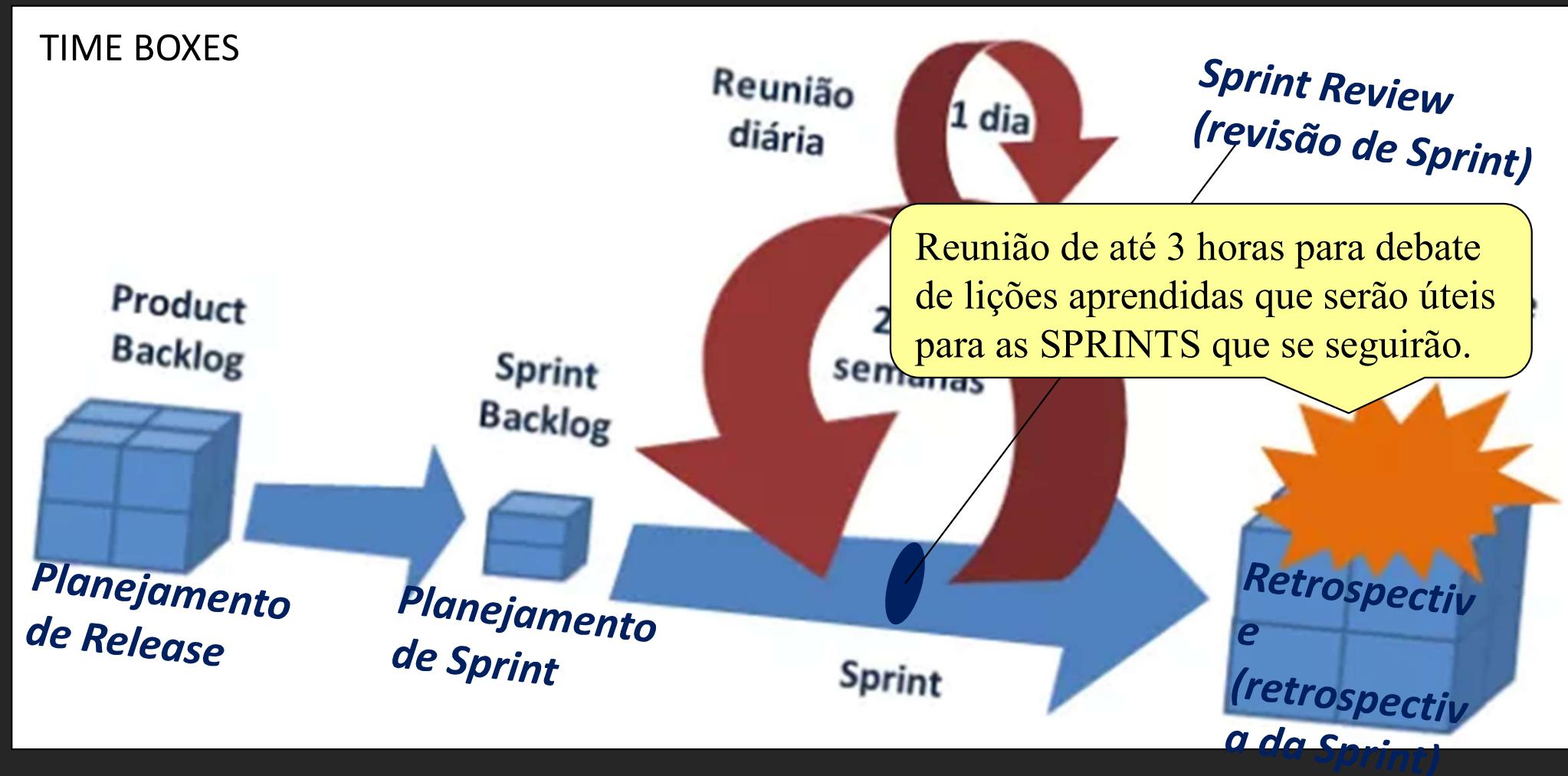
SCRUM

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:



SCRUM – APLICANDO O CICLO DE VIDA ESPIRAL

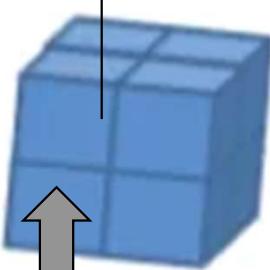
CICLO DE VIDA E CICLO DE VIDA ESPIRAL

SCRUM

Define um/algum

O que o BD precisa atender:

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



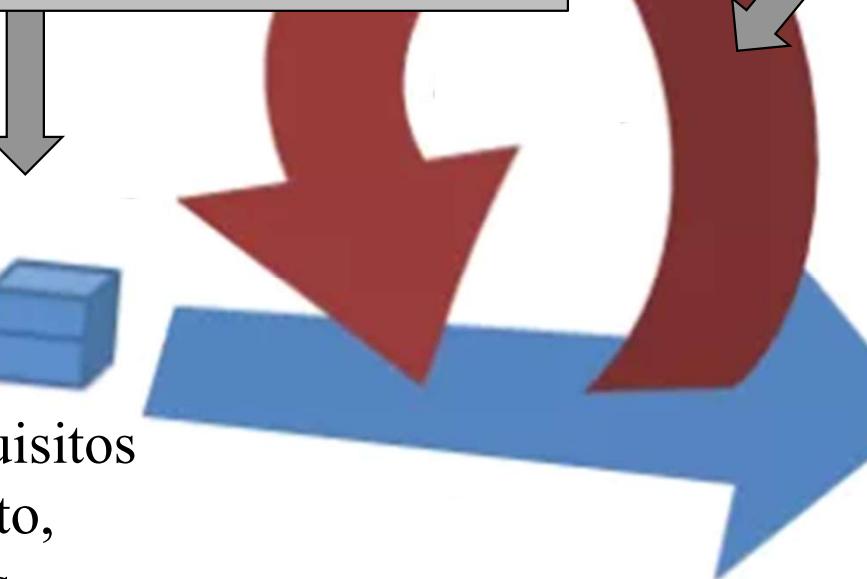
Lista e documenta os requisitos identificados para o projeto, agrupando temas/assuntos

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 e FLEKS

Quando da elaboração do backlog do produto e preparação da argumentação de defesa do projeto, o FLEKS recomenda a observação do QUADRO DE IDEAÇÃO DE SOLUÇÃO para orientar a produção final do documento de Visão do Escopo do projeto, acompanhado de PITCH e CANVAS, conforme interesse.



SCRUM e FLEKS

O Scrum se fundamenta em ciclo de iteração chamados Sprints, que contém cerimônias de planejamento, organização e controle que possibilitam flexibilizar as entregas e tarefas do projeto.

Ao abrir o início dos trabalhos de uma nova Sprint, o modelo FLEKS recomenda que seja elaborado um quadro resumo da interação:



SCRUM e FLEKS

Conforme os itens do backlog de produto vão sendo concluídos, atualize o status no QUADRO DE BACKLOG DE PRODUTO!

Ao final de cada SPRINT, faça a retrospectiva, observando os compromissos da SPRINT (representados no quadro de ideação) e aponte os resultados (quadro de retrospectiva).



SCRUM Funções da equipe e responsabilidades

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



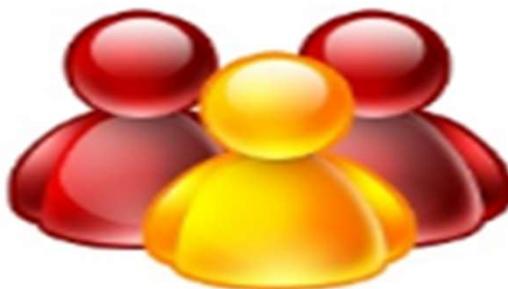
Product Owner (PO)

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)



Time (SQUAD)

Definir metas das iterações
Auto-gerenciamento
Produzir produto com qualidade e valor para o cliente
(atribuído por projeto)

SCRUM Funções da equipe e responsabilidades

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

TIME SCRUM



Product Owner

Define a necessidade de negócio



SCRUM Master

Capacita e avalia a aplicação da metodologia SCRUM



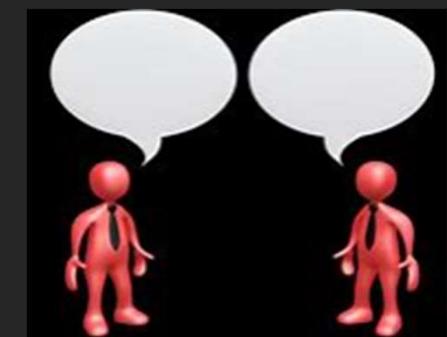
Time de Desenvolvimento

Produz a solução e gerencia o projeto

SCRUM Funções da equipe e responsabilidades

Durante o projeto, ocorrerão **Cerimônias**, que são encontros entre as partes interessadas do projeto para realizar planejamentos ou avaliações de resultados:

