[aula 1 - 05/08/2019 - sala f10 - prof. Danilo Graça]
danilo.graca@etep.edu.br
projeto: escrever um projeto UML
livros: Engenharia de software uma abordagem profissional – Pressman

Definir o que é cada conceito e mandar para o email do professot:
- UML: Unified Modeling Language, é uma linguagem que define uma série de artefatos que nos ajuda na tarefa de modelar e documentar os sistemas orientados a objetos que desenvolvemos.
- padrão de projeto: Os padrões de projetos tornam mais fácil reutilizar soluções e arquiteturas bem sucedidas para construir softwares orientados a objetos de forma flexível e fácil de manter. O uso de padrões de projeto pode reduzir a complexidade do processo de projetar software. Além disso, o software orientado a objetos bem projetado possibilita aos projetistas reutilizar e empregar componentes preexistentes em sistemas futuros.
- padrão de código: O uso de um padrão de codificação é uma prática de desenvolvimento de software extensamente aceita. A necessidade desta prática reside na importância da criação de um ambiente altamente colaborativo. A equipe deve ter uma forma padrão para nomear e formatar o código para que possam compreendê-lo rapidamente e saber onde procurar o que precisam. Isto permite a propriedade do código compartilhado de forma que qualquer membro da equipe possa entender rapidamente o código escrito pelos outros.
Idealmente, o padrão de codificação deve ser o resultado do consenso da equipe. O envolvimento dos membros da equipe ajudará na adoção dos padrões.
Os Padrões de Codificação cobrem as seguintes áreas:
Padrão de nomenclatura. Inclui a nomenclatura de elemento de todas as formas até a menor variável.
Organização de arquivos. Inclui a convenção de nomenclatura de arquivos e como os arquivos serão organizados no sistema de arquivos.
Padrões para comentários. Muita ênfase nos comentários implica em uma falta de confiança de que o código escrito não é inteligível, e também existe sempre uma preocupação de que os comentários não estejam atualizados. No entanto, uma abordagem coerente para os comentários melhora o entendimento e pode suportar a capacidade de gerar documentação a partir do código.
Convenções de Codificação A aplicação coerente de convenções específicas em nível de código e a exclusão de algumas formatações consideradas pobres melhoram a qualidade do código.
Espaço em branco. Embora se possa argumentar ser menos crítico do que os outros itens listados aqui, o uso consistente de espaços em branco, como indentações e linhas em branco, também melhora a legibilidade.
- gestão de mudança: A gestão da mudança é uma metodologia estruturada para preparar a empresa, gradualmente, para assimilar mudanças importantes. Ela deve proporcionar todas as condições para que o colaboradores e o negócio se adaptem a mudança com o menor atrito possível.
- modelo cascata: O modelo cascata é utilizado principalmente quando os requisitos de um determinado problema são bem compreendidos. Uma forma de utilizar o

modelo cascata é quando precisamos fazer adaptações ou aperfeiçoamentos em um sistema já existente. Por exemplo, quando temos um sistema já pronto e precisamos fazer uma adaptação porque alguma lei governamental foi alterada ou criada.
Também podemos utilizar o modelo cascata quando um software necessita de uma nova funcionalidade e os requisitos estão bem definidos e são estáveis.
O modelo cascata também é chamado de ciclo de vida clássico ou tradicional.
Este modelo sugere uma abordagem sequencial e sistemática para o desenvolvimento de software.
- MVP: É uma prática de administração de empresas que consiste em lançar um novo produto ou serviço com o menor investimento possível, para testar o negócio antes de aportar grandes investimentos. É o menor produto possivel
- padrão de arquitetura: Um padrão de arquitetura é uma solução geral e reutilizável para um problema que ocorre com frequência em arquitetura de software dentro de um determinado contexto. Padrões de arquitetura são similares aos padrões de projeto de software, mas possuem um escopo mais amplo.
- auditoria de qualidade: É o teste de qualidade do produto ou software
- diagrama de caso de uso: Descreve a funcionalidade proposta para um novo sistema que será projetado, é uma excelente ferramenta para o levantamento dos requisitos funcionais do sistema.
- métodos ageis: A engenharia de software ágil combina filosofia com um conjunto de princípios de desenvolvimento. A filosofia defende a satisfação do cliente e a entrega de incremental prévio; equipes de projetos pequenas e altamente motivadas; métodos informais; artefatos de engenharia de software mínimos e, acima de tudo, simplicidade no desenvolvimento geral. Os princípios de desenvolvimento priorizam a entrega mais que a análise e projeto (embora essas atividades não sejam desencorajadas); também priorizam a comunicação ativa e contínua entre desenvolvedores e clientes.
- MVC: model, view e controller
- EAP: EAP significa Estrutura Analítica de Projetos ou em inglês WBS – Estrutura analítica do trabalho. A EAP é estruturada em árvore exaustiva, hierárquica e utilizada para evidenciar o que é realmente necessário para a execução de um projeto, desmembrando as fases e facilitando a realização das tarefas.

