

PARADIGMAS DE ENGENHARIA DE SOFTWARE

Prof. Adailton Araújo

1



AGENDA

- Paradigma?
- Processo?
- Processo de Software?
- Modelos de Processo de Software
 - Sequencial
 - Incremental
 - Iterativo
 - Híbrido
- Desenvolvimento Ágil

PARADIGMA?

Exemplo típico ou modelo de algo. É a representação de um padrão a ser seguido.

PARADIGMA DE ENGENHARIA DE SOFTWARE?

Recordando...

Engenharia de Software = é uma disciplina de engenharia relacionada com todos os aspectos da produção de software. (*Sommerville, 2008*)

PROCESSO?

- No latim *procedere* é verbo que indica a ação de avançar, ir para frente (pro+cedere)
- Conjunto de manipulações para obter um resultado.
- Modo de fazer alguma coisa.

PROCESSO DE SOFTWARE?

“Combinação de **atividades**, **ferramentas** e **procedimentos** visando o desenvolvimento ou evolução de um software”. (*Sommerville, 2011*)

“**Roteiro**, ou conjunto de **passos**, **previsível** que ajuda a criar a tempo um software de alta qualidade (*Pressman, 2011*)

PROCESSO DE SOFTWARE?

- Um processo de software

- prescreve a **ordem** e a **frequência** de cada fase/atividade
- especifica **critérios** para mudar de uma fase para outra
- define o que tem que ser **entregue** ao final de cada fase

- Um processo de software **NÃO** significa

- “sobrecarga”, “papelada desnecessária”, “perda da tempo”

- Um processo de software tem efeito **positivo**

- para atender ao **cronograma** e obter software com mais **qualidade** e mais **fácil de manter**

PROCESSO DE SOFTWARE?

- Como escolher um processo
 - As **CARACTERÍSTICAS DA APLICAÇÃO** (domínio do problema, tamanho, complexidade etc);
 - A **TECNOLOGIA** a ser adotada na sua construção (paradigma de desenvolvimento, linguagem de programação, mecanismo de persistência etc), a organização;
 - **ONDE** o produto será desenvolvido;
 - O **PERFIL DA EQUIPE** de desenvolvimento.

Quando se escolhe um processo define-se um **MODELO DE PROCESSO (CICLO DE VIDA)**.

MODELOS DE PROCESSO DE SOFTWARE


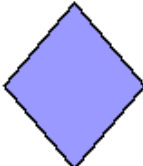

“É uma representação **abstrata** de um processo de software. Cada modelo de processo representa um processo sob determinada perspectiva e, desta forma, fornece somente informações parciais sobre esse processo”. *(Sommerville, 2008)*