- Planejamento de produto
- Planejamento de Sprint
- Reunião de avaliação de status de projeto
- Revisão de plano e entregas
- Retrospectiva de avaliação de lições aprendidas na condução do projeto

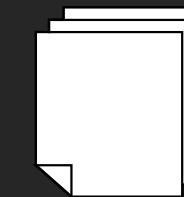


SCRUM Artefatos

A aplicação do SCRUM demanda os seguintes materiais para documentação de apoio ao gerenciamento:

| ID | NOME | DATA DE PRODUÇÃO | NOME DA TAREFA | IMPORTÂNCIA | ESTIMATIVA INICIAL COMO DEMONSTRAR | NOTAS | PRECISÃO | NÚMERO DE PONTOS ESTIMATIVAS RELACIONADOS | RESPONSÁVEL |
|--|--------------|------------------------------------|----------------|-------------|------------------------------------|----------------------------------|--------------|---|-------------|
| Processo 01 - Gestão de cadastro de clientes | ponto-01-001 | Incluir cliente | Alta | 1 ponto | Tela de manutenção | ponto-01-005 | Sprint 1 | João | |
| | ponto-01-002 | Alterar dados do cliente | Alta | 1 ponto | Tela de manutenção | ponto-01-005 | Sprint 1 | Rita | |
| | ponto-01-003 | Consultar cadastro de cliente | Alta | 1 ponto | Tela de manutenção | ponto-01-005 | Sprint 1 | Maria | |
| | ponto-01-004 | Desativar cliente | Alta | 1 ponto | Tela de manutenção | Desativar e não excluir clientes | ponto-01-005 | Sprint 1 | João |
| | ponto-01-005 | Criar tabela de cliente | Alta | 1 ponto | Requerer dados guardados | | Sprint 1 | Pedro | |
| Processo 02 - Consulta de vendas por cliente | ponto-02-001 | Total de compras por cliente | Média | 3 pontos | Cabecalho no BI | | Sprint 2 | João | |
| | ponto-02-002 | Cliente e Local Total condado | Média | 5 pontos | Cabecalho no BI | | Sprint 2 | Pedro | |
| | ponto-02-003 | Cliente e Físico Gerofáixa x Total | Média | 8 pontos | Cabecalho no BI | | Sprint 2 | João | |

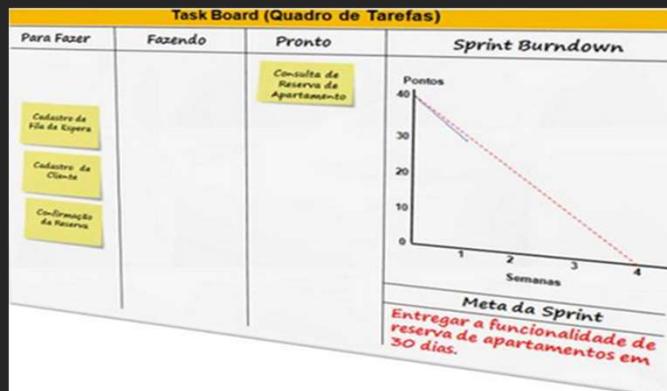
Product backlog



Histórias de usuários

| | | | | | | | | | |
|---|--|--|--|--|--|--|--|--|--|
| FICHA DE TRABALHO | | | | | | | | | |
| ID DA ATIVIDADE/TAREFA: | | | | | | | | | |
| BREVE DESCRIÇÃO (NOME DA ATIVIDADE/TAREFA): | | | | | | | | | |
| RESPONSÁVEL POR REALIZAR A TAREFA: | | | | | | | | | |
| NÚMERO DE PONTOS (PESO ESTIMADO) ATRIBUÍDO: | | | | | | | | | |
| TEMPO ESTIMADO, ASSOCIADO AOS PONTOS: | | | | | | | | | |
| TEMPO REALIZADO ATÉ O MOMENTO (Caso a tarefa não tenha sido concluída no momento do relatório): | | | | | | | | | |
| TEMPO TOTAL DECORRIDO ATÉ A FINALIZAÇÃO: | | | | | | | | | |
| DATA ESTIMADA DE INÍCIO: | | | | | | | | | |
| DATA REAL DE INÍCIO: | | | | | | | | | |
| DATA ESTIMADA DE FINALIZAÇÃO: | | | | | | | | | |
| DATA REAL DE FINALIZAÇÃO: | | | | | | | | | |

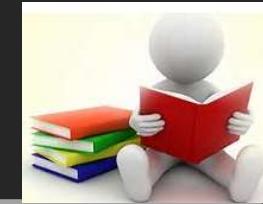
*Sprint backlog
(task cards – cartões de trabalho)*



*Quadro de tarefas
(KANBAN Board e Sprint BurnDown Graph)*

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 'Área de Trabalho' and 'Google Drive' selected. The main pane displays a folder structure under 'RunMIDIA > Google Drive'. A table lists three items: 'Metodologia' (modified 02/02/2016 11:56, type folder), 'Docência (I)' (modified 14/03/2016 21:06, type folder), and 'Recibos 2016' (modified 11/07/2016 16:41, type folder). To the right of the table 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

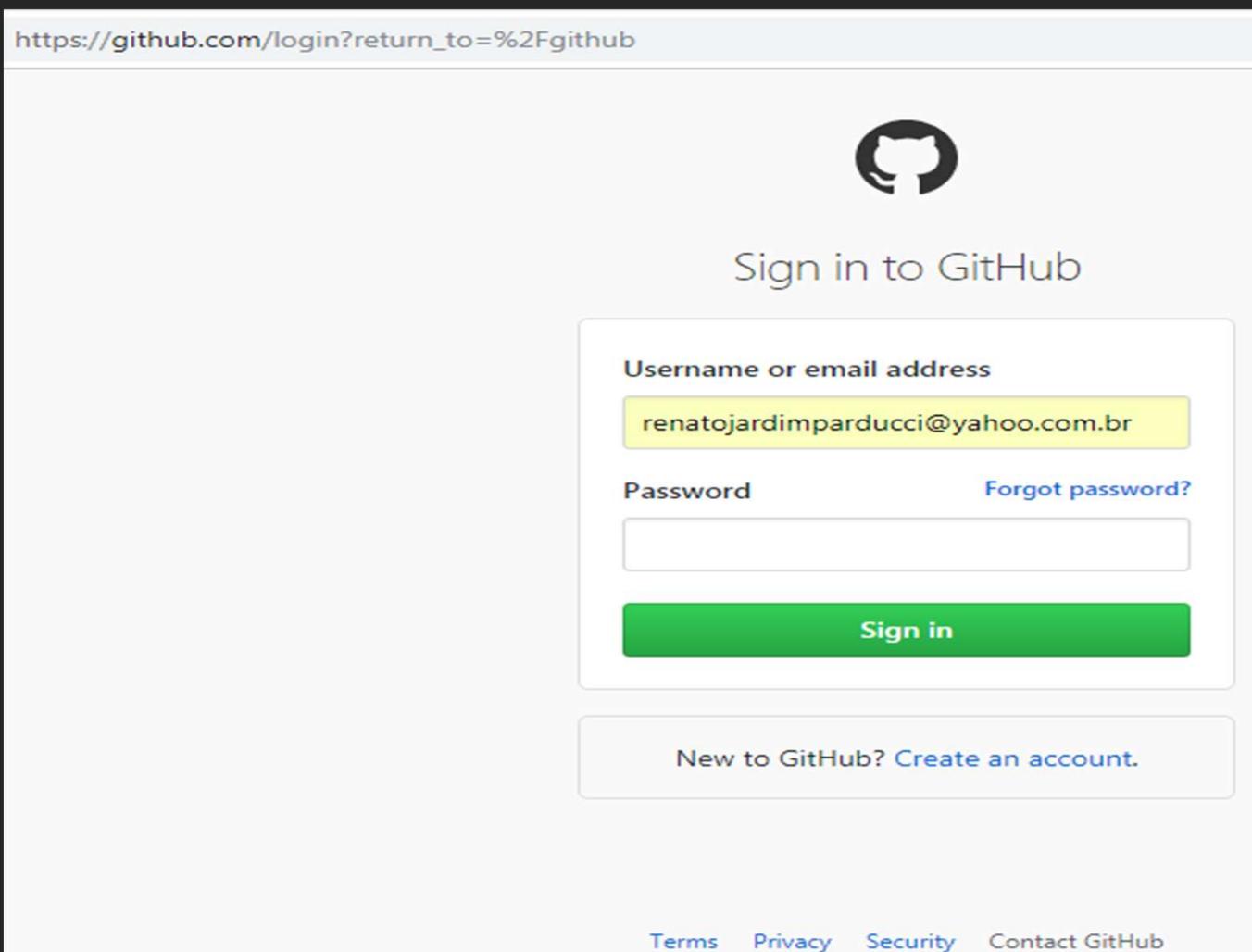


Ponto de necessidade de manutenção para:

- 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, Notifications, +, and a user icon.

