

Processo de Desenvolvimento de Software

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br

Histórico (era pré-ES)

- 1940s: Primeiro computador eletrônico de uso geral – ENIAC
 - Custo estimado de US\$ 500.000,00
 - Início da programação de computadores
- 1950s: Primeiros compiladores e interpretadores
- 1960s: Primeiro grande software relatado na literatura – OS/360
 - Mais de 1000 desenvolvedores
 - Custo estimado de US\$ 50.000.000,00 por ano
- 1968: Crise do software – nasce a Engenharia de Software

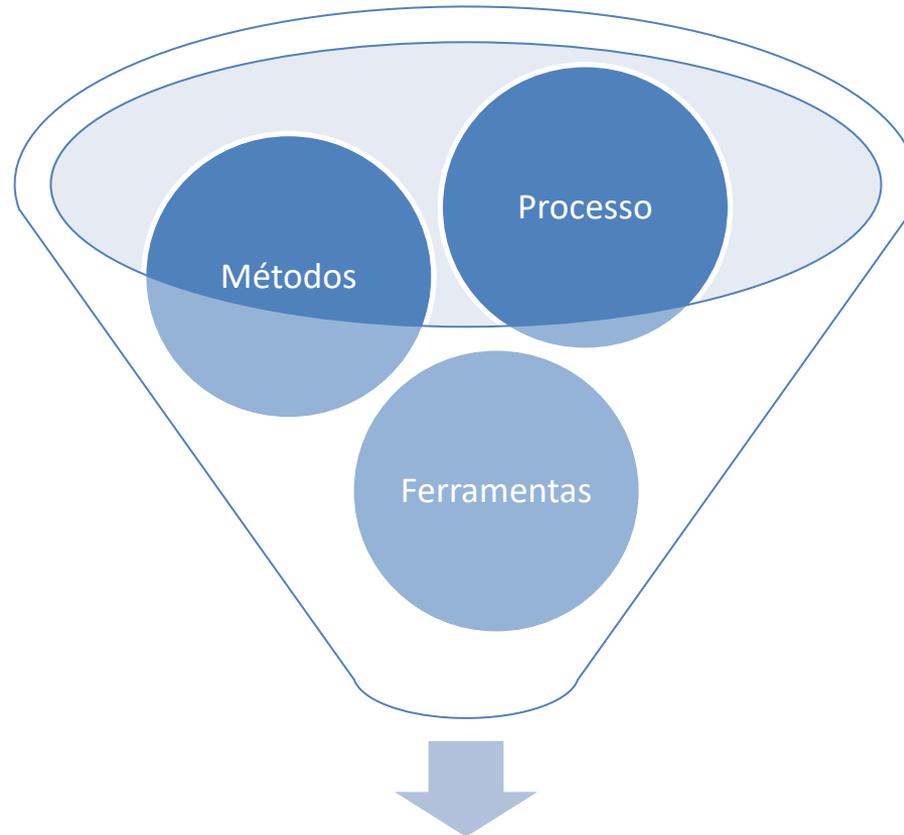
Histórico (era pós-ES)

- 1970s:
 - Lower-CASE tools (programação, depuração, colaboração)
 - Ciclo de vida cascata
 - Desenvolvimento estruturado
- 1980s:
 - Ciclo de vida espiral
 - Desenvolvimento orientado a objetos
 - Controle de versões
 - Testes
- 1990s: Upper-CASE tools
 - Processos
 - Modelagem

Histórico (era pós-ES)

- 2000s:
 - Métodos ágeis
 - Desenvolvimento dirigido por modelos
 - Linhas de produto
 - Experimentação
- Atualmente
 - DevOps
 - Continuous*
 - Software Analytics
 - ...

Elementos da ES



Engenharia de Software

Elementos da ES

- Processo
 - Define os passos gerais para o desenvolvimento e manutenção do software
 - Serve como uma estrutura de encadeamento de métodos e ferramentas
- Métodos
 - São os “how to’s” de como fazer um passo específico do processo
- Ferramentas
 - Automatizam o processo e os métodos

Elementos da ES

- Cuidado com o “desenvolvimento guiado por ferramentas”
 - É importante usar a ferramenta certa para o problema
 - O problema não deve ser adaptado para a ferramenta disponível



“Para quem tem um martelo, tudo parece prego”

Elementos da ES

1. Coloque em uma panela funda o leite condensado, a margarina e o chocolate em pó.
2. Cozinhe [no fogão] em fogo médio e mexa sem parar com uma colher de pau.
3. Cozinhe até que o brigadeiro comece a desgrudar da panela.
4. Deixe esfriar bem, então unte as mãos com margarina, faça as bolinhas e envolva-as em chocolate granulado.

O que é
processo,
método ou
ferramenta?

<http://tudogostoso.uol.com.br/receita/114-brigadeiro.html>

Elementos da ES

1. **Coloque** em uma **panela** funda o leite condensado, a margarina e o chocolate em pó.
2. **Cozinhe** [no **fogão**] em fogo médio e **mexa** sem parar com uma **colher de pau**.
3. **Cozinhe** até que o brigadeiro comece a desgrudar da **panela**.
4. Deixe esfriar bem, então **unte** as mãos com margarina, **faça as bolinhas** e **envolva**-as em chocolate granulado.



método



ferramenta

Processo

<http://tudogostoso.uol.com.br/receita/114-brigadeiro.html>

O Supermercado de ES

- ES fornece um conjunto de métodos para produzir software de qualidade
- Pense como em um supermercado...
 - Em função do problema, se escolhe o processo, os métodos e as ferramentas
- Cuidado
 - Menos do que o necessário pode levar a desordem
 - Mais do que o necessário pode emperrar o projeto

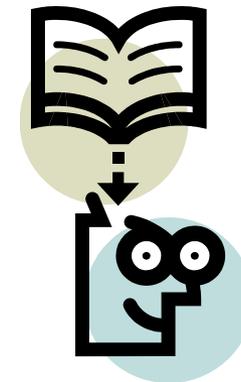
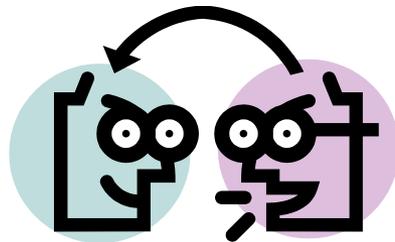