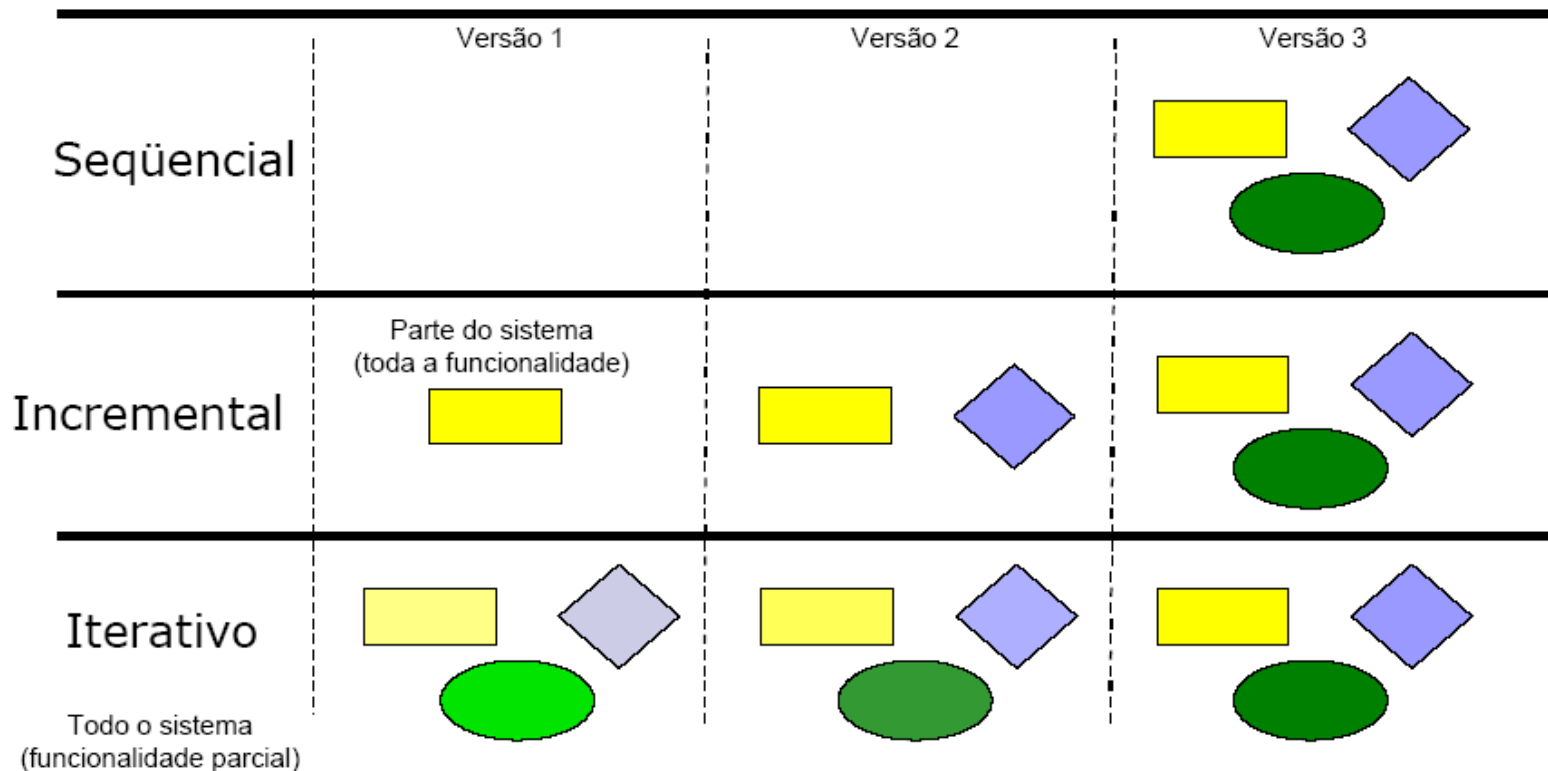
MODELOS DE PROCESSO DE SOFTWARE

- Sugerem um roteiro de atividades, ações, tarefas, marcos e produtos de trabalho necessários para desenvolver um software com qualidade.
- Engenheiros de software e gerentes adaptam um modelo prescritivo (genérico) a suas necessidades.

MODELOS DE PROCESSO DE SOFTWARE

Três abordagens principais

Software desejado =  +  + 



MODELO SEQUENCIAL

MODELO SEQUENCIAL - CASCATA

- Clássico, requer abordagem **sistemática** e **sequencial** ao desenvolvimento de software
- Principal característica
 - **“O resultado de uma fase é a entrada da próxima”**

Comunicação

- Iniciação do projeto
- Levantamento de Requisitos

Planejamento

- Estimativas
- Cronograma
- Monitoramento

Modelagem

- Análise
- Projeto

Construção

- Codificação
- Teste

Implantação

- Entrega
- Manutenção
- Feedback

MODELO SEQUENCIAL - CASCATA

MODELO SEQUENCIAL - CASCATA

○ Problemas

- Em projetos reais, é difícil estabelecer **todos os requisitos** no início de um processo (incertezas)
- Difícil acomodar **mudanças** com o processo em andamento, pois uma fase deve estar **completa** para passar para a próxima (sem paralelismo)
 - Inflexibilidade em estágios distintos dificulta resposta aos requisitos de mudança do **cliente**
- Cliente **paciente**
 - Uma versão executável só fica disponível em uma etapa **avançada** do desenvolvimento

MODELO SEQUENCIAL - CASCATA

○ Quando usá-lo?

- Apenas quando os **requisitos** são muito bem **compreendidos**, e quando as **mudanças** forem bastante **limitadas** durante o desenvolvimento
 - Poucos sistemas de negócio têm requisitos estáveis
- “... é significativamente **melhor** do que uma abordagem casual de desenvolvimento de software”
 - Modelo Cascata é simples e fácil de usar e gerenciar

MODELO SEQUENCIAL - CASCATA

○ Contribuições

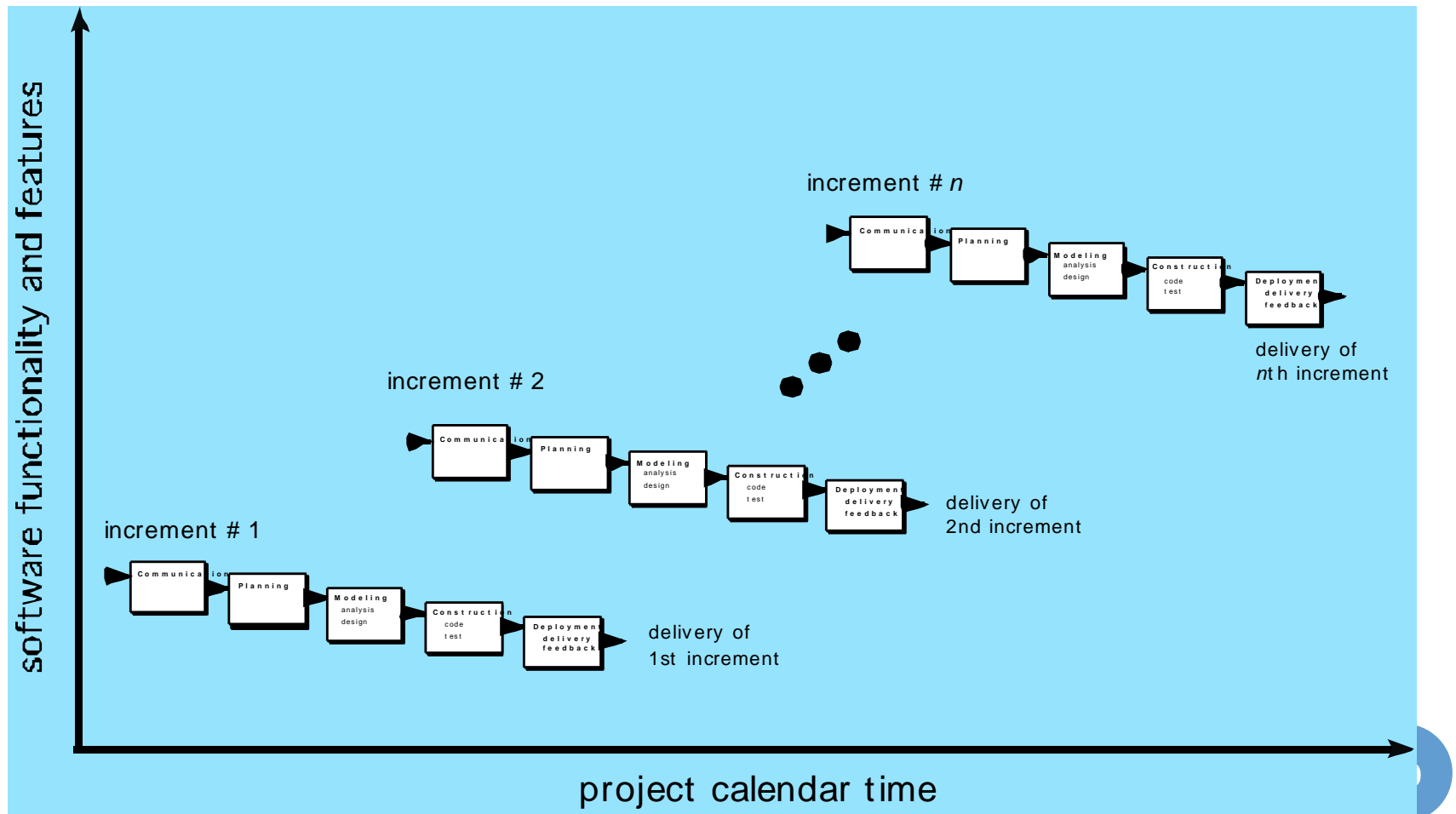
- O processo de desenvolvimento de software é submetido a **disciplina**, planejamento, gerenciamento e documentação
- A implementação do produto de software é postergada até que os requisitos tenham sido completamente entendidos
- Modelo **mais antigo** e foi amplamente usado na Engenharia de Software