[aula 2 - 20/08/2019 - sala f10 - prof. Danilo Graça]

Software - é um conjunto de componentes lógicos; estrutura de dados; documentação;
um software é desenvolvido e não fabricado.
Processo de software - o processo de software tem um ciclo de desenvolvimento, ele não se desgasta, mas se deteriora, pois há muitas mudanças na análise de negócio;

como melhorar as mudanças?
mapear as mudanças, pode se encaixar em gestão de configuração e mudança;

> 'curva da banheira' fala sobre hardware, fala que na primeira fase tem muitos problemas chamam-se de "mortalidade infantil"
se passado a primeira fase, o hardware tem uma vida útil que pode expirar
a última fase, perto do fim da vida útil, o hardware passa a ter mais e mais falhas até não funcionar mais. O hardware se desgasta.

> a indústria de software move-se rapidamente;
> criam-se nichos específicos;
> desenvolvimento baseado em componentes;
> grande maioria busca softwares personalizados;
> modelo de negócios disruptivo- trabalhando na barreira da inovação (tesla)
> softwares personalizados: insere mais problemas e bugs;
> COST - software pronto de prateleira;
> custom-build -> produzir seu próprio software para ter menos bugs;

obs: Java nasceu para rodar em micro-ondas;
obs: focar em data Science, cientista de dados;

Tipos de softwares

- **Softwares de sistemas** - pensa no sistema e depois no fim desenvolve o sistema, por exemplo Avião;
- **Softwares de aplicação** - aplicativos, word, excel..;
- **Software cientifico de engenharia** - malab., aplicações especificas onde faz uma análise e dá um resultado;
- **Software embarcado** - software que funcionam em placas de circuito, para rodar em placa, como Arduino;
- **Software para linhas de produtos** - geladeira, máquina de lavar;
- **Aplicações Web/ Apps** - aplicativos mobile;
- **Software de inteligência artificial** - são softwares específicos;

Software legado - é um software que funciona sem bugs, mas que tem uma tecnologia obsoluta;

Por que softwares devem ser renovados?
1. precisam ser melhorados para novas análises de negócio;
2. precisam ser melhorados para trabalhar com outros sistemas com tecnologias mais avançadas;

interoperabilidade - operar com qualquer coisa;

Cloud Computing
- infraestrutura como um serviço - IaaS
- software como um serviço - SaaS
- plataforma como serviço: PaaS

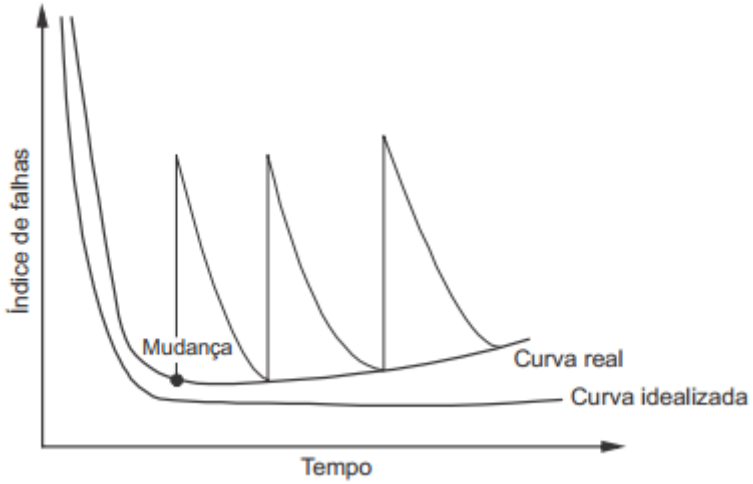
Exercício

1- forneça exemplos positivos e negativos que indique o impacto do software na sociedade.
R. positivos: os softwares tornam a vida das pessoas muito mais ágeis e fáceis;
negativos: os softwares podem ter atualizações continuas que podem afetar a usabilidade dos usuários;

2- Muitas aplicações modernas mudam com frequência, antes de apresentadas ao usuário final e só então a primeira versão ser colocada em uso. Indique maneiras de construir software para impedir que deterioração decorrente de mudanças;
R. Ter uma boa analise de negócio e de requisito, e se mesmo assim ainda houver mudanças no sistema decorrente de novas análises de requisitos
a equipe deve mapear as mudanças para analisar o caso junto a gestão se de configuração e mudança;

[aula 3 - 27/08/2019 - Danilo Graça]

Revisão: gráfico de software, conforme há mudanças o software se detereora.