Exercício

- Problema
 - Definir o procedimento de implantação para os dois cenários a seguir
- Caso 1: urna eletrônica
 - O software da urna eletrônica acabou de ser implementado, e precisa ser instalado em 480 mil urnas
- Caso 2: padaria
 - O software de controle de venda de pão da padaria do seu Zé acabou de ser implementado, e precisa ser instalado

Processos implícitos x explícitos

- Lembrem-se: Processos sempre existem, seja de forma implícita ou explícita!
 - Processos implícitos são difíceis de serem seguidos, em especial por novatos
 - Processos explícitos estabelecem as regras de forma clara



Processo de qualidade

- Última palavra para medir a qualidade de um processo: **Satisfação do Cliente**
- Outros indicadores importantes
 - Qualidade dos produtos gerados
 - Custo real do projeto
 - Duração real do projeto

Modelos de ciclo de vida

- Existem alguns processos pré-fabricados
 - Esses processos são conhecidos como modelos de ciclo de vida
 - Esses processos apresentam características predefinidas
- Devem ser adaptados para o contexto real de uso
 - Características do projeto
 - Características da equipe
 - Características do cliente

Exercício

- Assuma as atividades básicas de todo processo como sendo
 - Comunicação
 - Planejamento
 - Modelagem
 - Construção
 - Implantação
- Projete um processo que determina a ordem com que cada uma dessas atividades é executada
- Quais as características positivas ou negativas desse processo?

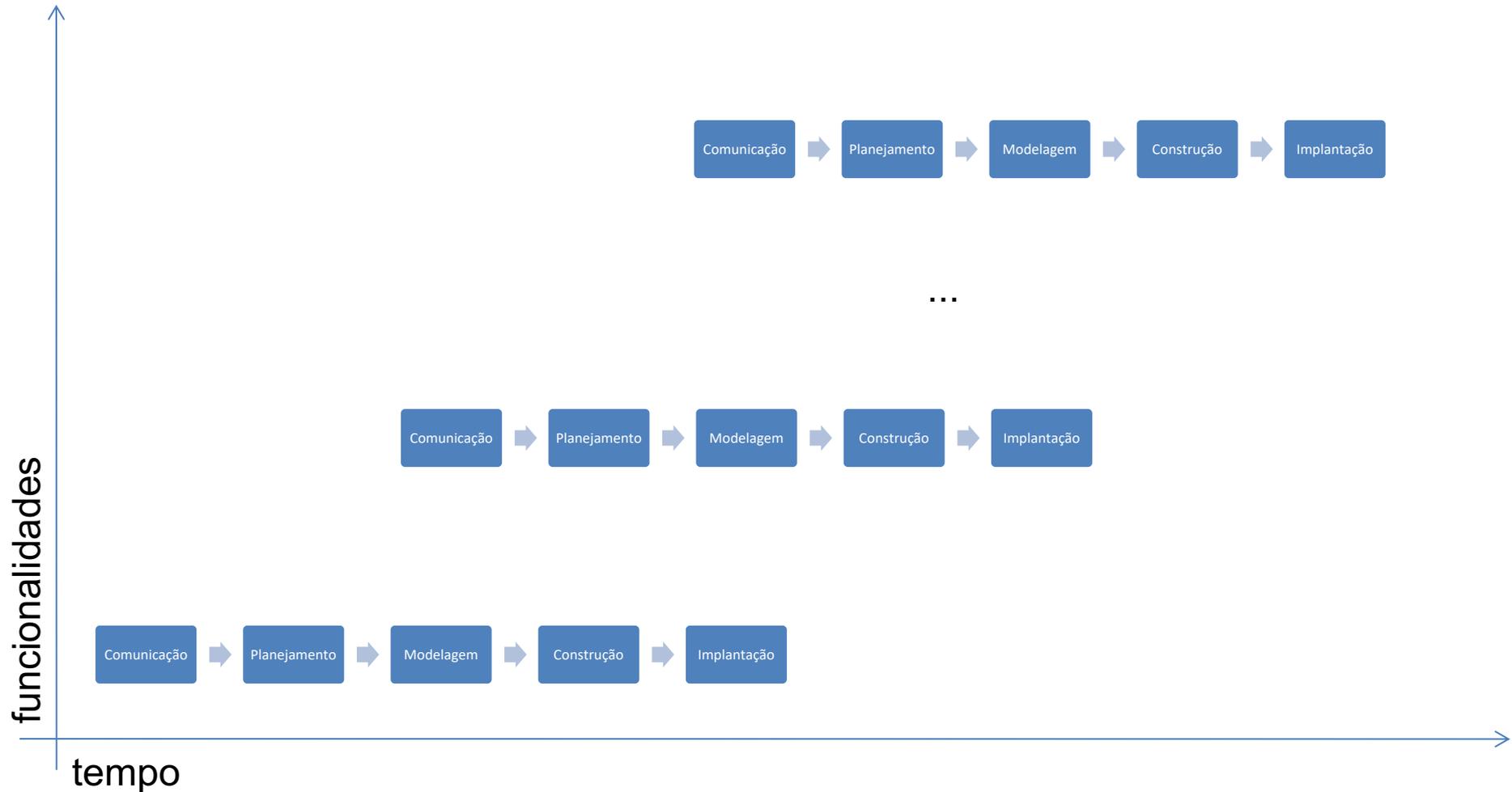
Ciclo de vida Cascata



Ciclo de vida Cascata

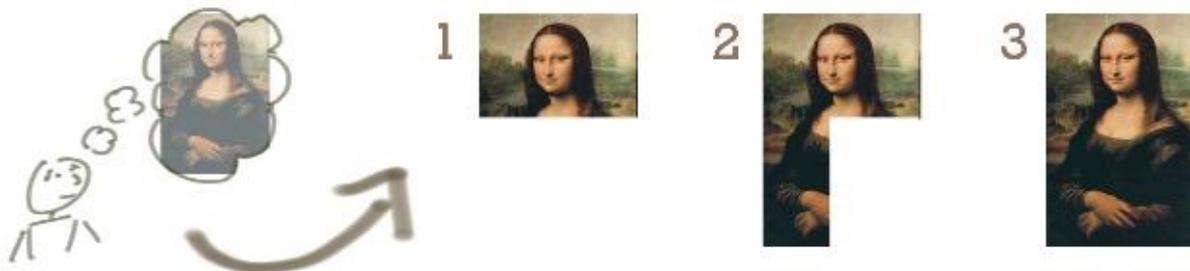
- Útil quando se tem requisitos estáveis e bem definidos
 - Ex.: Adicionar um novo dispositivo legal em um sistema de contabilidade
- Não lida bem com incertezas
- Fornece pouca visibilidade do estado do projeto
 - Muito tempo para a primeira entrega
 - Dificuldade na obtenção de feedback do cliente