MODELO INCREMENTAL

MODELO INCREMENTAL

- Ao invés de o sistema sofrer uma única entrega, o **desenvolvimento** e a **entrega** são **separados** em **incrementos**, sendo que cada incremento fornece **parte da funcionalidade** solicitada
- Requisitos de usuário são **priorizados** (primeiros são o núcleo básico) e os requisitos de prioridade mais alta são incluídos nos incrementos iniciais
 - Se o desenvolvimento de um incremento é iniciado, seus requisitos são **congelados**, embora os requisitos para incrementos posteriores possam continuar evoluindo

MODELO INCREMENTAL



MODELO INCREMENTAL

○ Quando usá-lo?

- Quando não há **mão de obra disponível** para uma **implementação completa** dentro do prazo de entrega estabelecido
- Quando o **núcleo do software** a ser desenvolvido é **bem conhecido**, mas não os requisitos em sua totalidade (mais realista)

MODELO INCREMENTAL

◦ Exemplo Prático

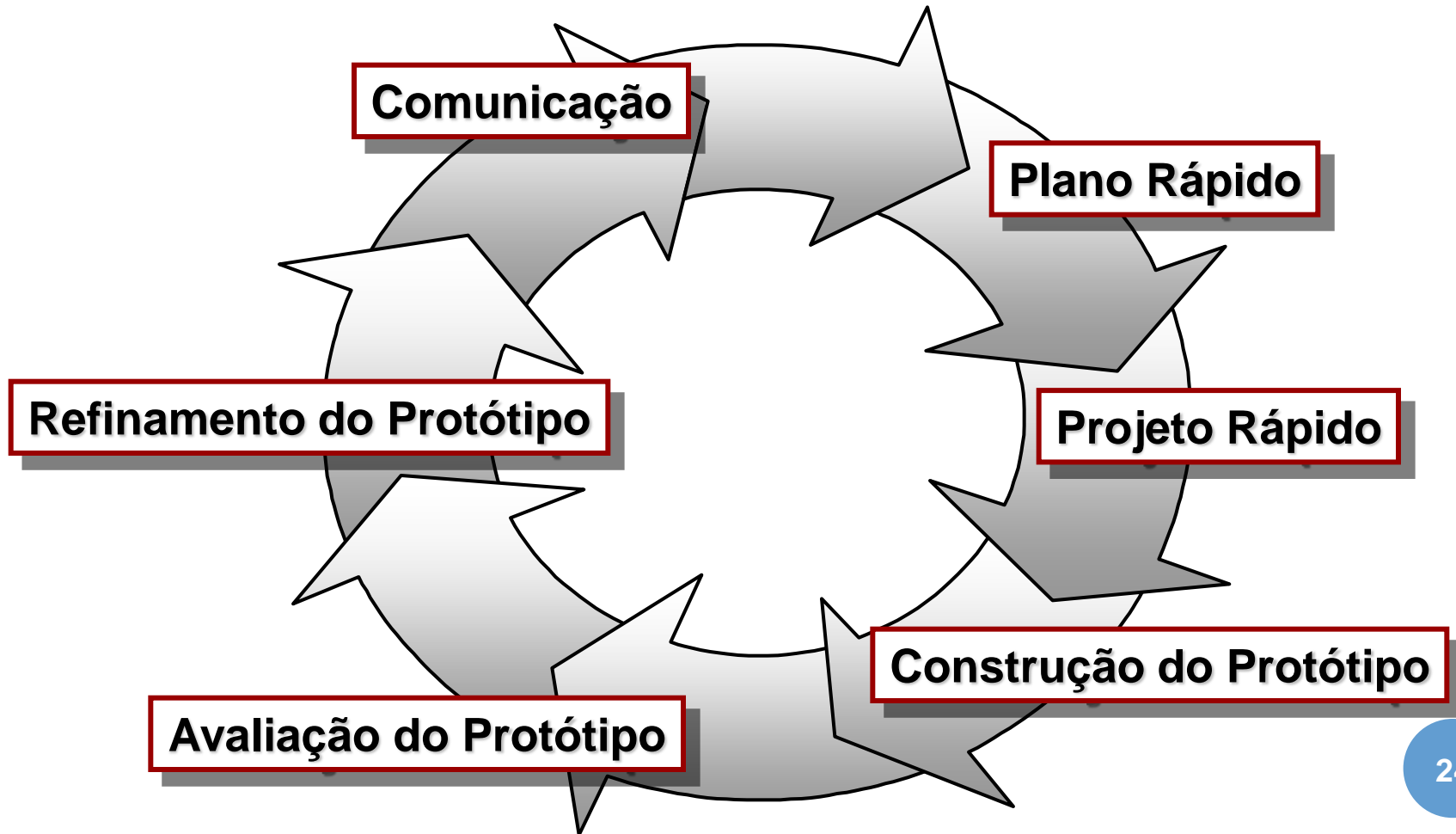
- Software de processamento de texto
 - Incremento inicial (núcleo)
 - gestão básica de arquivos, edição e produção de documentos (ex: novo, abrir, salvar, imprimir, formatação com tipo e tamanho de fonte, N, I, S, ...)
 - Incrementos posteriores
 - edição e produção de documentos mais sofisticados (ex: salvar como, gerar PDF,)
 - verificação ortográfica e gramatical
 - ...