Engenharia de Software

Desafios que envolvem o desenvolvimento de SW:
- Incorporado em todos os aspectos de nossas vidas
- Cada vez mais funcional, complexo e interoperável;
- Demanda maior compreensão do problema para o desenvolvimento ser mais assertivo, com isto, projetar tornou-se essencial;
- Falhas pequenas à catastróficas;
- Maior qualidade e facilidade de manutenção;

Cidade inteligente: A partir da internet 5g, vc tem tudo conectado via internet; Semáforos, ônibus, câmeras de segurança. Todos sincronizados em tempo real.

Engenharia de software:

- uma abordagem sistemática, disciplinada e quantificável, a qual podemos traduzir como um processo.
- Aplicação de um processo;
- Esta abordagem deve ser customizada por empresa, projeto, equipe, agilidade..etc
- Possui qualidade como principal diretriz;
- Evolução continuada em busca de eficácia;

Melhoria continua: melhorar processos que trazem benefícios, que podem ser: novas ferramentas, novos recursos. Etc.

Processos: estabelece as diretrizes para o desenvolvimento e controle do projeto. Ou seja é conjunto de coisas que tem na empresa;
Exe: Requisitos, testes, arquiteturas;

Métodos: fornecem informações técnicas para desenvolvimento e envolvem uma ampla variedade de tarefas;

Ferramentas: Suporta automatizado ou semi-automatizado para o desenvolvimento de atividades e o proprio processo;

Exercício

Foco na qualidade. Identifique os princípios básicos de um programa de gestão da qualidade total:

1. Total satisfação do cliente
2. Gerencia participativa
3. Delegação
4. Garantia da qualidade

Dia 14 Jugue Vale – para horas complementares

Problemas no desenvolvimento de projetos

- Problemas de comunicação;
- Escopo não definido adquadamente;
- Recursos humanos insuficientes;
- Riscos não avaliados corretamente;
- Falta de uma metodologia de apoio;

O que é processo?

Roteiro para elaborar um produto ou sistema.

Porque é importante o processo para desenvolver um produto/software?
Controle, a gestão de projetos vai ser mais clara e mais bem definida.

Quais os beneficios de um processo?
- Conhecer e institucionalizar o fluxo de trabalho;
- Padronizar atividades e tarefas;
- Documentar o conhecimento;
- Identificar oportunidades de melhoria;
- Definir papéis e responsabilidades;
- Melhorar a qualidade do produto;

Como desenvolver um processo bem definido?
- É definido através de conjuntos de atividades, onde contém ações e tarefas realizadas, tendo por objetivo a entrega de um ou mais artefatos (pacote de entrega).
- Cada uma dessas atividades, ações e tarefas se alocam dentro de uma metodologia ou modelo que determina sua relação com o processo de uma com as outras;
- Fluxo de processo: é um diagrama usado para representar graficamente um processo de forma simplificada.

Metodologia de software -> Atividades de apoio

Modelo de processos:
- Linear;
- Evolucionário;
- Paralelo;

Obs: Projetos diferentes exigem processos diferentes.

Exercício:
[Para a próxima aula - Capitulo 4 do livro Engenharia de Software]
Em dupla, criar um resumo para comentar na próxima aula, para o professor passar o capitulo 5;
Anotar como definir um projeto ideal para o projeto;

Como avaliar processos
- A existência de um processo não garante que o produto será entregue dentro do prazo, de acordo com os requisitos.
- Os processos devem ser avaliados para que esteja de acordo com um conjuntos de regras;

Avaliação de processo:

- **CMMI:** modelo de referência para saber se o projeto está maduro o suficiente;
- **ISSO 9001:2015**: Indica padrões de documentação, padrão de entregas, faz a empresa seguir um processo, controlar versões de documentos;

[Aula 4 dia 03/09/2019 ----------]

Link do Livro Online
- https://books.google.com.br/books?id=wexzCwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Como enviar a resenha:
DSLE01 - Nome da Dupla
Enviar até Terça-Feira(10/09) até 12h
E-mail: danilo.graca@etep.edu.br

Manifesto Ageis
- Em meio a um cenario de crescimento da demanda de software e a imprevisibilidade inicial identificada, um grupo ("Agile Alliance") criou um manifesto de "oposição";
- O manifesto agil propoe mudanças as abordagens tradicionais e oferece beneficios importantes;
- Não pratica a antese da tradicional engenharia de software, mas apresenta uma nova filosofia;
- A nova proposta é ser AGIL, atender a novas mudanças ao ponto que elas acontecem.