Ciclo de vida Incremental



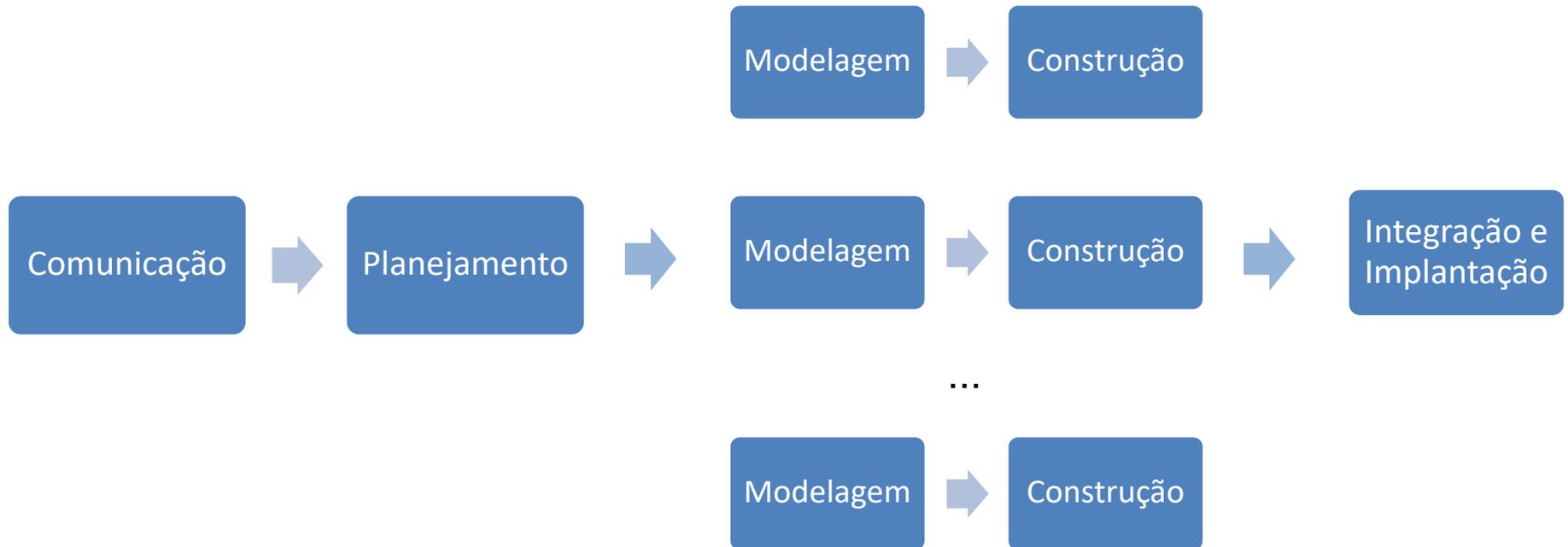
Ciclo de vida Incremental

- Faz entregas incrementais do software
 - Cada incremento é construído via um mini-cascata
 - Cada incremento é um software operacional
- Versões anteriores ajudam a refinar o plano
 - Feedback constante do cliente
- Diminuição da ansiedade do cliente
 - O cliente rapidamente recebe uma versão funcional do software



Jeff Patton (2008)