MODELO ITERATIVO/EVOLUCIONÁRIO

MODELO ITERATIVO

- Modelos de processo **iterativos** que permitem desenvolver **versões cada vez mais completas** de um software
 - Desenvolve-se uma implementação **inicial**, expondo-a aos comentários do **usuário**
 - Depois, refina-se esse resultado por meio **de várias versões** até que seja desenvolvido um sistema adequado

MODELO ITERATIVO - PROTOTIPAGEM



MODELO ITERATIVO - PROTOTIPAGEM

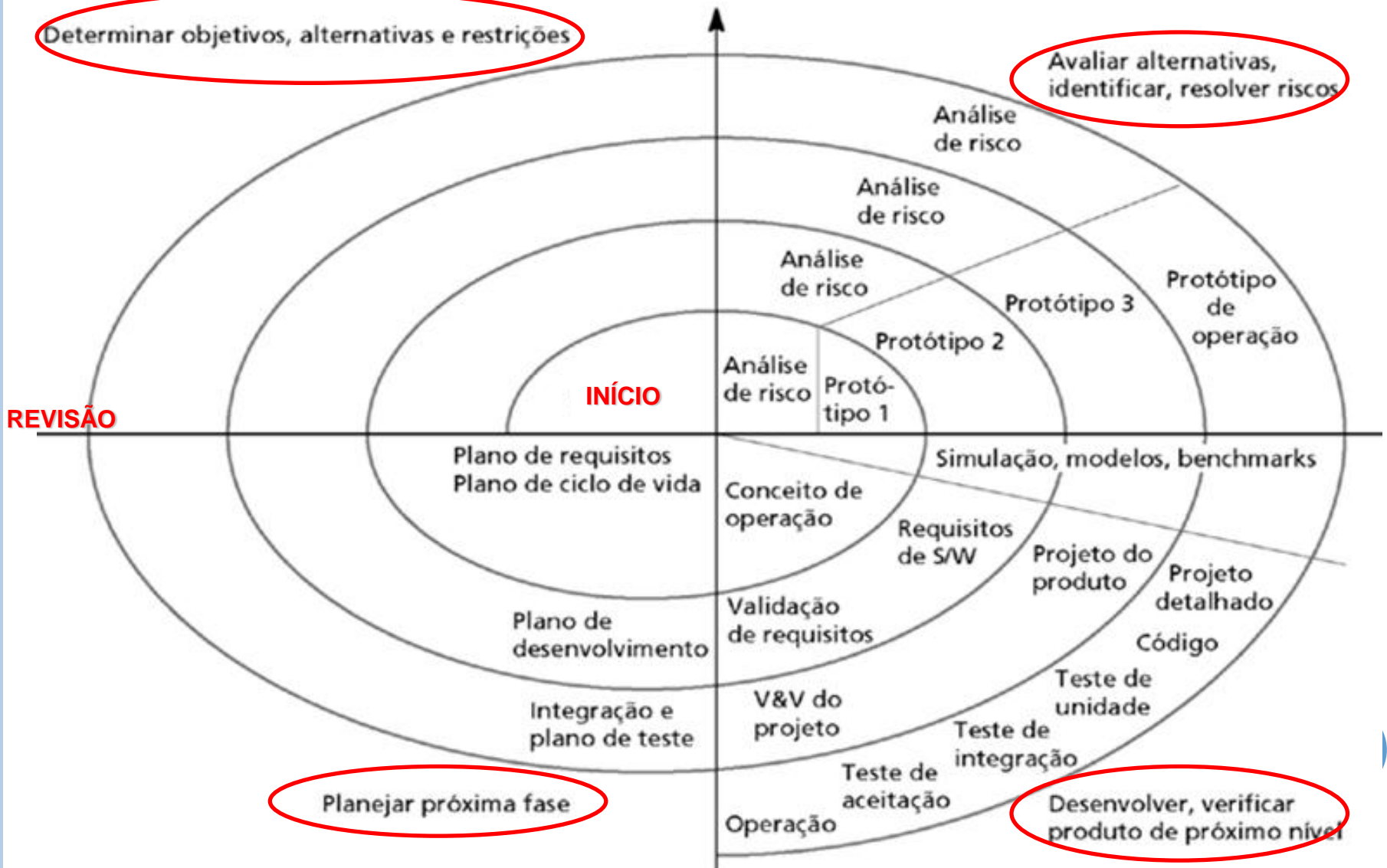
○ Vantagens

- Usuários têm o “**sabor**” de um sistema real precocemente
- Desenvolvedores conseguem “**entender**” o sistema e construir “**algo**” em prazo curto

○ Desvantagens

- Cliente “**pensa**” estar usando uma versão **operacional**
- Concessões **equivocadas** do desenvolvedor para entregar logo o protótipo (Algoritmo ineficiente, SO ou linguagem inapropriados)
- O descarte do protótipo pode ser visto com perda de tempo para o cliente

MODELO ITERATIVO - ESPIRAL



MODELO ITERATIVO - ESPIRAL

○ Vantagens?