Rup - Processo de desenvolvimento de software, onde tem muito documento e não é agil. O RUP usa a abordagem da orientação a objetos em sua concepção e é projetado e documentado utilizando a notação UML (*Unified Modeling Language*) para ilustrar os processos em ação. Utiliza técnicas e práticas aprovadas comercialmente.

MÉTODO AGIL **MUDANÇAS**
RAPIDAS **ENTREGAS FRACIONADAS**

MÉTODO AGIL: Está ligado a atender mudanças onde se tem impactos minimos.

O MANIFESTO AGIL

Premissas:
Indivíduos e interações > processo e ferramentas
- Valoriza a interação da equipe

Software Operacional > Documentação completa
- Valoriza entregar um software que funciona do que uma documentação completa

Colaboração dos clientes > Negociação contratual

- Cliente participa do desenvolvimento

Respostas e mudanças > Seguir um plano

- Ter condições de fazer mudanças sem afetar o desempenho

Métodos ágeis:

O que é agilidade?

- Efetiva a resposta a mudanças;
- Efetiva comunicação entre todos Stakeholders;
- Projetar o cliente para dentro as equipe;
- Organizar uma equipe para que ela controle o trabalho realizado;
- Produzindo rápido e incrementalmente entregas de software;

Ambiente Complexo, instável e desafiador Método não-prescritivo

O método ágil veio para fazer frente a mudanças que ocorrem a todo momento;

Processo ágil:

Um processo ágil reduz o custo de alterações porque o software é entregado de forma incremental;

- Guiado pelas descrições do cliente sobre o que é requerido;
- Reconhece que planos tem vida curta;
- Desenvolvimento com ênfase em atividades de codificação, a ênfase é entregar o produto;
- Entrega de múltiplos incrementos de software;
- Adaptável conforme as mudanças vão ocorrendo;
- Requisitos e prioridades do cliente mudam o tempo todo;
- Projeto e construção intercalados;
- Processo capaz de administrar a imprevisibilidade;
- Entrega + Feedback = mudanças;
- Incrementos entregues em curtos períodos do tempo;

Software funcional é importante, mas não se deve esquecer que também deve
Apresentar uma série de atributos de qualidade, incluindo confiabilidade, Usabilidade e facilidade de manutenção.

Diferenças

Tradicionais	Método Ágil
Prazos mais longos; Previsibilidade de riscos; Noção clara dos requisitos; Documentação previa; Foco na aplicação de técnicas e ferramentas	Prazos curtos; Atuação constante sobre riscos; Pouca noção dos requisitos; Documentação obtida ao longo do desenvolvimento Foco na interatividade das pessoas;

Métodos híbridos
Usa-se conceitos dos dois lados da moeda.

Extreme Programming – XP

- Abordagem ágil mais utilizada;
- Envolve um conjunto de regras e conceitos associados a cada uma das atividades metodológicas;

1. **Planejamento**
2. **Projeto**
3. **Codificação**
4. **Teste**

Programação em pares – quando tem um problema muito grande, se programa em duas pessoas;
Refatoração de código - melhorar o código;

Planejamento:

- Início com a criação de Estória de Usuários;
- Time avaliar as estórias e define um custo (Poker sequência de Fibonacci)
- Estórias agrupadas para definir uma entrega incremental;

Projeto:

- Seguir o princípio KISS;
- Estimula o uso dos cartões CRC;
- Para problemas usa-se protótipo;
- Estimula a refatoração

Codificação:

- Recomenda a construção de testes unitários para estórias antes do inicio da codificação;
- Estimula a programação em pares;

Testes:

- Todos os testes unitários são executados diariamente (integração continua)
- Testes de aceitação são definidos pelo cliente e executados para avaliar a funcionalidade visivel ao usuário;

SCRUM

- Desenvolvimento do trabalho é particionado em pacotes;
- Testes e documentação são desenvolvidos juntos com o produto;
- O trabalho ocorre em Sprints e são derivados de um backlog de requisitos ao cliente

Perguntas que são feitas nas reuniões diárias:

O que você realizou deste a última reunião?
Você está tendo alguma dificuldade?
O que você vai fazer antes da próxima reunião?

Product Owner(PO): responsável pelo ROI(retorno de investimento), conhece as necessidades dos clientes;
Quebra o projeto em estórias

Scrum Master: É o responsável por garantir que os valores e práticas do **Scrum** estejam vivos dentro do dia-a-dia do time. Ele é considerado o coach do time, ajudando o Time **Scrum** a fazer o melhor trabalho possível.