Ciclo de vida RAD

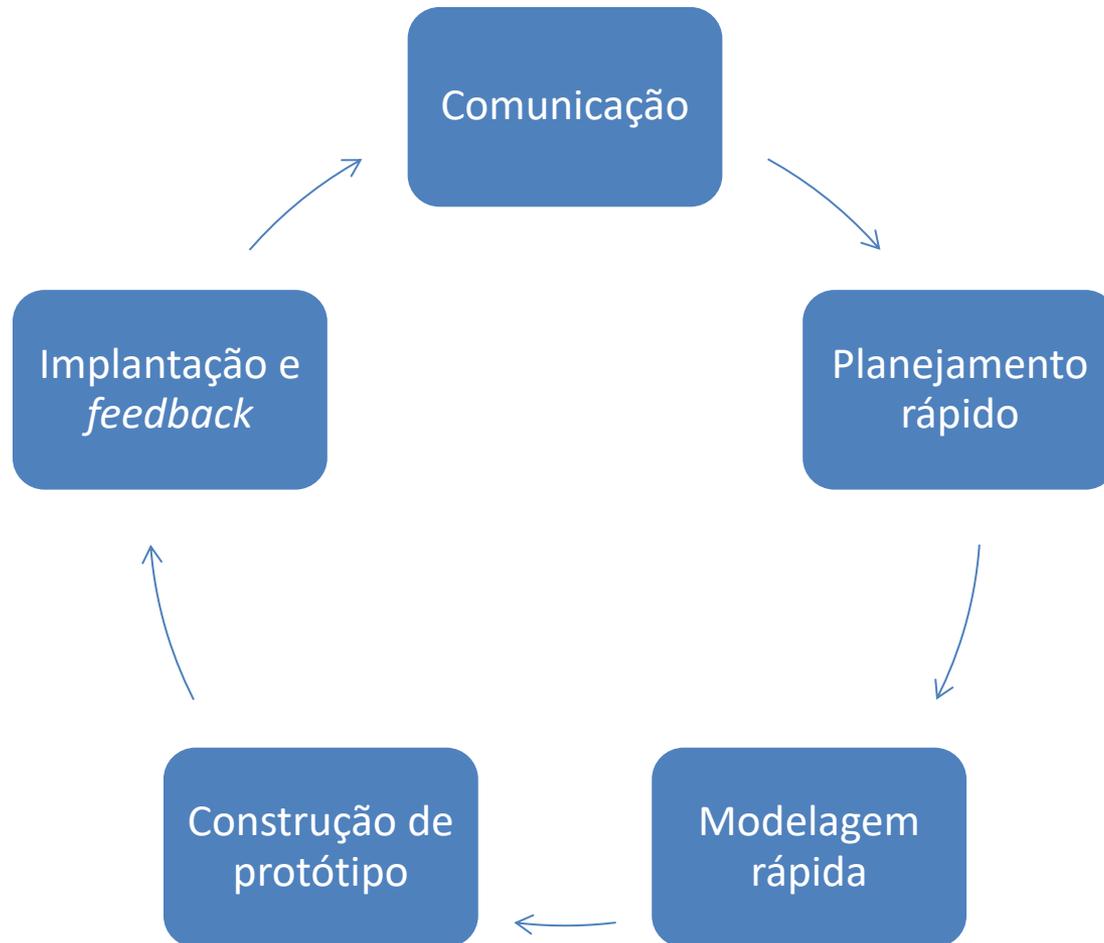


tempo

Ciclo de vida RAD

- Funcionamento equivalente ao cascata
- Principais diferenças
 - Visa entregar o sistema completo em 60 a 90 dias
 - Múltiplas equipes trabalham em paralelo na modelagem e construção
 - Assume a existência de componentes reutilizáveis e geração de código
- Difícil de ser utilizado em domínios novos ou instáveis

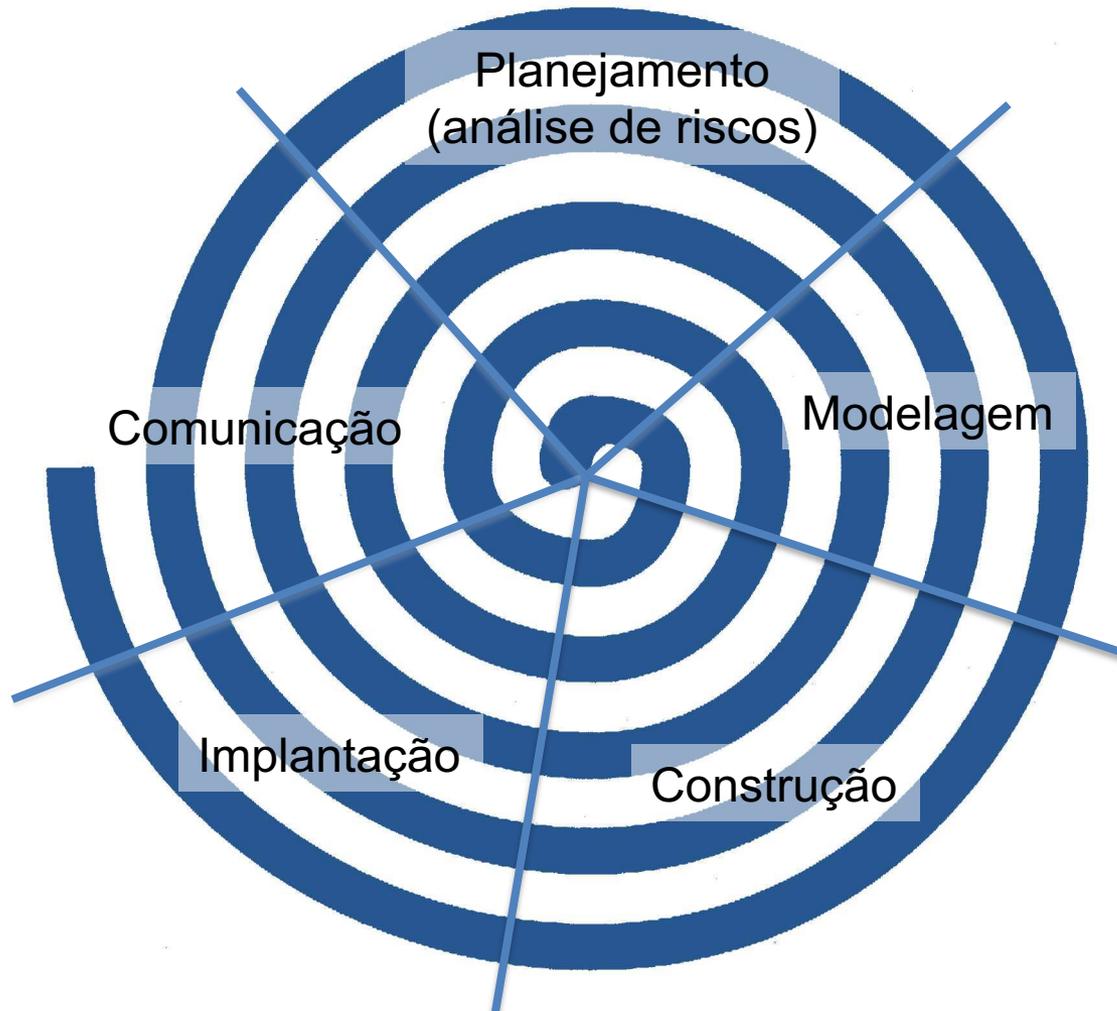
Prototipação



Prototipação

- Usualmente utilizado como auxílio a outro modelo de ciclo de vida
- Útil para
 - Validar um requisito obscuro com o cliente
 - Verificar o desempenho de um algoritmo específico
- Deveria ser jogado fora no final
 - Protótipos não são produtos
 - Usualmente os clientes desejam colocar protótipos em produção

Ciclo de vida Espiral



Ciclo de vida Espiral

- Foco principal no gerenciamento de riscos
- A cada ciclo
 - O conhecimento aumenta
 - O planejamento é refinado
 - O produto gerado no ciclo anterior é evoluído (não é jogado fora)
- Cada ciclo evolui o sistema, mas não necessariamente entrega um software operacional
 - Modelo em papel
 - Protótipo
 - Versões do produto
 - Etc.