- Riscos são gerenciados cedo e ao longo do processo – reatividade a riscos, que são reduzidos antes de se tornarem problemáticos
- Usa prototipação para reduzir riscos
- Software evolui enquanto o projeto prossegue – erros/alternativas não atrativas são eliminadas cedo
- Planejamento é construído sobre o processo – cada ciclo inclui um passo de planejamento para auxiliar o monitoramento do projeto

MODELO ITERATIVO - ESPIRAL

○ Desvantagens?

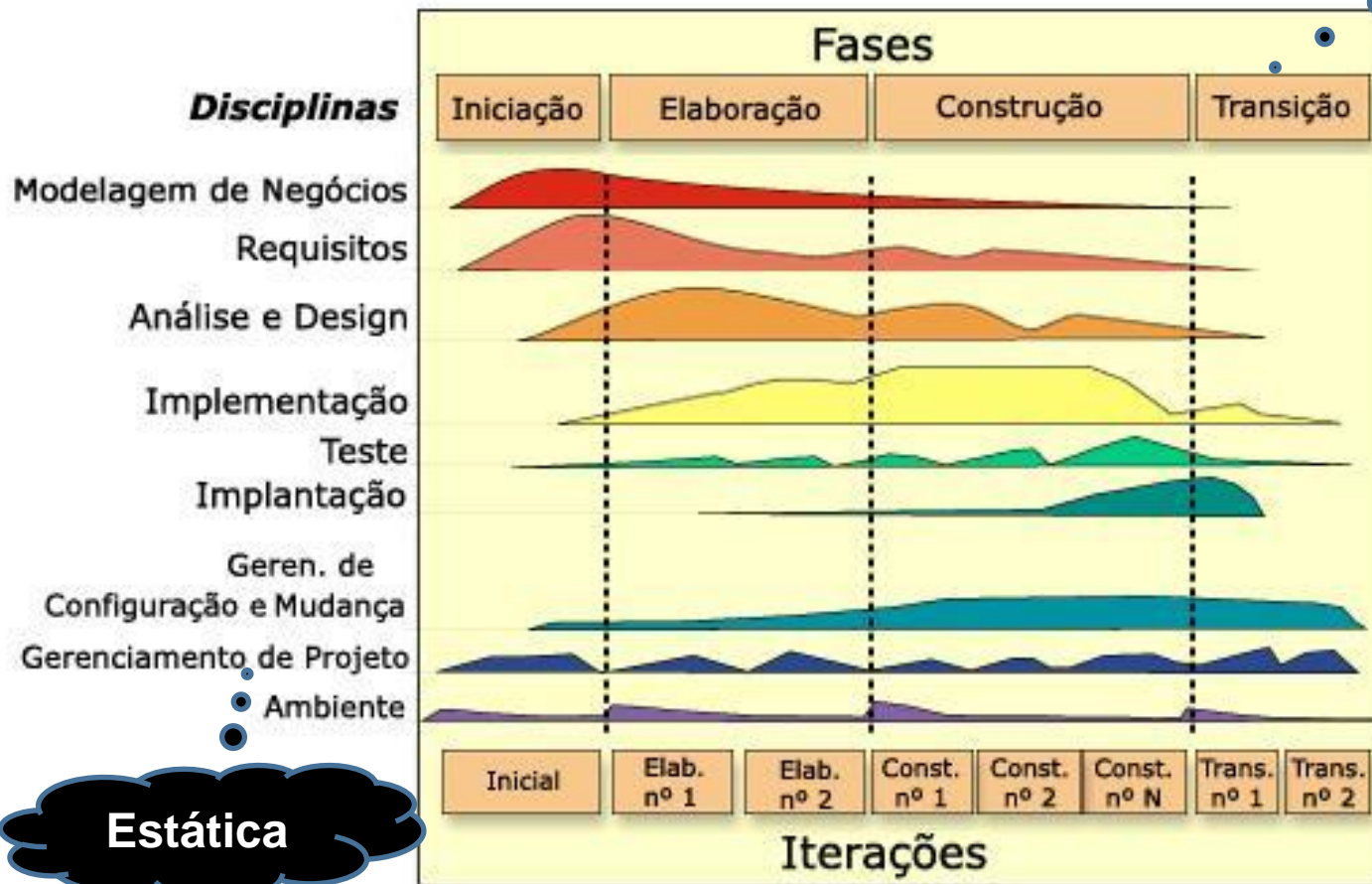
- Complicado de usar e controlar – análise de riscos requer experiência e, conseqüentemente, \$\$\$
 - Se um risco importante não for detectado ou bem gerenciado, ocorrerão problemas
- Pode ser inadequado para pequenos projetos – não faz sentido se o custo da análise de riscos é grande parte do custo do projeto como um todo