Time de desenvolvimento: é o time que irá estar ligado ao desenvolvimento do software;

Gráfico BurnDown

- Facilita o acompanhamento
- Quanto ainda falta para entregar

Azul: O que deveria acontecer
Laranja: Onde está o projeto dentro da Sprint

[Dia 17/09]

Fases do projeto
- requerimento
- análises
- design
- implementação
- teste
- suporte

Curva de gerenciamento de projeto
Acompanhamento do projeto

Fases de um projeto
PCp- proposta (exemplo Canvas)
- concepção
- análise
- construção
- entrega

Em cada final de fase existe um ponto de controle de projeto

Gestão de riscos
- identifica e planeja a gestão dos riscos e oportunidades do projeto
- define respostas aos riscos e oportunidades, o que deve impactar as iniciativas iniciais do projeto como prazo, custos, ainda, qualidade.
- os riscos e oportunidades são constantemente monitorados juntos ao monitoramento do projeto

- Quando necessário, as respostas de mitigação ao riscos são executados, ou ainda não foram mesmo riscos.
Dando a oportunidade de não ocorrer.

-mitigar um risco: é diminuir o máximo possível a possibilidade de ocorrer o risco.

Gestão de configuração
- A gestão de configuração permite controlar os produtos de trabalho quanto a integridade de suas versões e a correta entrega do produto.
- reduz a perda de trabalho por meio de uma ferramenta de controle de versão, com cópia em outro local.
- monitora e controla as mudanças em produtos de trabalho, avaliando seu impacto em outros produtos

Controle de versão
- permite recuperar versões passadas e permite verificar quem faz determinada alteração
- pode ser dois tipos: informal e formal (informal não precisa de autorização para mudança, enquanto o formal precisa)
- o tipo de cada item é definido no início do projeto e após determinada etapa os itens passam de controle informal para formal.

Controle de mudança
- os itens definidos com controle formal precisam de uma avaliação de impacto quando precisam ser alterados.
- avaliação de impacto de haverá se houver a mudança.

Baseline
- são pontos do projeto onde no podem ser alterados.
- alteração somente com o processo de mudança.
- as baselines devem ter nomes que a identifique.

Garantia da qualidade
- a garantia da qualidade identifica problemas através de auditorias em produtos de trabalho e/ou nos próprios processos.
- a garantia da qualidade permite r qualidade das entregas, melhor desempenho do projetor, sobretudo maior a satisfação do cliente.
- a garantia da qualidade é confundida com a qualidade de produto, como por exemplo, a aplicação dos testes em software.

[Dia 24/09 Revisão para a prova]

Aula 1

- **O que é software?**
  É um conjunto de componentes lógicos; estrutura de dados; documentação;
- **Porque documentação é importante para desenvolver um software?**
  Para poder passar o conhecimento do software para programadores futuros. Para manutenção.
- **Porque o software se deteriora?**
  Software não se desgasta, se deteriora. Se houver muitas mudanças pode aparecer bugs.
- **Diferença do hardware para o software?**
  O hardware(curva da banheira) se desgasta e o software(ciclo de vida do software) não.
- **Grande busca por software personalizado (custom-build)**
- **O que é software legado?**
  É um software escrito com uma tecnologia obsoleta, mas funciona perfeitamente. Pode passar por atualizações de tecnologias de desenvolvimento.
- **Quais 3 tipos de serviços Cloud Computing?**
  Infraestrutura como serviço; software como serviço; e plataforma como serviço.
- **ENGENHARIA DE SOFTWARE: Quais desafios encontrados hoje no desenvolvimento de software?**
  Cada vez mais complexo, demanda maior, Inter operável, cada vez mais funcional, projetar, maior qualidade e facilidade de manutenção;
- **Conceito de engenharia de software?**
  Uma disciplina sistemática, formada por processos.
- **Quais as camadas de engenharia de software?**

  Foco na qualidade: Processo, Métodos e ferramentas.
- **Definição das 3 camadas da engenharia de software:**
  Processos: estabelece as diretrizes para o desenvolvimento e controle do projeto. Ou seja é conjunto de coisas que tem na empresa;
  Exe: Requisitos, testes, arquiteturas;
  Métodos: fornecem informações técnicas para desenvolvimento e envolvem uma ampla variedade de tarefas;
  Ferramentas: Suporta automatizado ou semi-automatizado para o desenvolvimento de atividades e o proprio processo;