Jeff Patton (2008)

Cascata x Evolutivo

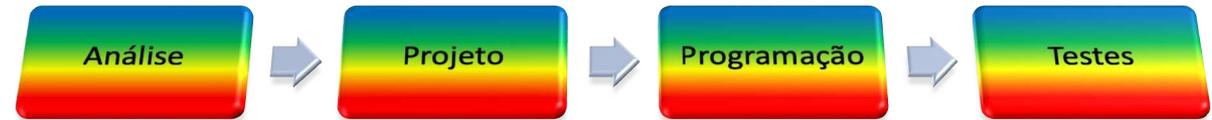


Cascata x Evolutivo

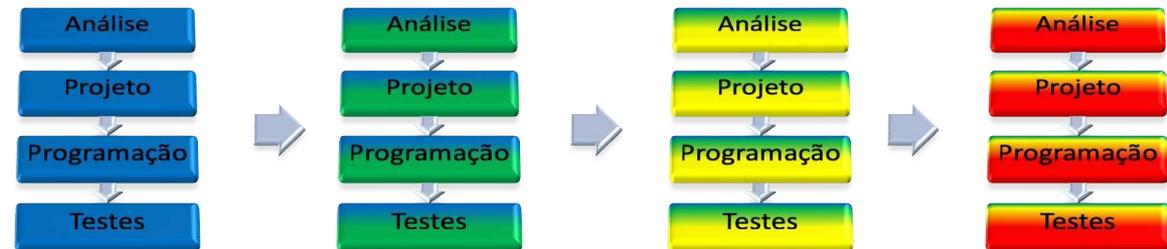


Cascata x Evolutivo

Ciclo de vida cascata



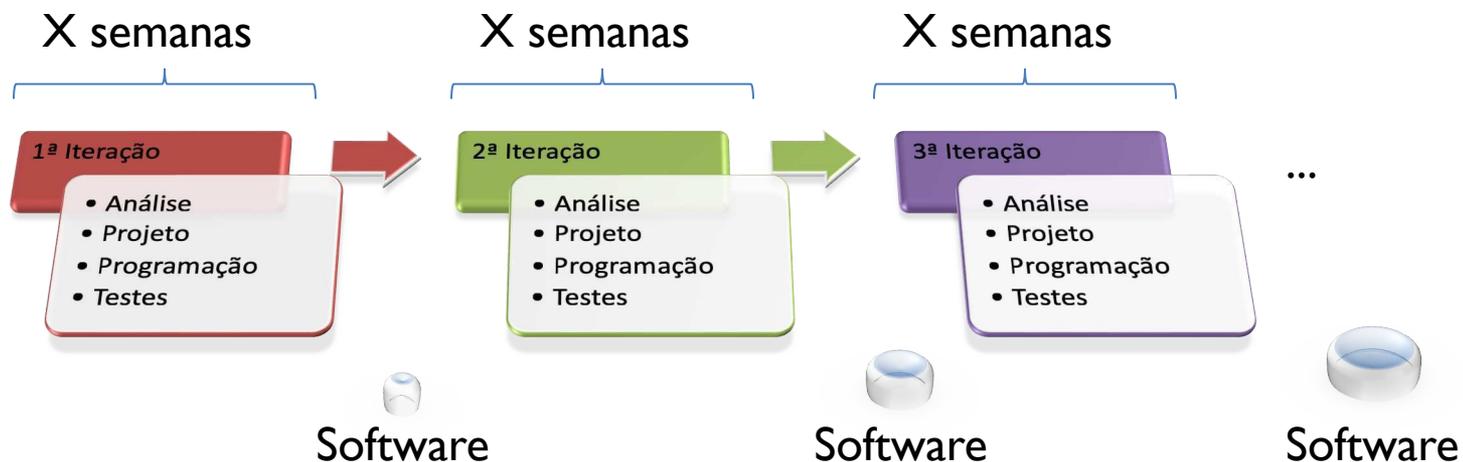
Ciclo de vida evolutivo



- Objetivo: Processo Unificado com aspectos de...
 - Desenvolvimento iterativo
 - Desenvolvimento evolutivo
 - Desenvolvimento ágil

Desenvolvimento Iterativo

- O desenvolvimento é organizado em “mini-projetos”
 - Cada “mini-projeto” é uma iteração
 - Cada iteração tem duração curta e fixa (de 2 a 6 semanas)
 - Cada iteração tem atividades de análise, projeto, programação e testes
 - O produto de uma iteração é um software parcial

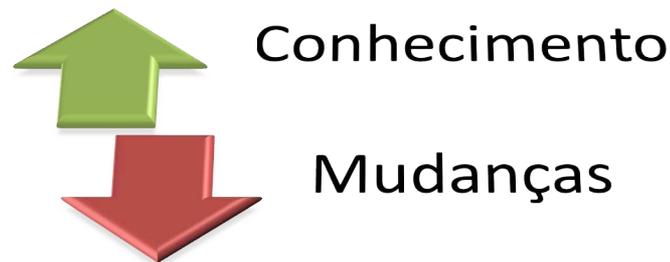
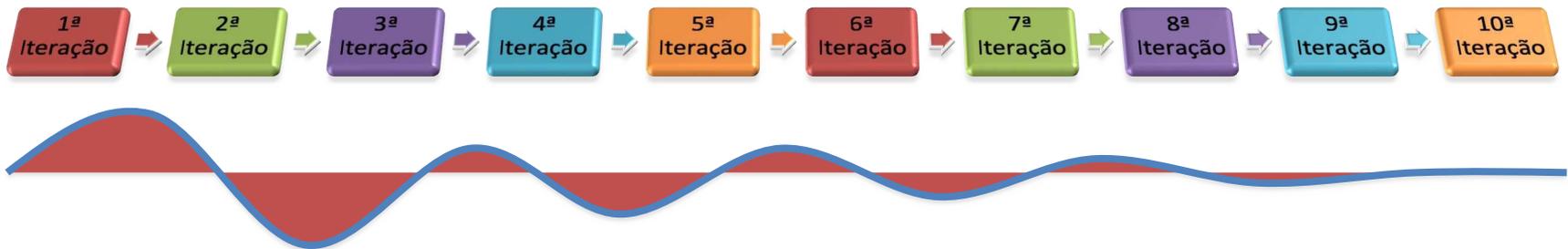