MODELO HÍBRIDO

MODELO HÍBRIDO - RUP

○ Rational Unified Process (RUP)

Dinâmica



Prática



Principal inovação: separação das fases e disciplinas

DESENVOLVIMENTO ÁGIL

DESENVOLVIMENTO ÁGIL

- Manifesto para o Desenvolvimento Ágil
 - Assinado em **2001** por Kent Beck e outros 16 desenvolvedores, autores e consultores de software

“Desenvolvendo e ajudando outros a desenvolver software, estamos desvendando formas melhores de desenvolvimento. Por meio deste trabalho passamos a valorizar:

- *Indivíduos e interações acima de processos e ferramentas*
- *Software operacional acima de documentação completa*
- *Colaboração dos clientes acima de negociação contratual*
- *Respostas a mudanças acima de seguir um plano”*

DESENVOLVIMENTO ÁGIL

- Surgiram de um esforço para **sanar fraquezas** reais e perceptíveis da engenharia de software convencional
- Oferece **benefícios** importantes

MAS, não é indicado para todos os tipos de projetos, produtos, pessoas e situações

DESENVOLVIMENTO ÁGIL



DESENVOLVIMENTO ÁGIL

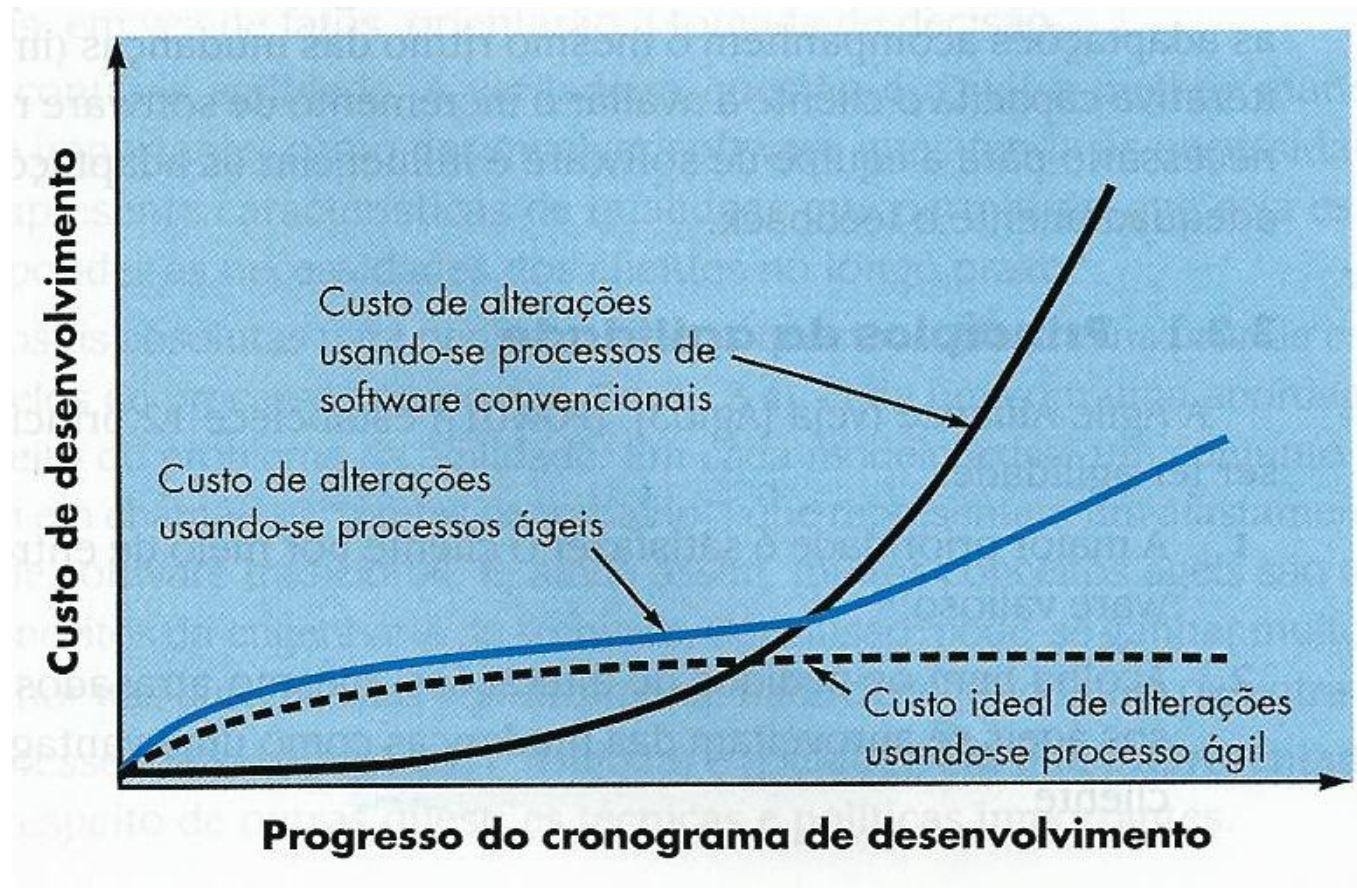
- **AGILIDADE**

- Entregar versões funcionais em **prazos curtos**
- Estar preparado para **requisitos mutantes**
- Pessoal de negócios e desenvolvedores **juntos**
- Cliente é considerado parte da equipe (visão nós e eles)
- Troca de informações através de **conversas diretas**
- Plano de projeto deve ser **flexível**

DESENVOLVIMENTO ÁGIL

- **Agilidade e Custo das Mudanças**
 - Desenvolvimento de software **convencional** afirma que os **custos com de mudanças aumentam** de forma não linear conforme o projeto avança
 - Defensores da **agilidade** argumentam que o processo ágil bem elaborado **“achata” o custo da curva de mudança**
 - Processo ágil envolve entregas incrementais
 - Custo das mudanças é atenuado com entrega incremental associada a outras práticas ágeis: testes contínuos de unidade e programação por pares

DESENVOLVIMENTO ÁGIL



DESENVOLVIMENTO ÁGIL

Processo Ágil?