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 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

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 displayed. 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. Below the tabs, there's a note: 'No description, website, or topics provided.' and a 'Manage topics' link. On the left, there are stats: 1 commit, 1 branch, Branch: master (dropdown), New pull request button, and a commit history for 'Initial commit' by 'RenatoJardimParducci' (commit ID: 7296d6c). On the right, there are links for 0 releases, 1 contributor, Upload files, Find file, and Clone or download. A red callout box with a downward arrow points to the 'Create new file' button, which is highlighted with a red border. The text inside the callout box reads: 'Crie um nome para a pasta/nome de um arquivo Readme que será criado'.

RenatoJardimParducci / TesteGITHUB

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

Branch: master ▾ New pull request

RenatoJardimParducci Initial commit Latest commit 7296d6c a minute ago

README.md Initial commit a minute ago

README.md

Crie um nome para a pasta/nome de um arquivo Readme que será criado

Create new file

Upload files Find file Clone or download

TesteGITHUB

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

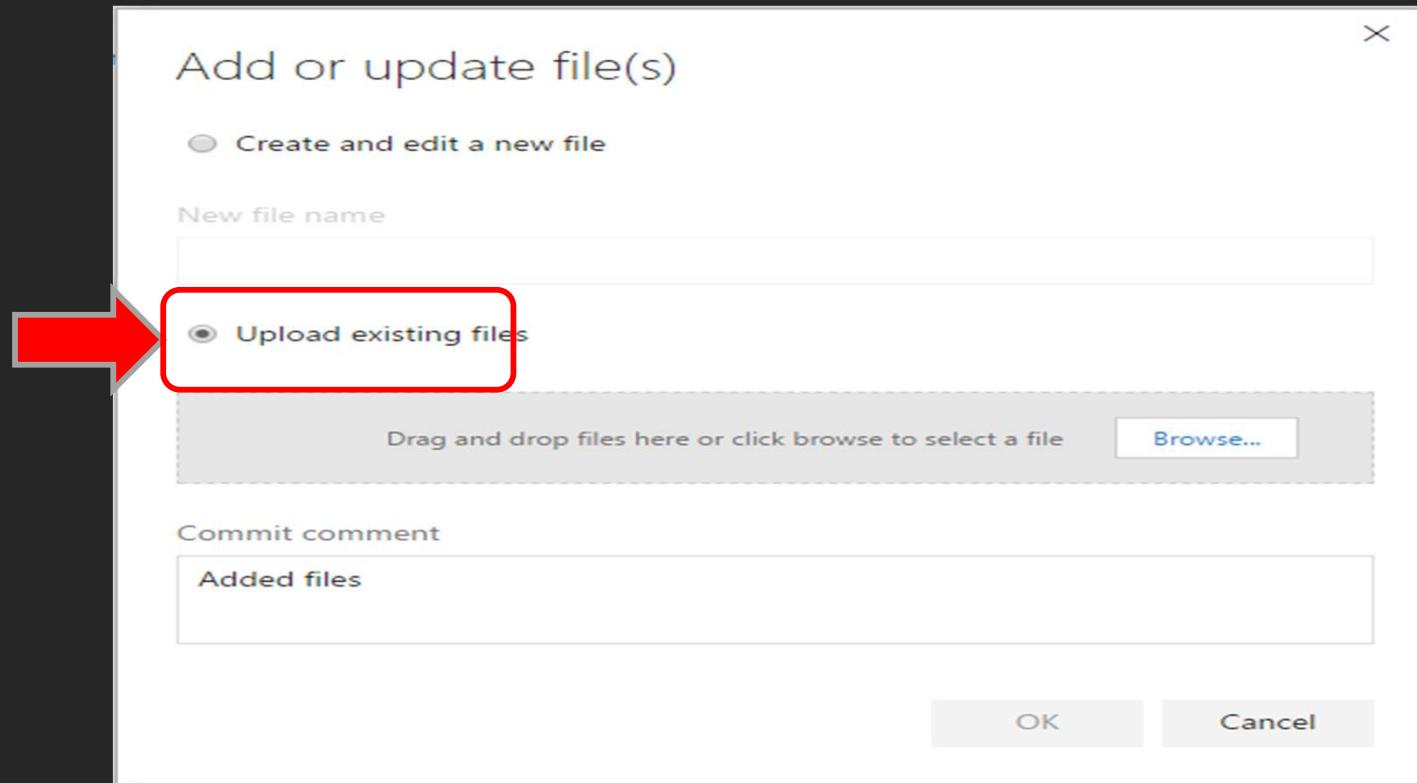
Neste ano, vamos trabalhar diretamente na **Branch MASTER**.

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.

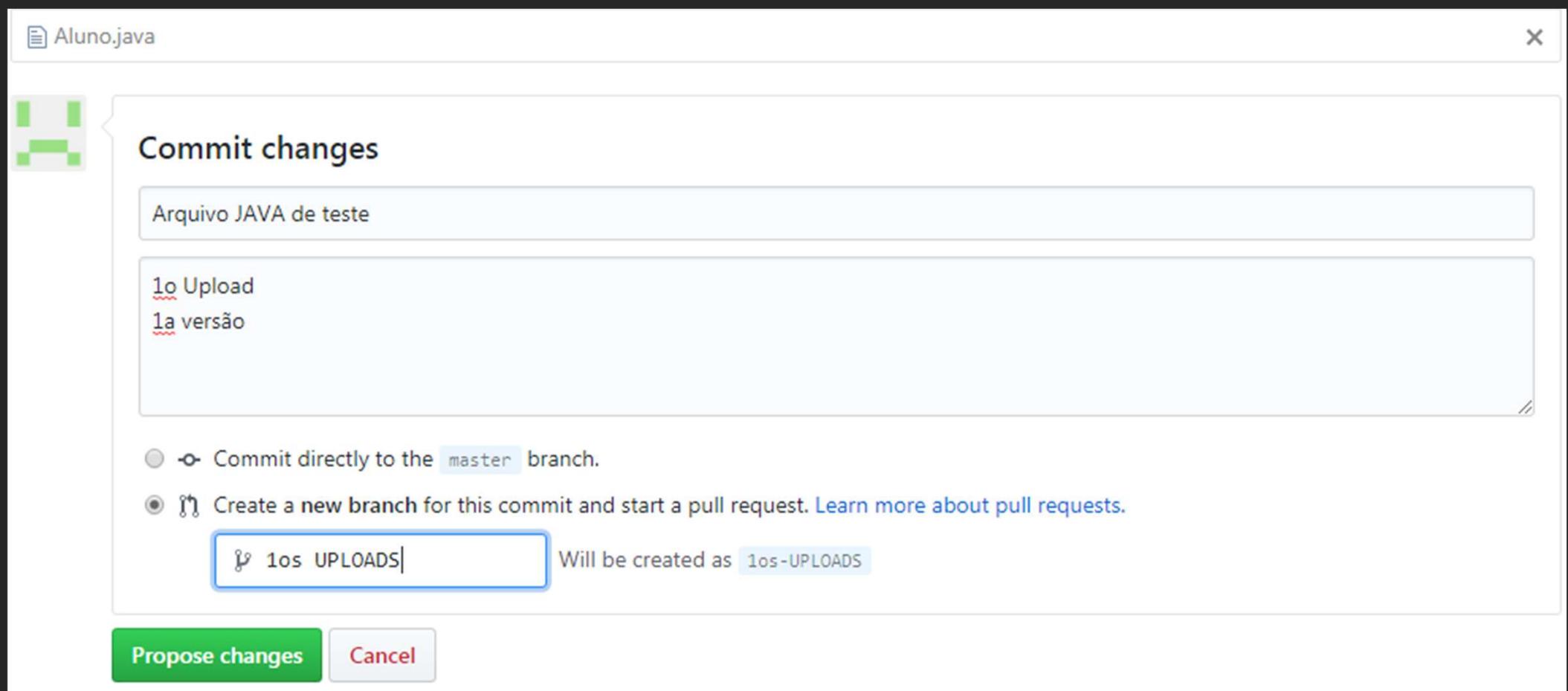
Suba um arquivo .JAVA ou .SQL para experimentar!