Desenvolvimento Iterativo

- A iteração deve ser fixa
 - Tarefas podem ser removidas ou incluídas
 - A iteração nunca deve passar da duração previamente estipulada
- O resultado de cada iteração é um software...
 - Incompleto
 - Em desenvolvimento (não pode ser colocado em produção)
 - Mas não é um protótipo!!!
- Esse software pode ser verificado e validado parcialmente
 - Testes
 - Usuários
- Podem ser necessárias diversas iterações (e.g. 10 a 15) para ter uma versão do sistema pronta para entrar em produção

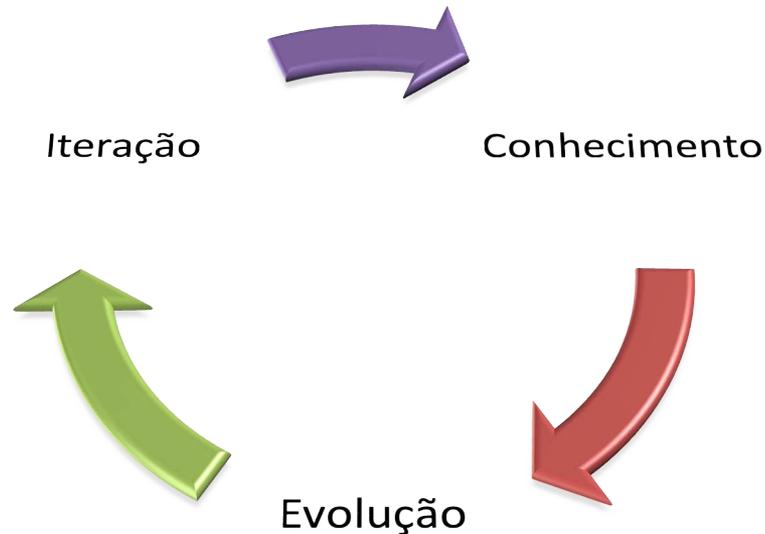
Desenvolvimento Iterativo

- Iterações curtas privilegiam a propagação de conhecimento
 - Aumento do conhecimento sobre o software
 - Diminuição das incertezas, que levam às mudanças



Desenvolvimento Evolutivo

- As especificações evoluem a cada iteração
 - A cada iteração, uma parte do software fica pronta
 - O conhecimento sobre o software aumenta
 - As especificações são evoluídas para retratar esse aumento de conhecimento sobre o que é o software



Desenvolvimento Evolutivo

- Mudanças sempre acontecem em projetos de software
 - Requisitos mudam
 - O ambiente em que o software está inserido muda
 - As pessoas que operam o software mudam
- Estratégias para lidar com mudanças
 - Evitar as mudanças (corretivas) fazendo uso de boas técnicas de engenharia de software
 - Acolher mudanças por meio de um processo evolutivo

Desenvolvimento Ágil

- São dadas respostas rápidas e flexíveis a mudanças
 - O projeto é replanejado continuamente
 - São feitas entregas incrementais e constantes do software, refletindo as mudanças solicitadas