- Processo capaz de administrar a *imprevisibilidade*
 - Processo facilmente **adaptável**
 - Adaptar **incrementalmente**
 - Equipe precisa de **feedback do cliente** para adaptações incrementais
 - Catalisador para feedback do cliente é um **protótipo operacional** ou parte de um sistema operacional entregues em curtos períodos de tempo

DESENVOLVIMENTO ÁGIL

Abordagens?

- **Extreme Programming – XP – Programação Extrema**
- Industrial XP – IXP – Programação extrema industrial
- **Scrum**
- Adaptive Software Development – ASD – Desenvolvimento de software adaptativo
- Dynamic Systems Development Method – DSDM – Método de Desenvolvimento de Sistemas Dinâmicos
- Crystal

DESENVOLVIMENTO ÁGIL

XP – Extreme Programming

- Amplamente divulgado em 2004 por Kent Beck
- Abordagem mais amplamente usada para desenvolvimento de software ágil (dados de 2011)
- A que se destina
 - Grupos de 2 a 10 programadores
 - Projetos de 1 a 36 meses

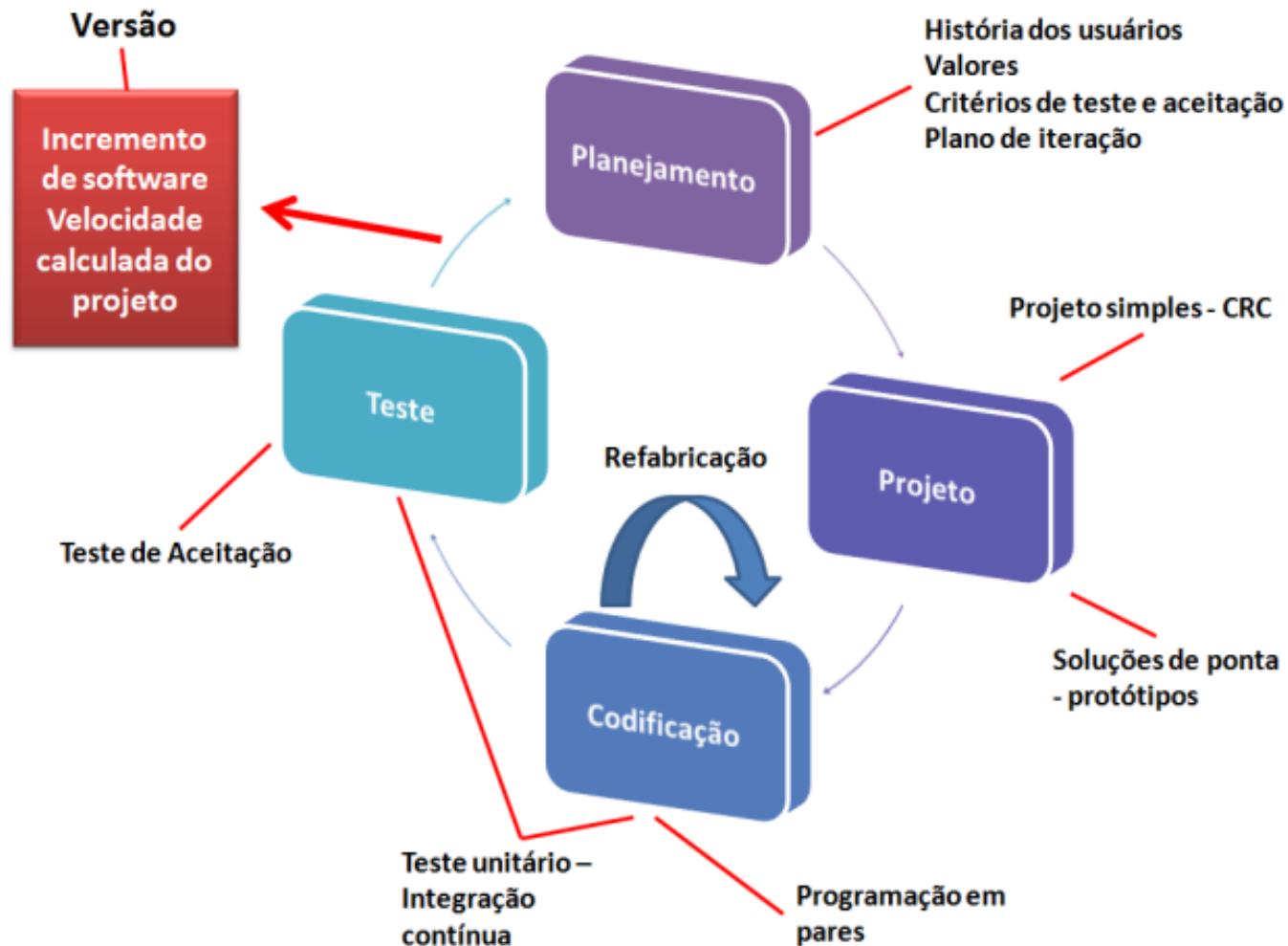
DESENVOLVIMENTO ÁGIL

XP – Extreme Programming

- Valores para trabalhos realizados com a XP
 - **Comunicação** – contínua entre cliente e desenvolvedores
 - **Simplicidade** – projetar apenas para as necessidades imediatas
 - **Feedback** – com base no próprio software implementado
 - **Coragem** – coragem e disciplina para projetar para hoje
 - **Respeito** – Membros da equipe, Clientes e o próprio software desenvolvido (faz bem feito o que deve fazer HOJE)