Aula 2

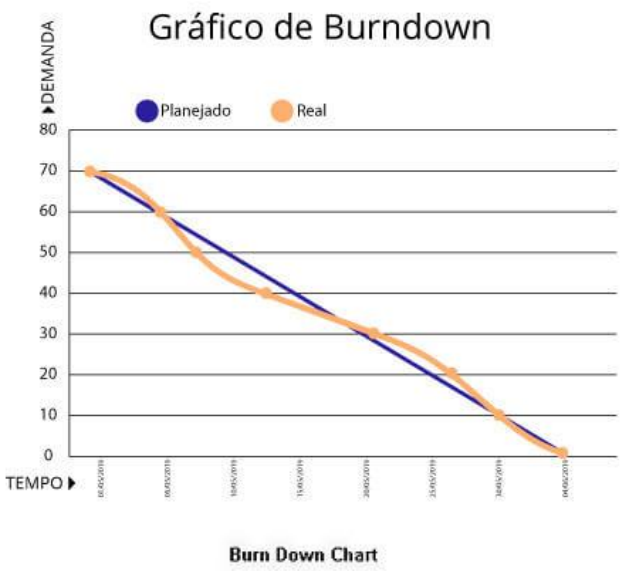
- **O que é processo?**
  É um roteiro que ajuda a criar resultado com alta qualidade dentro do prazo estabelecido.
- **Por que é importante usar processo?**
  Conhecer e institucionalizar o fluxo de trabalho; Padronizar atividade e tarefas; Documentar o conhecimento; Identificar oportunidades de melhorias; Definir papeis e responsabilidades; Melhorar a qualidade do produto;
- **Como definir o processo de uma empresa?**
  Existe vários tipos de modelos que pode ser moldado para se encaixar da melhor forma possível dentro da empresa.
- **Estrutura de um processo?**
  Deve conter Começo, meio e fim, onde o fim deve conter algo entregável.

Aula 3

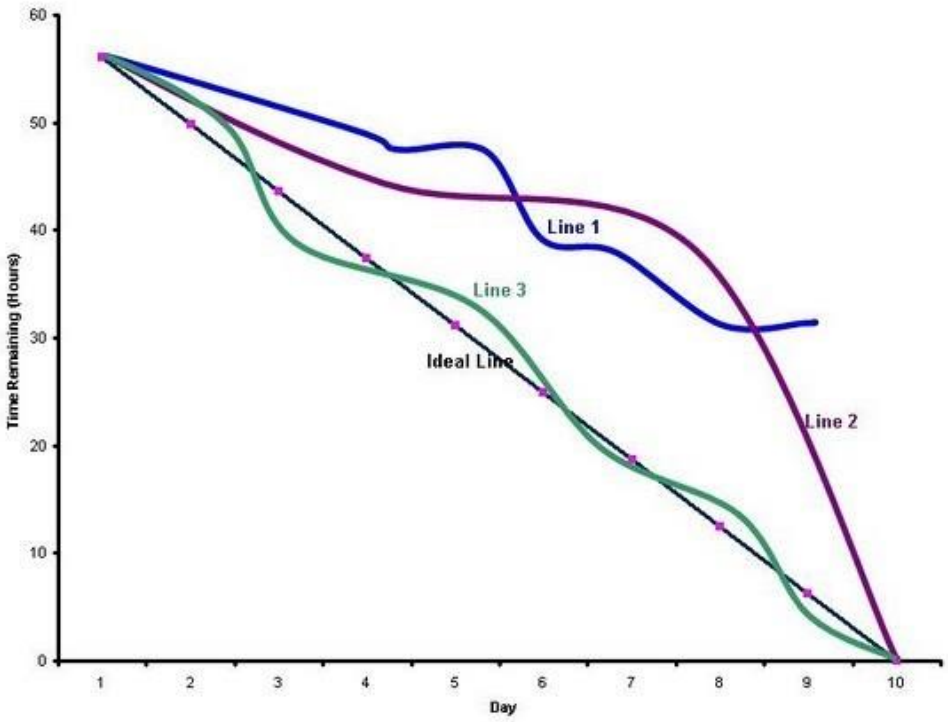
- **Manifesto ágil, o que é?**
  O manifesto ágil propõe o desenvolvimento ágil. O método ágil veio para fazer frente a mudanças que ocorrem a todo momento; Estar preparado a mudanças.
- **O que é agilidade?**
  É responder a mudanças de forma rápida; ter comunicação entre a equipe. Projetar o cliente para dentro da equipe; Organizar a uma equipe que se auto gerencia; Produção rápida e incrementavel em entregas de software(entregas em 15/15 dias por exemplo)
- **Diferenças método tradicional vs ágil**

Tradicionais	Método Ágil
Prazos mais longos; Previsibilidade de riscos; Noção clara dos requisitos; Documentação previa; Foco na aplicação de técnicas e ferramentas	Prazos curtos; Atuação constante sobre riscos; Pouca noção dos requisitos; Documentação obtida ao longo do desenvolvimento Foco na interatividade das pessoas;