Desenvolvimento Ágil

12 Princípios Ágeis

Satisfazer o cliente com software que agregue valor

Acolher modificações nos requisitos

Entregar o software funcional com frequência

Trabalhar junto ao cliente

Manter as pessoas motivadas e confiar nelas

Promover conversas face a face

Medir o progresso com software funcionando

Manter um ritmo constante de trabalho

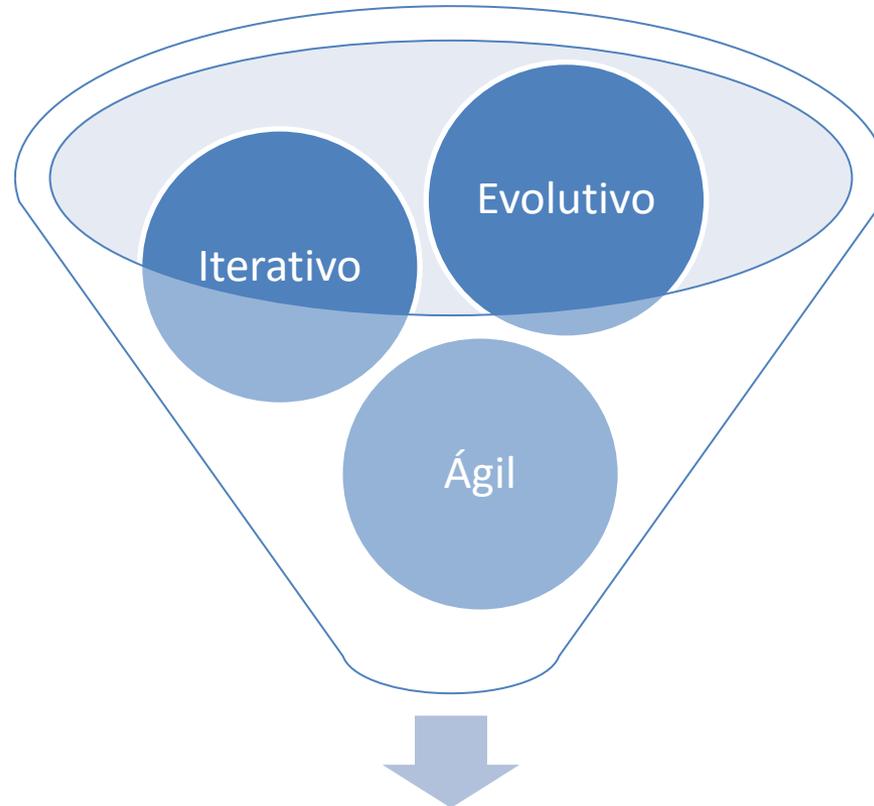
Prezar por excelência técnica

Buscar por simplicidade

Trabalhar com equipes auto-organizadas

Ajustar o comportamento da equipe buscando mais efetividade

Processo Unificado



Processo Unificado

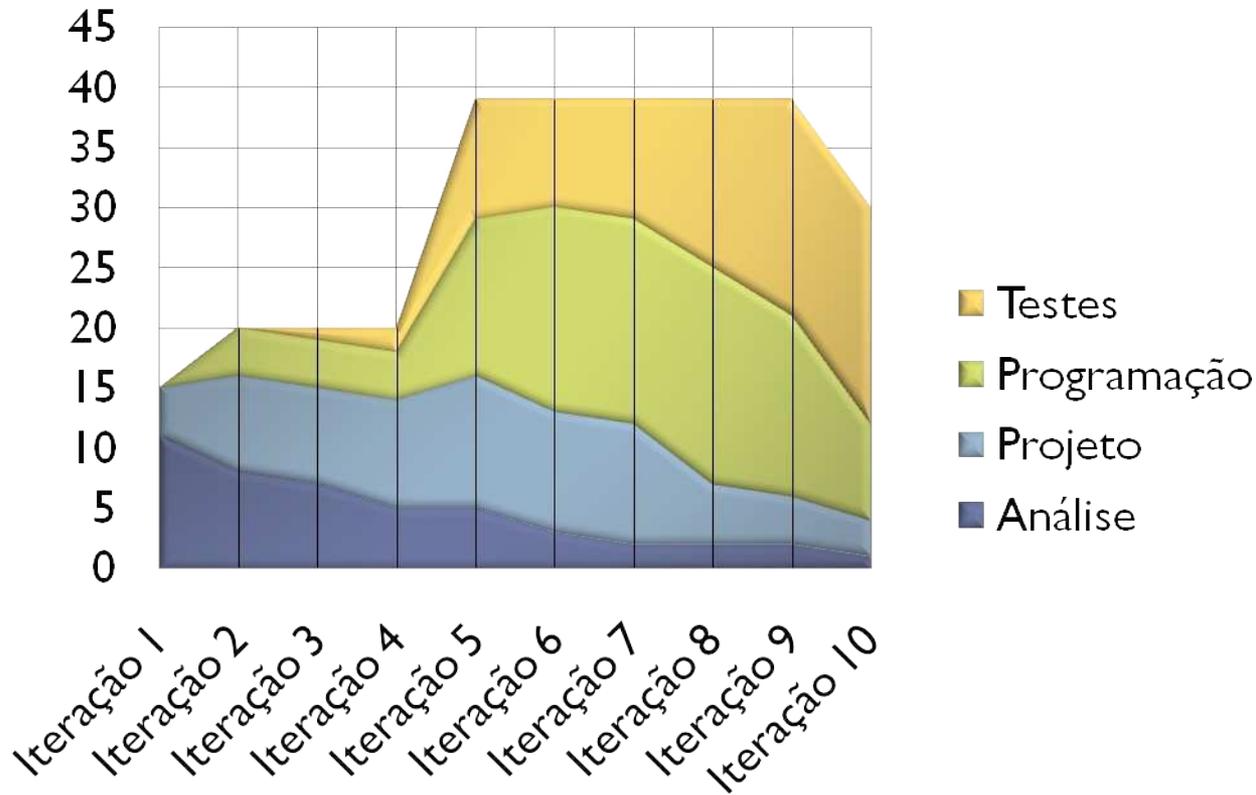
Processo Unificado (benefícios esperados)

- Mitigação de riscos precoce
- Visibilidade do progresso
- Envolvimento e comprometimento do usuário
- Controle sobre a complexidade
- Aprendizado incremental
- Menos defeitos
- Mais produtividade

Processo Unificado (exemplo)

- Analisar os requisitos no início do projeto
 - Casos de uso
 - Lista de requisitos não funcionais
- Priorizar os casos de uso
 - Significativos para a arquitetura como um todo
 - Alto valor de negócio
 - Alto risco
- Em cada iteração
 - Selecionar alguns casos de uso por ordem de prioridade para serem analisados em detalhes
 - Atribuir tarefas para a iteração a partir da análise detalhada desses casos de uso
 - Fazer projeto e programação de parte do software
 - Testar a parte do software recém projetada e programada e criar a *baseline* da iteração
 - Apresentar a *baseline* da iteração ao usuário

Processo Unificado (exemplo)