DESENVOLVIMENTO ÁGIL

XP – Extreme Programming



DESENVOLVIMENTO ÁGIL

Scrum

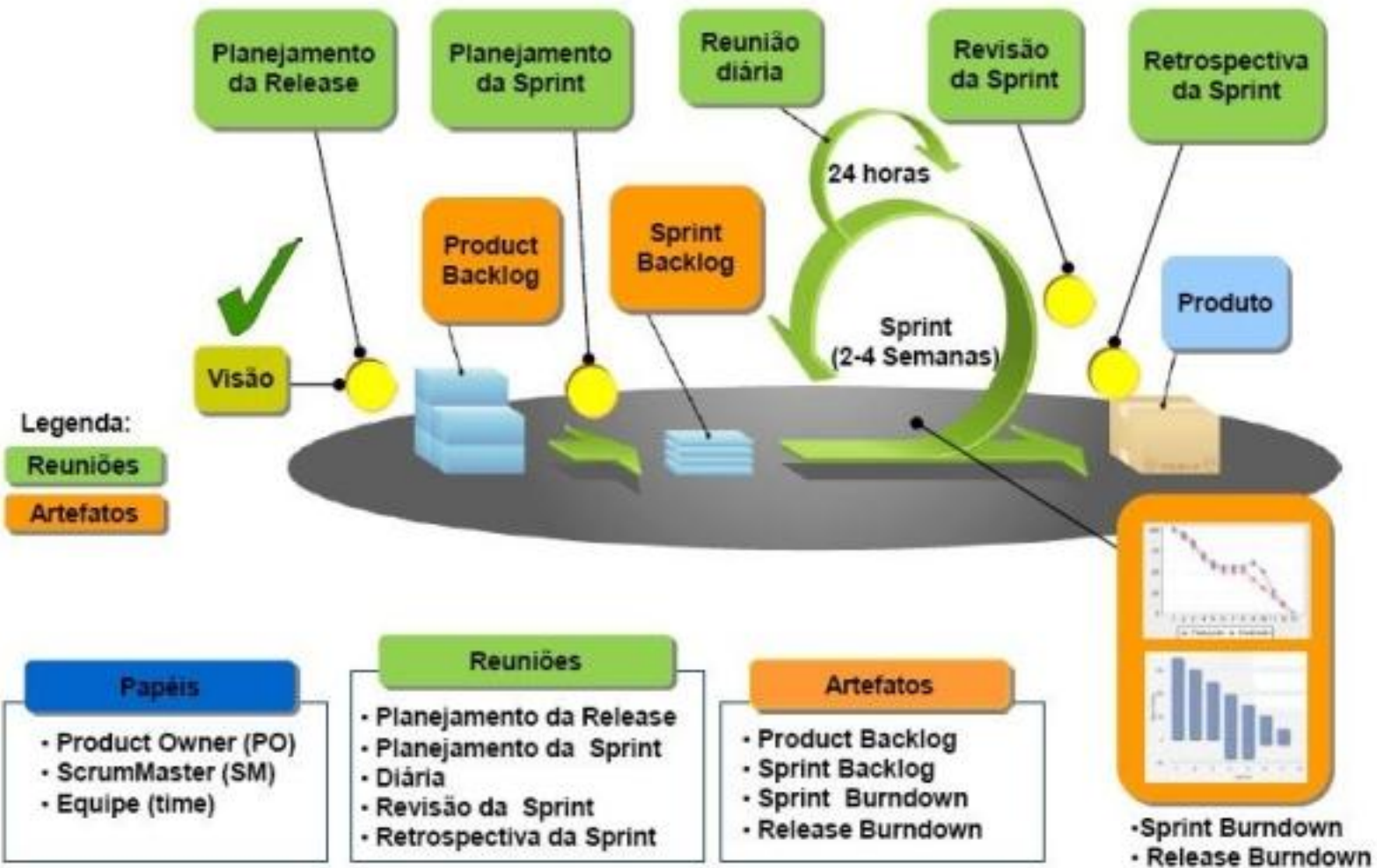
- Definição Informal
 - Estratégia em um jogo de rugby onde jogadores colocam uma bola quase perdida novamente em jogo através de trabalho em equipe
- Método de desenvolvimento **ágil** de software
- Concebido por Jeff Sutherland no início de 1990
- Princípios são consistentes com o **manifesto ágil** e usados para **orientar** as atividades de desenvolvimento dentro de um processo que incorpora as atividades de requisitos, análise, projeto, evolução e entrega

DESENVOLVIMENTO ÁGIL

Scrum

- Enfatiza o uso de padrões de processos de software
 - **Backlog**: lista com requisitos pendentes com prioridades
 - **Sprints**: unidades de trabalho com requisitos estabelecidos e prazo de entrega (geralmente 2 à 4 semanas)
 - **Alterações**: não são introduzidas em Sprints já iniciados – membros trabalham em ambiente de curto prazo, mas estável
 - **Reuniões Scrum**: reuniões curtas (15 minutos) realizadas diariamente
 - O que realizou desde a última reunião?
 - Quais obstáculos está encontrando?
 - O que planeja realizar até a próxima reunião?

DESENVOLVIMENTO ÁGIL



DÚVIDAS