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

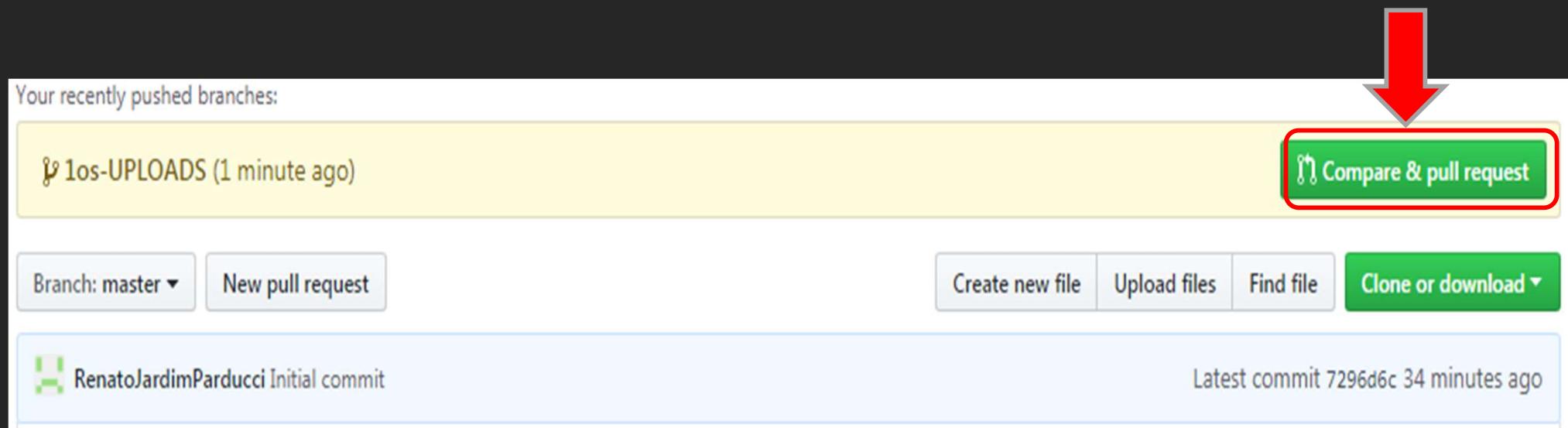
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

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 a bullet point: '- Updated CalcHoras.js'. Below the description, it says 'Markdown supported. Use # to mention a work item or @ to mention a person.' and lists another bullet point: '• 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 placeholder text 'Search work items by ID or title'. At the bottom left, a large red arrow points to a blue button labeled 'New pull request'.

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 file content is a Java class definition:

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

The History button is highlighted with a red box, and a red arrow points to it from the top right.

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 GitHub tutorial with the title "Learn Git and GitHub without any code!". It features a call-to-action button "Read the guide" and a "Start a project" button. Below the video, there is a "Discover interesting projects and people" section with a news feed and a "Visualize your project's community" feature. The video player has standard controls like play/pause, volume, and a progress bar showing 0:30 / 13:49. The background of the video player is a screenshot of a web browser displaying the GitHub homepage.



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:

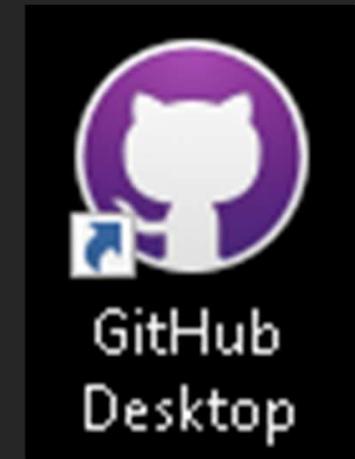
- GIT Gui (por menus em um software cliente);
- GIT CMD/BASH (por linha de comando).

Vamos explorar o uso de GUI (CMD/BASH não serão explorados nas aulas de Data Governance).

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

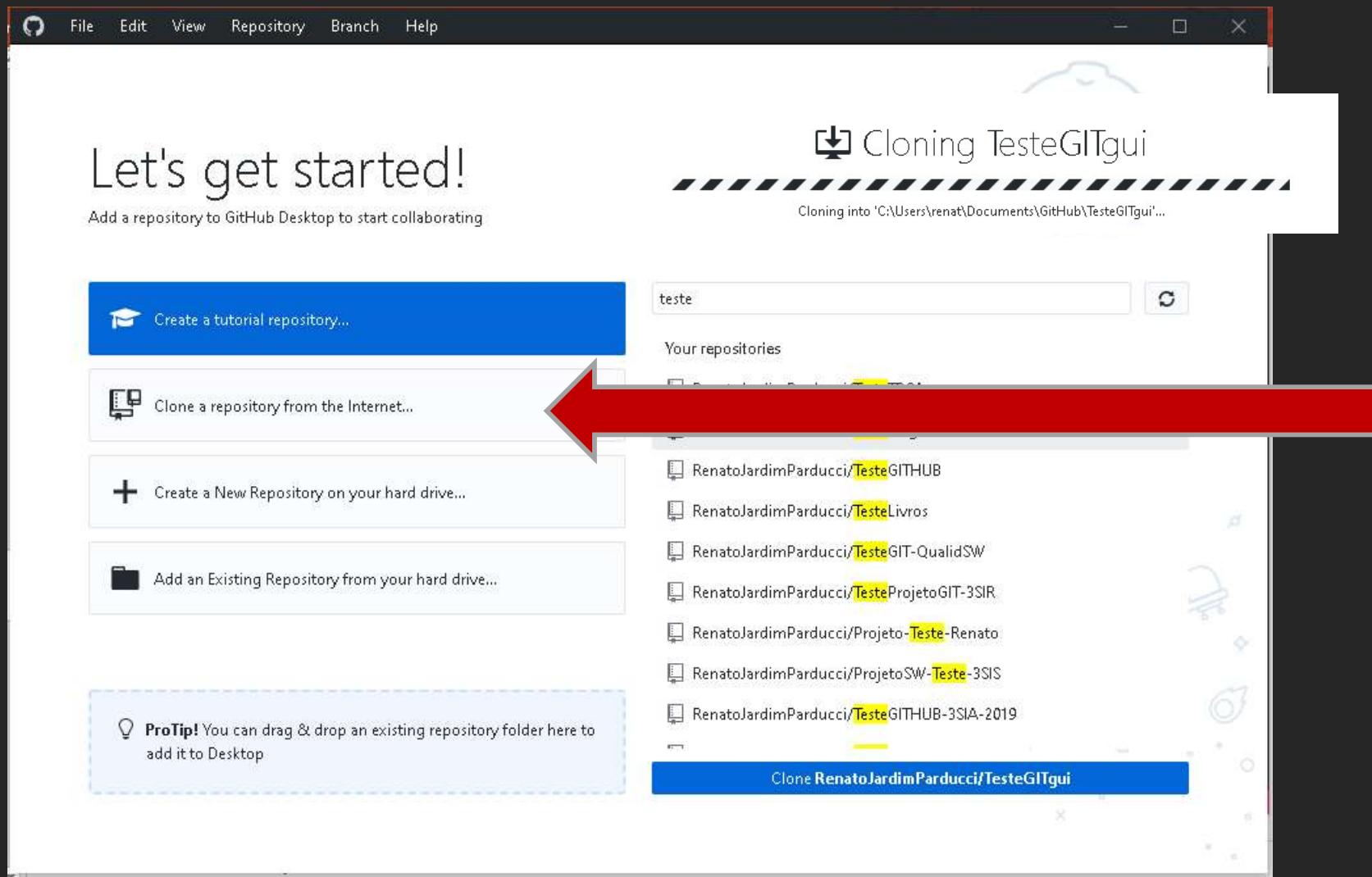
Os aplicativos client permitem que você publique e puxe arquivos para manutenção usando somente o seu computador pessoal, sem necessidade de usar o navegador, como nós fizemos no nosso exemplo.

Vamos explorar agora o GitHub Desktop que pode ser obtido no próprio portal do GITHUB.