- **Metodologias ágeis:**
  Xp: Envolve um conjunto de regras e conceitos associados a cada uma das atividades metodológicas;
  Planejamento, Projeto, Codificação e Teste.
  SCRUM: Passa a priorizar os requisitos, que são quebrados em pequenas atividades que são chamados de Sprints para ter uma entrega. Se baseia em entregas constantes.
- **Gráfico Burndown**



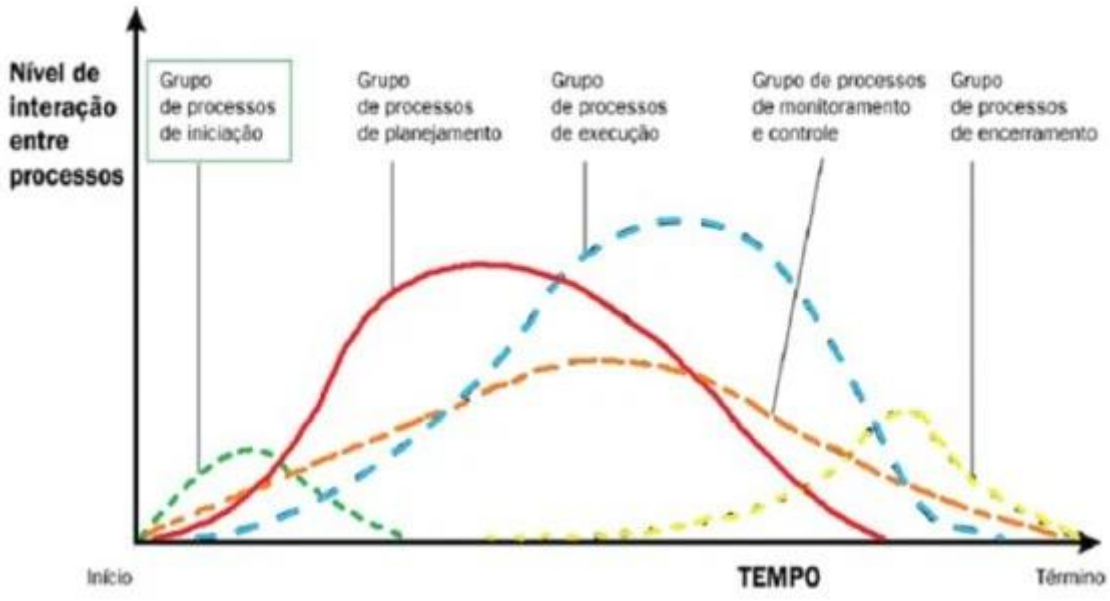
Burn Down Chart



A Linha 1 (Azul) nunca atinge o zero, o que indica que o planejamento e execução do time não estão bons, existindo vários motivos, como alteração de prioridade durante o sprint, planejamento de mais histórias do que a capacidade ou até mesmo uma falta de união no time. Além disso, indica que é necessário melhorar o planejamento.
A Linha 2 (Roxa) atingiu a meta, mas esteve longe da linha ideal e no meio uma curva acentuada o que pode indicar que o time não atualizou suas estimativas corretamente ou histórias incompletas foram retiradas do sprint.
A Linha 3 indica que o planejamento e execução foram bons. O time é auto-organizado, diz se a meta foi atingida da mesma forma nos últimos sprints e não deixa claro melhorias, mas sempre existem melhorias no planejamento.

Aula 4

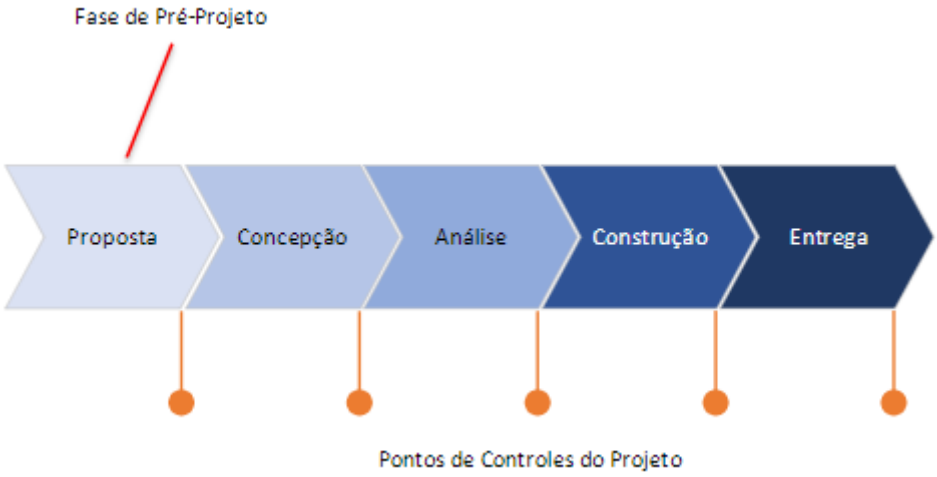
- **O que é método prescritivo?**
  Concentra-se em estruturar e ordenar o desenvolvimento de software com diretrizes de processos definidos.
- **Modelo Cascata?**
  Finaliza uma fase e passa para a próxima. Pode passar para outra fase devendo algo.
- **Diagrama em V?**
  É igual ao cascata, segue uma ordem sequencial, mas é mais focado em teste.
- **Modelo incremental?**
  Poucas mudanças, é como um cascata divido em pedaços. O objetivo nem sempre é entregar um produto funcional.
- **Modelo evolutivo - protótipo**
  Requisitos identificados, mas não detalhados; Aplicável junto com outro modelo; Protótipo auxilia a entender os requisitos.
- **Modelo evolutivo – espiral**
  Voltado a projetos complexos; considera riscos; requisitos parcialmente conhecidos; entregas funcionais; abordagem realista;
- **Monitorar o projeto:** gestão de projetos sempre ativo.
- **Fases do projeto:**
  - requerimento
  - análises
  - design
  - implementação
  - teste
  - suporte
- **Ciclo de grupo de projetos**