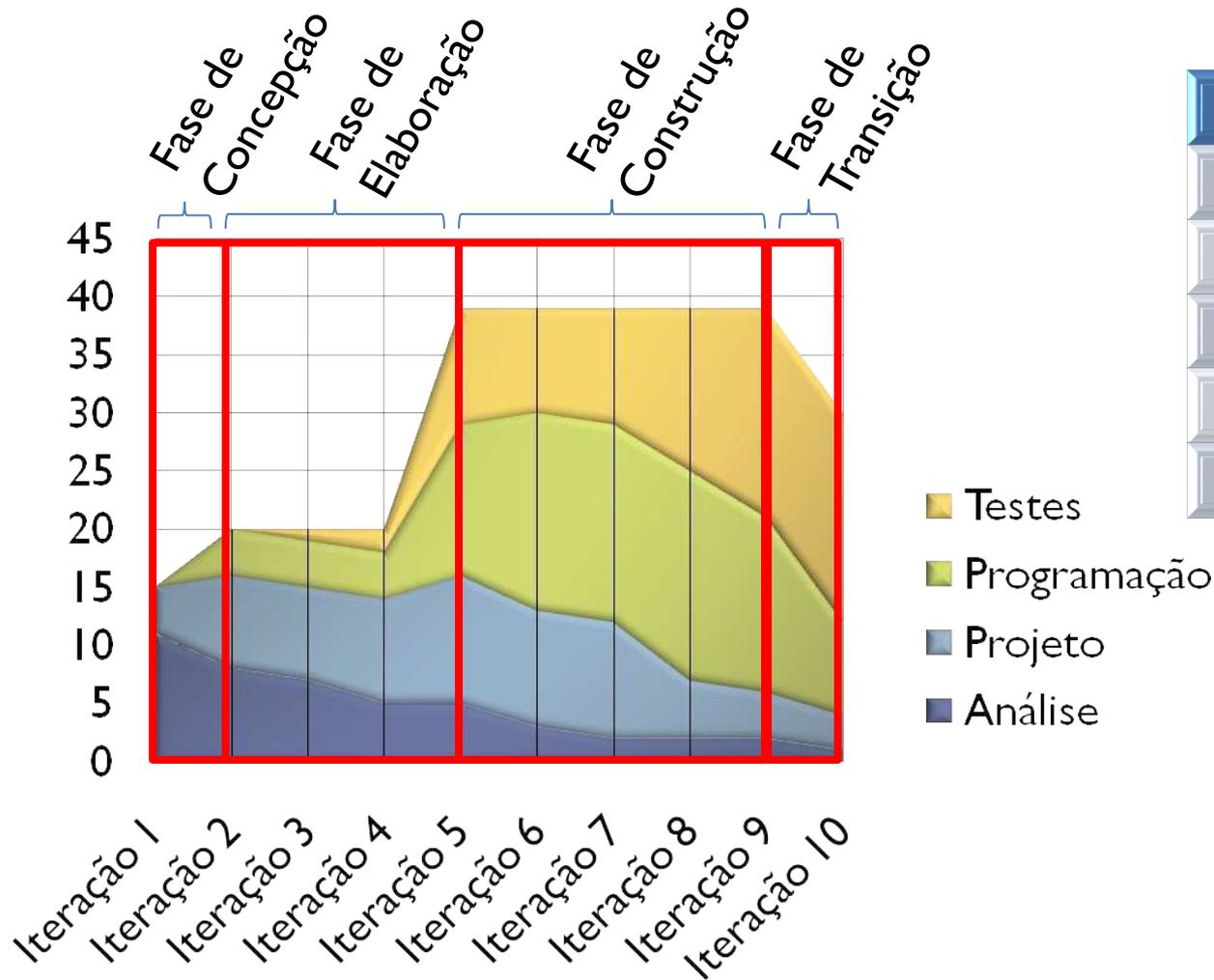


- Testes
- Programação
- Projeto
- Análise

Processo Unificado (fases)

- O desenvolvimento pode ser decomposto em fase, com o intuito de retratar a ênfase principal das iterações
 - Concepção
 - Elaboração
 - Construção
 - Transição
- Plano da fase
 - Abrangente e superficial
- Plano da iteração
 - Específico e detalhado

Processo Unificado (exemplo)



Atividade	Esforço
Análise	10%
Projeto	15%
Programação	30%
Testes	15%
Gerência	30%

Processo Unificado (concepção)

- Consiste de
 - Identificação de riscos
 - Listagem inicial dos requisitos
 - Esboço dos casos de uso
 - Identificação de arquiteturas candidatas
 - Estimativas iniciais de cronograma e custo
- Principais características
 - Menor fase do projeto
 - Escopo ainda vago
 - Estimativas ainda vagas
- Esforço e duração aproximados
 - 5% do esforço do projeto
 - 10% da duração do projeto

Processo Unificado (elaboração)

- Consiste de
 - Mitigação dos riscos
 - Detalhamento da maioria dos requisitos e casos de uso
 - Estabelecimento e validação da arquitetura do software
 - Detalhamento das estimativas de cronograma e custo
- Principais características
 - Grande parte das atividades de análise e projeto já concluída
 - Diminuição significativa das incertezas
 - *Baseline* da arquitetura é estabelecida
- Esforço e duração aproximados
 - 20% do esforço do projeto
 - 30% da duração do projeto

Processo Unificado (construção)

- Consiste de
 - Implementação dos demais componentes da arquitetura
 - Preparação para a implantação
- Principais características
 - Maior fase do projeto
 - *Baseline* de testes do produto é estabelecida
- Esforço e duração aproximados
 - 65% do esforço do projeto
 - 50% da duração do projeto

Processo Unificado (transição)

- Consiste de
 - Execução de testes finais
 - Implantação do produto
 - Treinamento dos usuários
- Principais características
 - *Baseline* de liberação do produto é estabelecida
- Esforço e duração aproximados
 - 10% do esforço do projeto
 - 10% da duração do projeto

Processo Unificado (características)

- Os requisitos não são completamente definidos antes do projeto
- O projeto não é completamente definido antes da programação
- A modelagem não é feita de forma completa e precisa
- A programação não é uma tradução mecânica do modelo para código
- As iterações não duram meses, mas sim semanas
- O planejamento não é especulativo, mas sim refinado durante o projeto

Exercício

- Analise com o seu grupo como o processo unificado será utilizado no trabalho
 - Qual será a duração de uma iteração?
 - O que vocês pretendem entregar em cada iteração?
 - Como e quando vocês vão se reunir para atingir esse objetivo?
 - Qual será o papel de cada membro do grupo?
 - Quais são os riscos envolvidos?
 - Quais decisões arquiteturais precisam ser tomadas (linguagem, SO, etc.)?



Como o cliente explicou



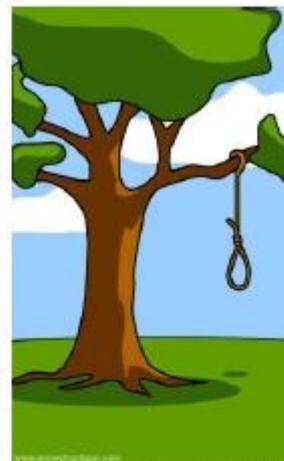
Como o lider de projeto entendeu



Como o analista planejou



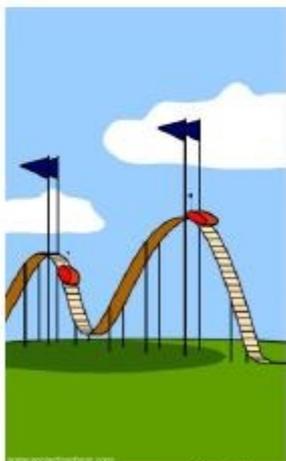
Como o programador codificou



O que os beta testers receberam



Como o consultor de negocios descreveu



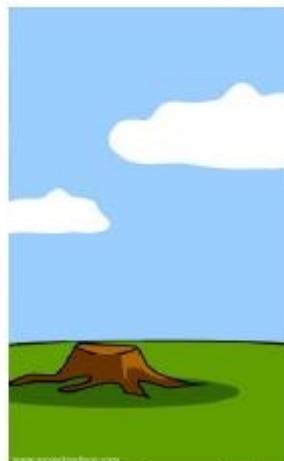
Valor que o cliente pagou



Como o projeto foi documentado



O que a assistencia tecnica instalou



Como foi suportado



Quando foi entregue



O que o cliente realmente necessitava

Bibliografia

- Larman, C.; 2007. Utilizando UML e Padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo. 3 ed. Bookman.
- Pressman, R. S.; 2004. Software Engineering: A Practitioner's Approach. 6 ed. McGraw-Hill.

Processo de Desenvolvimento de Software

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br