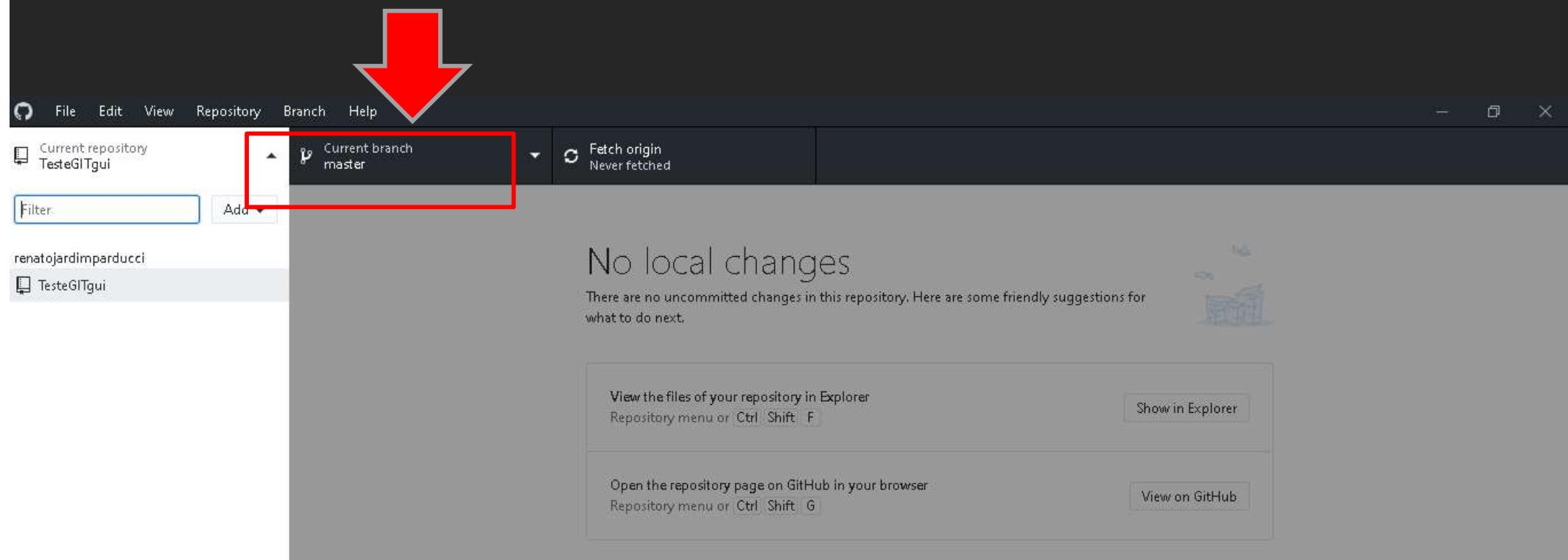
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Você pode sincronizar os repositórios em nuvem com o computador pessoal.



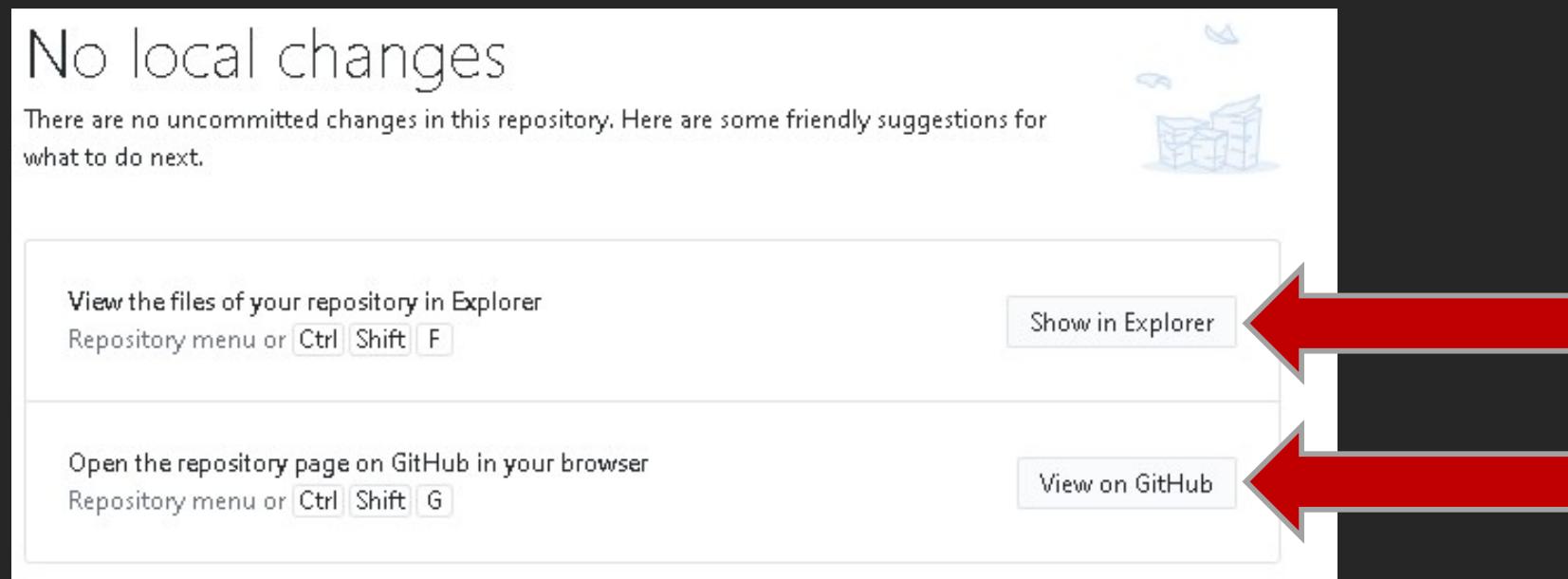
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Com o GitGUI, por exemplo, você pode acessar as áreas Master, Branches e pastas pelos menus o seu PC..



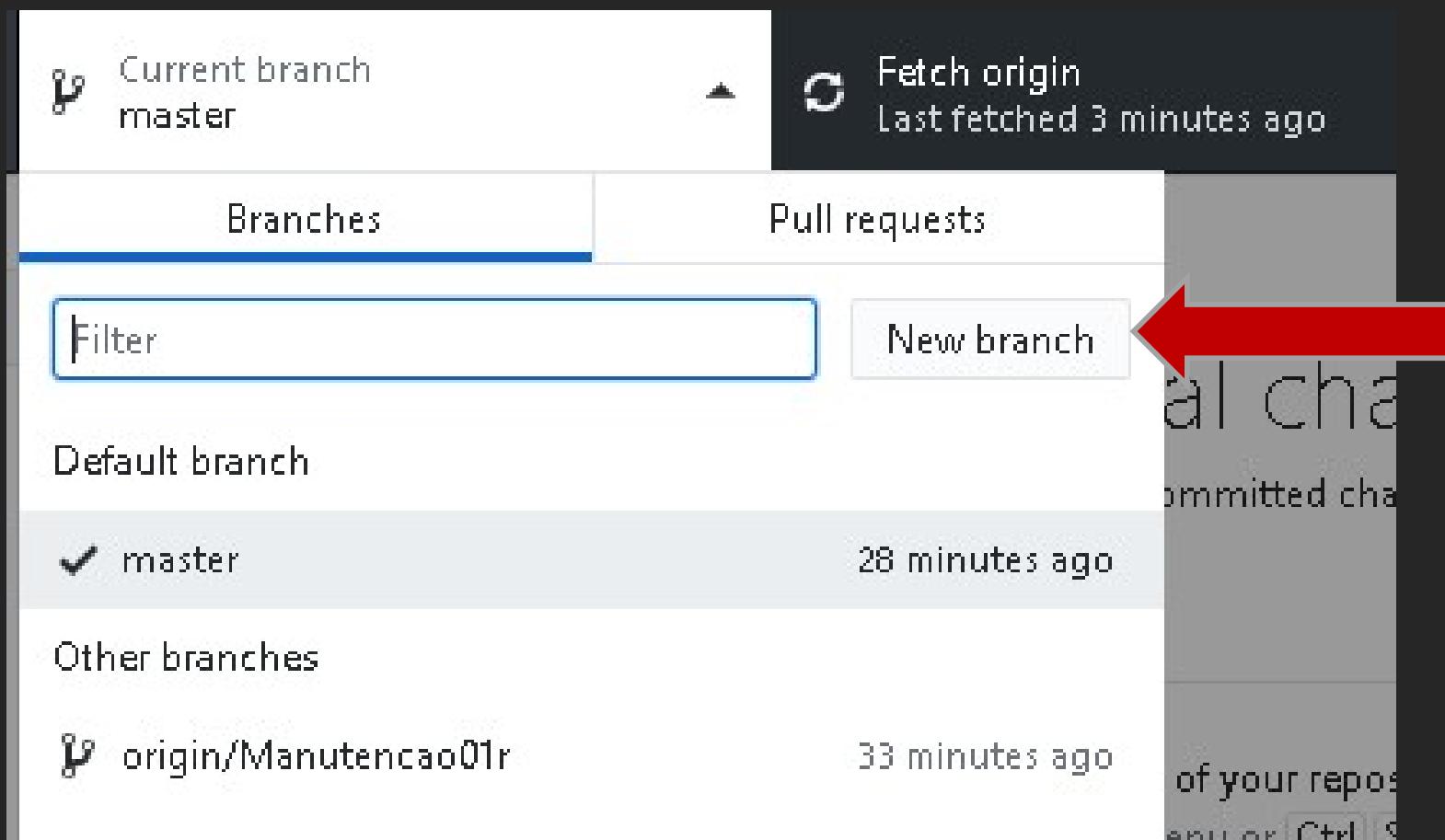
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Você pode acessar o clone local ou a pasta remota do GITHUB por esse software cliente.