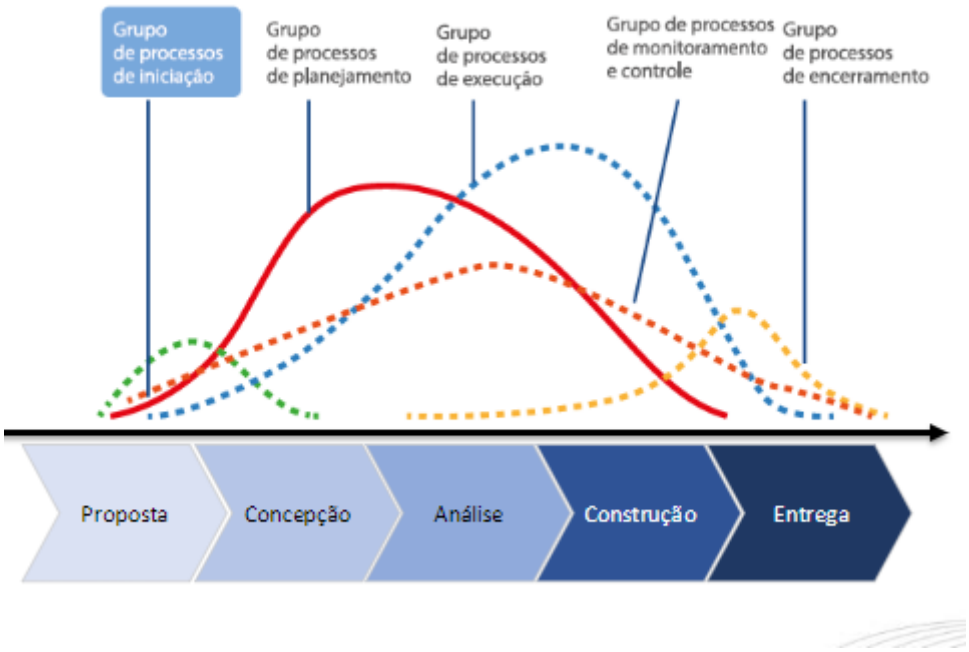


- **O que são métodos híbridos?**
  É uma combinação de vários métodos que é moldado para se encaixar da melhor forma possível dentro da empresa.

Aula 5

- **Gerenciamento de projetos**





- **O que é gestão de risco?**
  Identifica e planeja a gestão dos riscos e oportunidade do projeto; Define respostas aos riscos e oportunidades, o que deve impactar as iniciativas iniciais do projeto como prazo, custo e qualidade.
- **O que é gestão de configuração?**
  A Gestão de Configuração permite controlar os produtos de trabalho quanto a integridade de suas versões e a correta entrega do produto;
  Reduz a perda de trabalho por meio de uma ferramenta de controle de versão com a cópia em outro local.
  Monitora e controla as mudanças em produtos de trabalho, avaliando seu impacto em outros produtos.
- **Controle de versão?**
  Uso de uma ferramenta de versionamento
- **Controle de mudança?**
  Avaliação de impacto busca definir o impacto em outros produtos de trabalho, prazo, custo e outras demandas.
- **O que é baseline?**
  São pontos de congelamento do software/projeto. Sempre que houver uma baseline é obrigatório passar por gestão de mudanças para garantir a integridade do software.
- **Garantia da qualidade?**
  É garantir que o projeto em si está feito com qualidade. Garantir a qualidade apenas dos documentos, sem olhar a parte técnica.