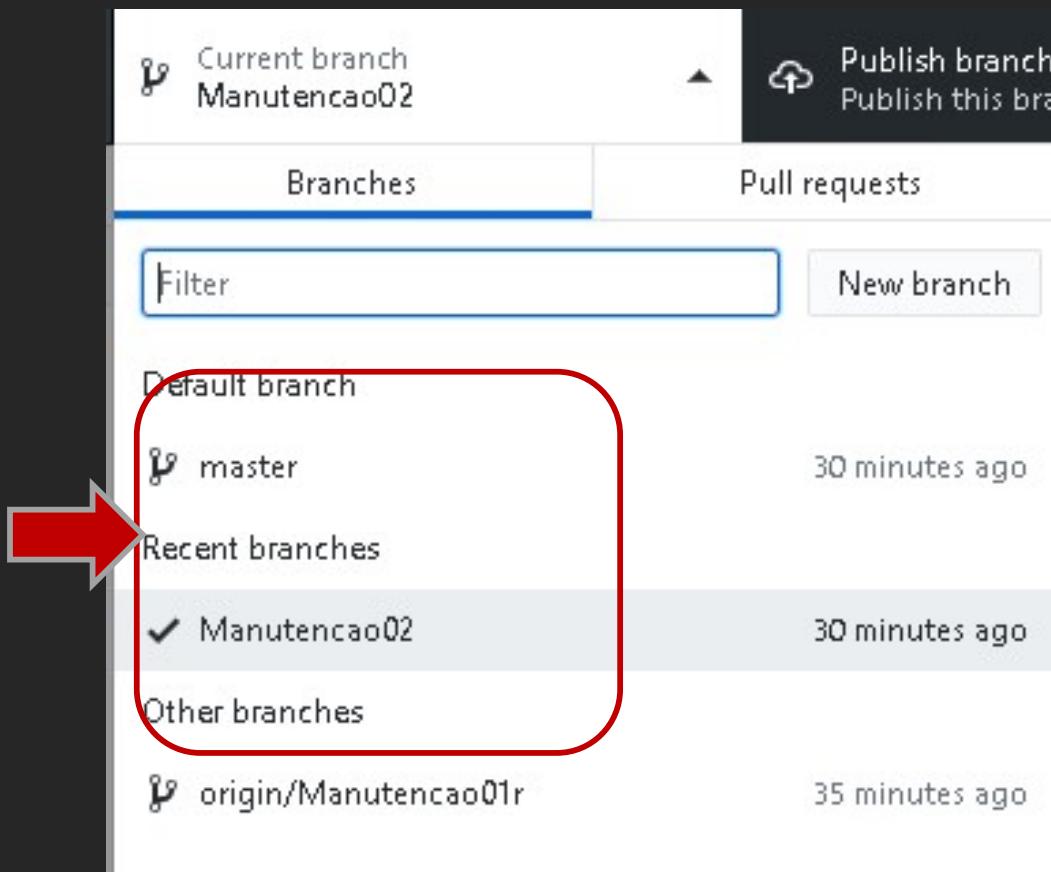
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Pode ainda criar branch...



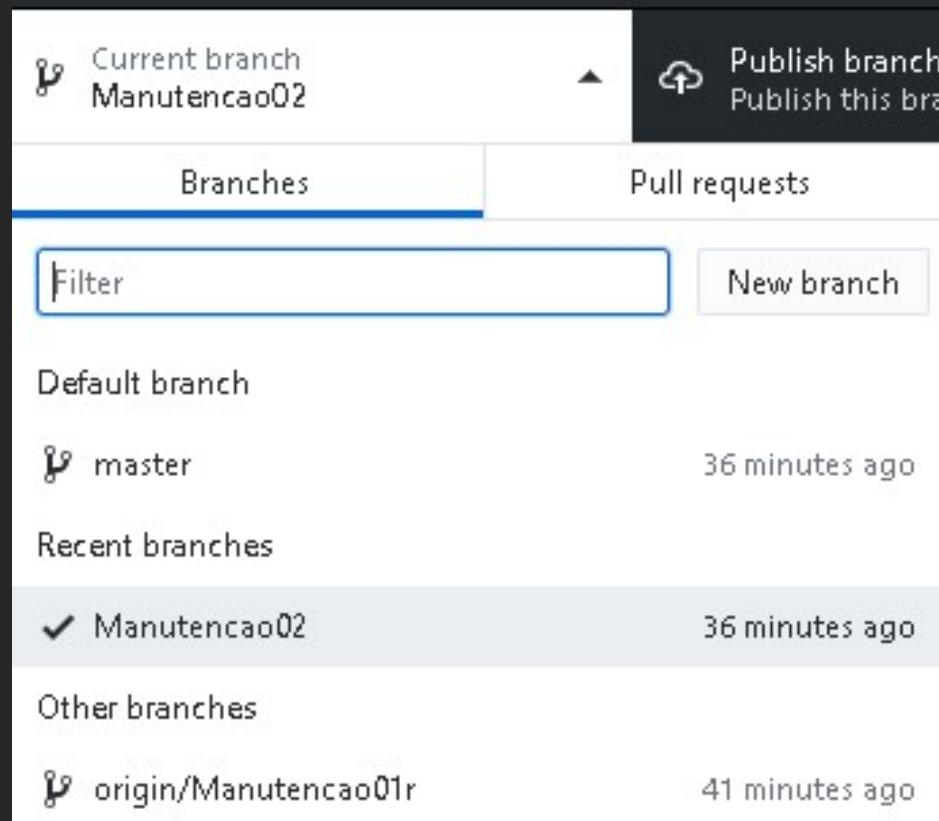
GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

E essa Branch quando criada localmente, pode ser visualizada no GitHub client.



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

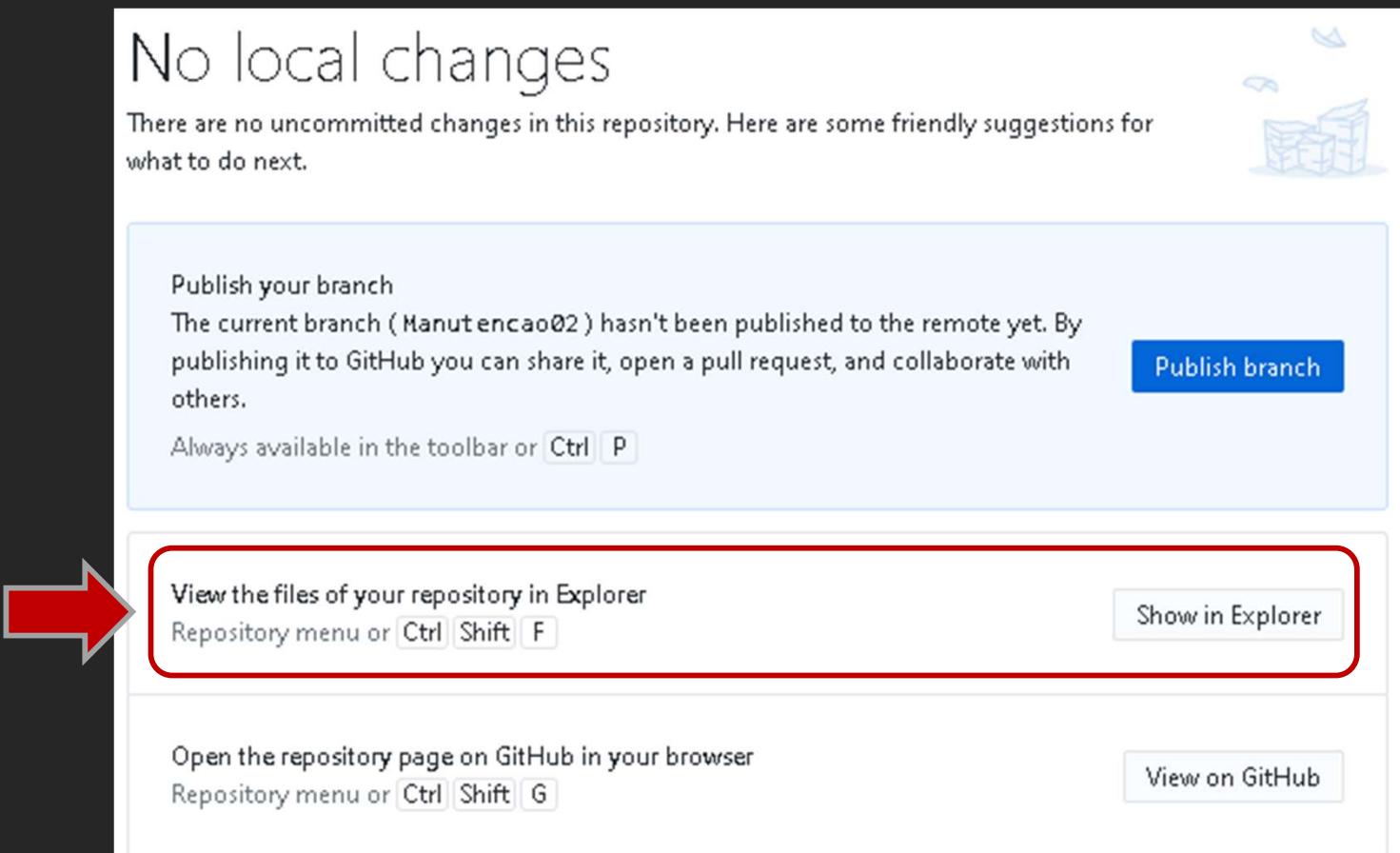
Para editar arquivos na Branch, faça um Checkout!



Selecione a branch no menu de navegação da ferramenta!

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Após criada a Branch, a navegação de acesso aos fontes é via menu da aplicação GUI.



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Assim que alterado o arquivo ou criado um arquivo novo na pasta do GIT local, o GitHub Client vai mostrar as diferenças entre a Master e a Branch.

The screenshot shows the GitHub Desktop application interface. The top navigation bar includes File, Edit, View, Repository, Branch, and Help. The repository dropdown shows "Current repository: TesteGITgui" and the branch dropdown shows "Current branch: Manutencao02". A "Publish branch" button is also present. The main window displays a "Changes" tab with 1 changed file, "README.md". The diff view shows three lines of code: line 1 is "# TesteGITgui", line 2 is "-# Texto NOVO", and line 3 is "+# Texto NOVO MESMO!!!".

| Line | Content |
|------|------------------------|
| 1 | # TesteGITgui |
| 2 | -# Texto NOVO |
| 3 | +# Texto NOVO MESMO!!! |

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Se você desejar publicar a Branch criada no seu PC no GitHub.com, primeiro, faça um Commit no Git local



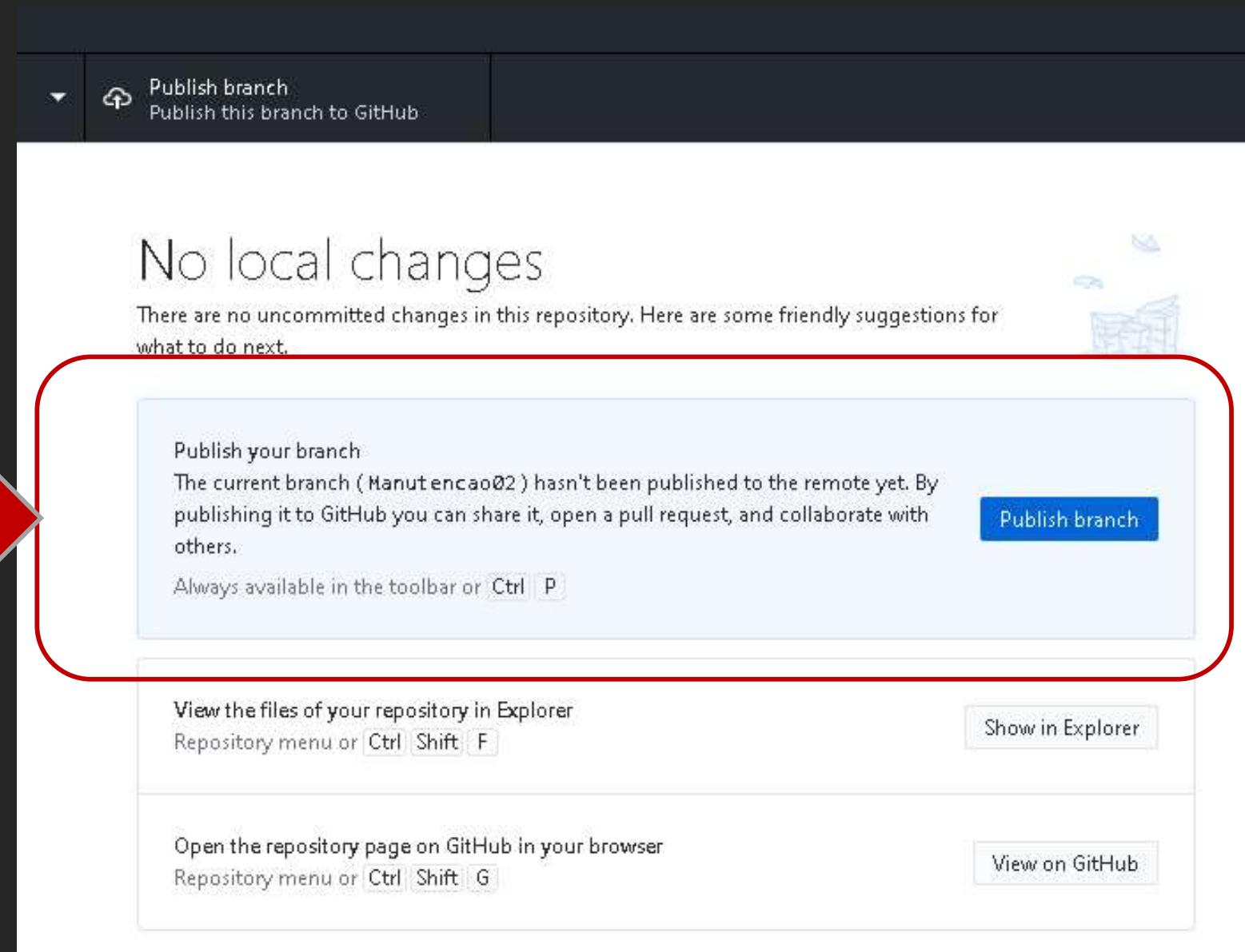
The screenshot shows a Git commit interface. At the top, it displays the current repository as 'TesteGITgui' and the current branch as 'Manutencao02'. A button for publishing the branch to GitHub is also visible. The main area shows a 'Changes' tab with one changed file, 'README.md'. The diff view shows the following changes:

| Line | Line | Change |
|------|------|---------------------------|
| 1 | 1 | @@ -1,2 +1,2 @@ |
| | | # TesteGITgui |
| 2 | 2 | -# Texto NOVO @@ |
| | | +# Texto NOVO MESMO!!! @@ |

At the bottom, there is a 'Commit to Manutencao02' button.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Faça o PUSH no
GITHUB remoto
(WEB).



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

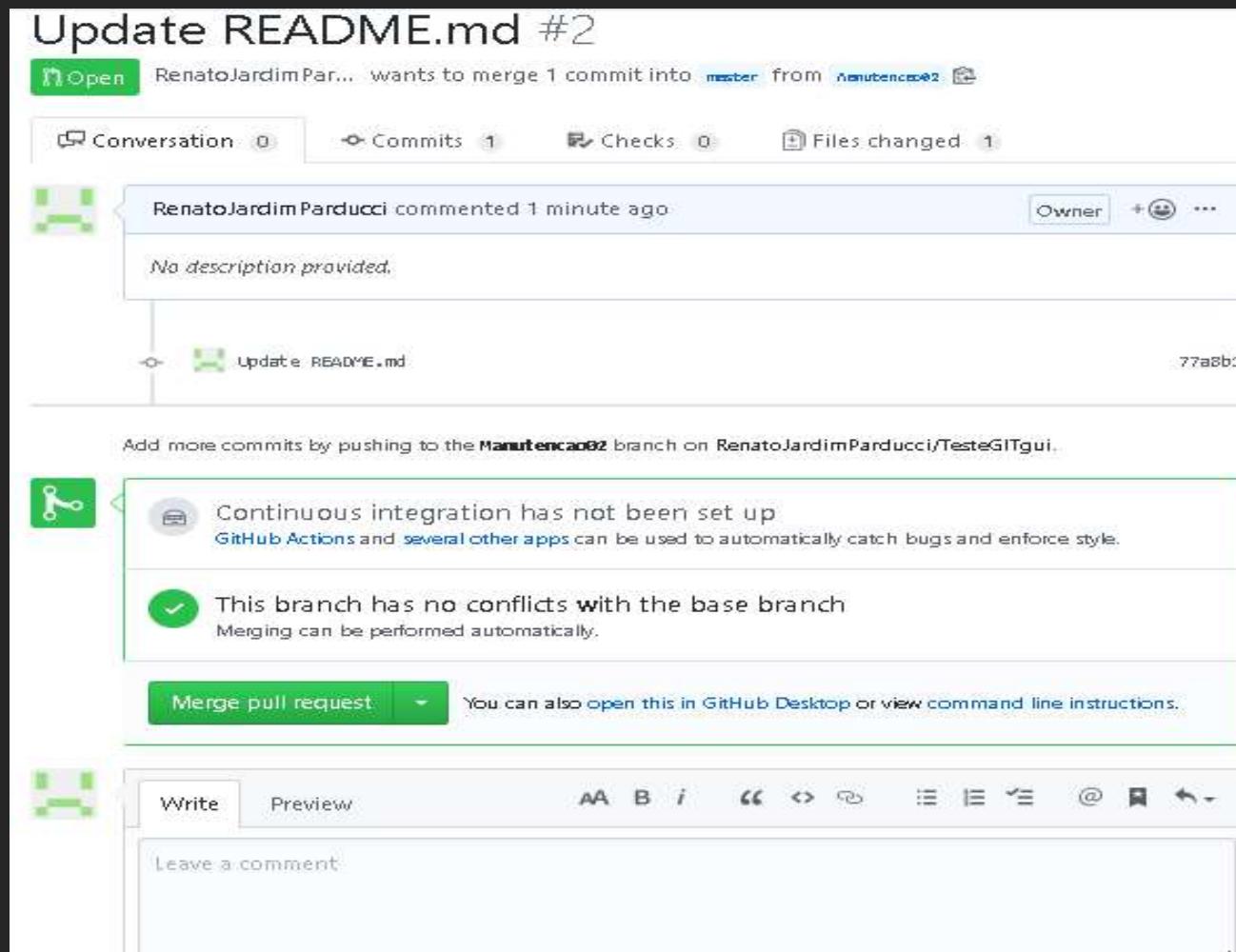
As edições são refletidas do GitGui para o GitHub!

Crie a Pull Request remota via GITHUB Client para atualizar a Master remota.

The screenshot shows the GitHub desktop application interface. At the top, it says "Fetch origin" and "Last fetched 2 minutes ago". Below this, a large button displays "No local changes". A message below the button states: "There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next." Three options are listed in boxes: 1) "Create a Pull Request from your current branch" (with a "Create Pull Request" button), 2) "View the files of your repository in Explorer" (with a "Show in Explorer" button), and 3) "Open the repository page on GitHub in your browser" (with a "View on GitHub" button). The background features a faint illustration of a city skyline.

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

Ao solicitar a Pull Request, o Git Hub é acionado. Você fará Login e confirmará o Merge/Pull Request!



GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

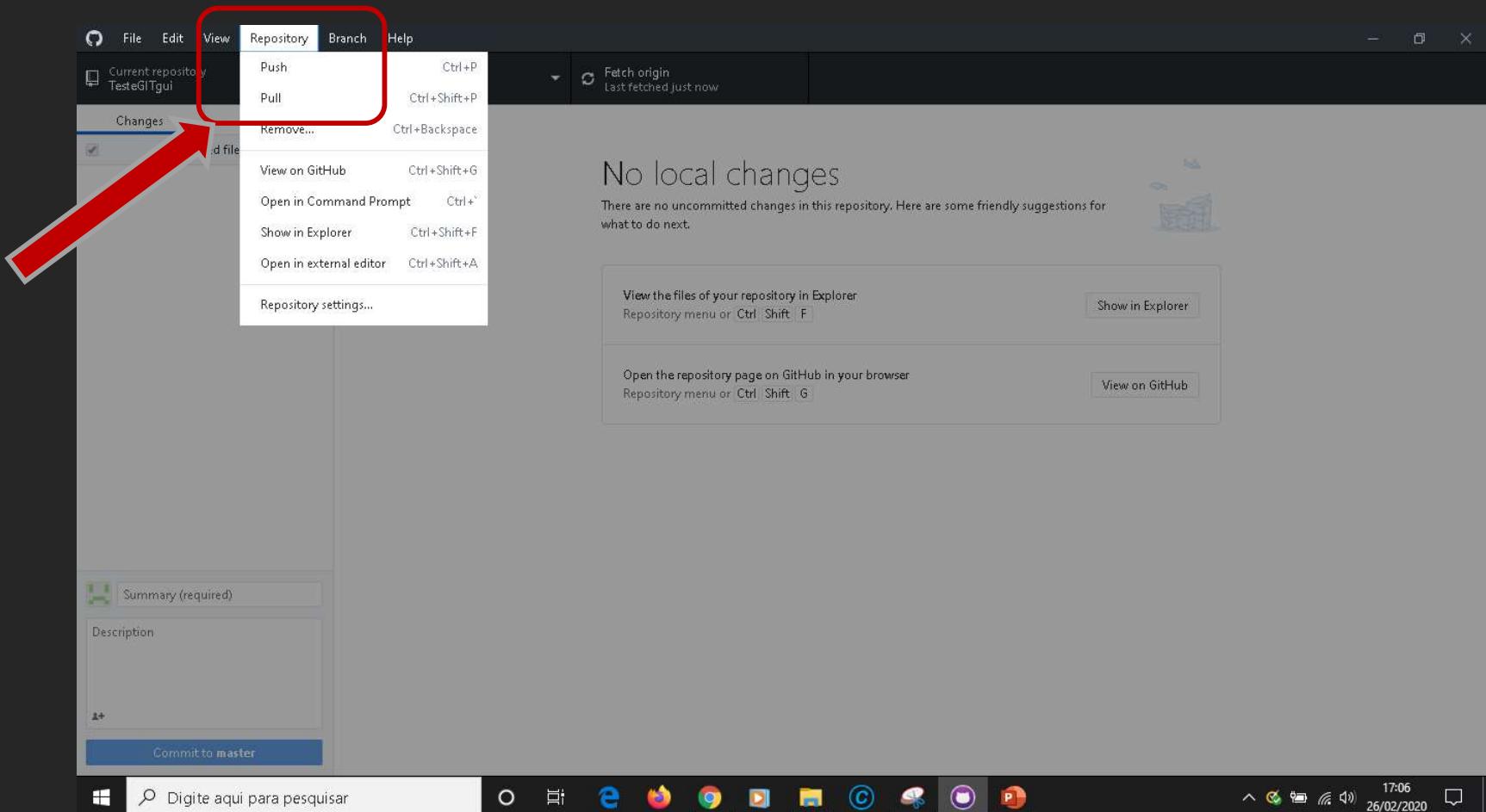
Com esses procedimentos, você consegue fazer Check out e Check in (Push) , mantendo sincronia entre o ambiente remoto dos fontes e o seu ambiente local de trabalho.

Após os Pushes, confira sempre se o GitHub.com foi atualizado corretamente.

The screenshot shows a GitHub repository interface. At the top, there are statistics: 5 commits, 3 branches, 0 packages, 0 releases, and 1 contributor. Below this, a pull request from RenatoJardimParducci is shown, merged into the master branch. The README.md file has been updated 9 minutes ago, with the new content "TesteGITgui" and "Texto NOVO MESMMO!!!".

GERENCIAMENTO DA PRODUÇÃO E ATUALIZAÇÃO DE SOFTWARE

IMPORTANTE: o fato de você ter atualizado a Branch e Master no GITHUB não implica que a Master Local tenha sido atualizada.
Para atualizar a Master Local, faça um Pull do Repositório.



ASSISTA OS VÍDEOS NO CANAL DO PROFESSOR



https://youtu.be/ps_ZTFrasAg

A screenshot showing a browser window with a GitHub repository page and a video player window side-by-side. The GitHub page displays a repository named 'RenatoJardimParducci / ExemploGitDesktop'. It shows 1 commit, 1 branch, 0 packages, and 0 releases. The README.md file contains the text 'Initial commit'. The video player window shows a close-up of a man's face, identified as Prof. Renato Jardim Parducci, speaking. A small text at the bottom right of the video player says 'Double Click on image to play movie'.



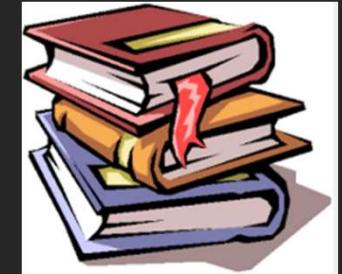


D Ú V I D A S

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.
- PMBoK 7^a edição. Disponível em [PMBOK Guide | Project Management Institute \(pmi.org\), 2022.](https://www.pmi.org/-/media/assets/project-management-best-practices/guides-and手冊s/pmbok-guide-2022.ashx)
- COSTA, Helio. FLEKS Guia completo. www.fleksmodel.com, 2022.



BONS ESTUDOS!