

# Introdução ao Processo Unificado (PU)

**Prof. Fernando Maia da Mota**

Slides gentilmente cedidos por Profa. Dra. Maria Istela Cagnin Machado  
UFMS/FACOM

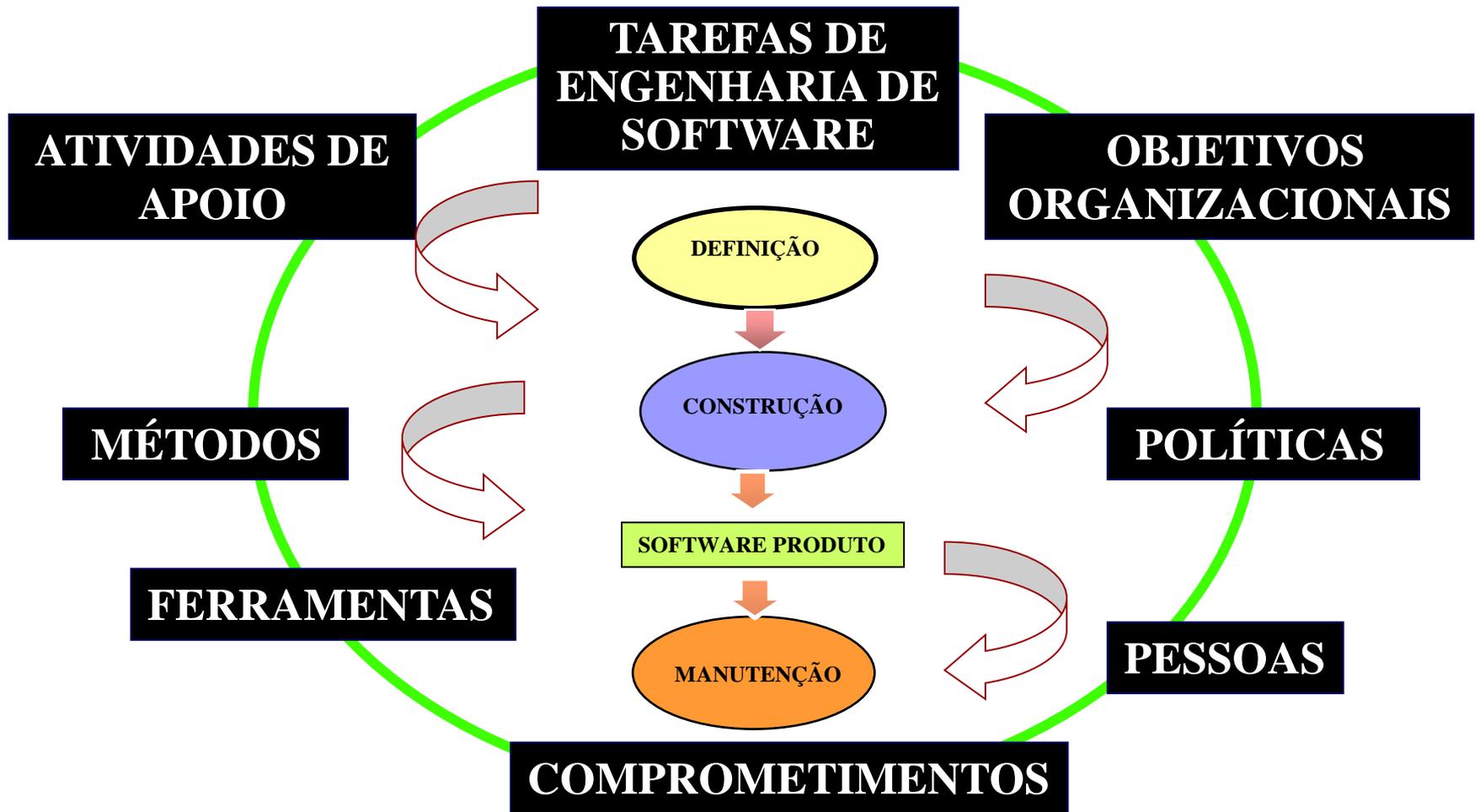
# Roteiro

- Considerações Iniciais
- Modelo Incremental
- Processo Unificado
  - O que é?
  - Elementos
  - Princípios básicos
  - Fases



# Considerações Iniciais

- Desenvolver software geralmente é uma tarefa complexa



# Considerações Iniciais

- Necessidade de estabelecer processos sistemáticos para desenvolvimento:  
***Modelos de processo de Software***
  - Modelo em Cascata
  - Modelo de Prototipação
  - Desenvolvimento Baseado em Componentes
  - Modelo de Métodos formais
  - Modelo incremental (***Processo Unificado***, RUP, ...)
  - Métodos ágeis (eXtremme Programming (XP), Scrum, OpenUp, ...)

# O Modelo Incremental

- O modelo incremental combina elementos do modelo cascata (aplicado repetidamente) com a filosofia iterativa da prototipação
- Objetivo: trabalhar junto com o usuário para descobrir seus requisitos, de maneira incremental, até que o produto final seja obtido

# O Modelo Incremental

- A versão inicial é frequentemente o núcleo do produto (a parte mais importante)
- A evolução acontece quando novas características são adicionadas à medida que são sugeridas pelo usuário
- Este modelo é importante quando é difícil estabelecer *a priori* uma especificação detalhada dos requisitos ou quando os requisitos mudam com frequência

# Processo Unificado (PU)

- O PU foi proposto pelos três gurus da orientação a objetos (Jacobson, Booch & Rumbaugh, 1999):
  - Ivar Jacobson
  - Grady Booch
  - James Rumbaugh
- É o resultado de mais de trinta anos de experiência acumulada

# O que é o Processo Unificado?

- É um modelo de processo de software baseado no modelo incremental, visando a construção de software orientado a objetos
- Usa como notação de apoio a UML (*Unified Modeling Language*)

# O que é o Processo Unificado?

- É um processo de software: *conjunto de **atividades** executadas para transformar um conjunto de **requisitos** do cliente em um **sistema de software***
- Um processo de desenvolvimento de software descreve uma abordagem para a ***construção**, **implantação** e **manutenção*** do software

# O que é o Processo Unificado?

- É um *framework* que pode ser personalizado de acordo com as necessidades específicas e recursos disponíveis para cada projeto
- Instanciações do PU
  - RUP (*Rational Unified Process*)
  - OpenUp (versão ágil do PU)

# Elementos do Processo Unificado

- Um processo descreve  
**quem** (papel)  
está fazendo **o quê** (artefato),  
**como** (atividades) e  
**quando** (disciplina)

# Quem? Trabalhadores

- Um ***trabalhador*** é alguém que desempenha um papel e é responsável pela realização de atividades para produzir ou modificar um artefato

# O quê? Artefatos

- Um **artefato** é uma porção significativa de informação interna ou que será fornecida a interessados externos que desempenhem um papel no desenvolvimento do sistema
- Um artefato é algum documento, relatório, modelo ou código que é produzido, manipulado ou consumido
- **Exemplos:** modelo de caso de uso, modelo do projeto, um caso de uso, um subsistema, um caso de negócio, um documento de arquitetura de software, código fonte, executáveis, etc.

# Como? Atividades

- Uma ***atividade*** é uma tarefa que um trabalhador executa a fim de produzir ou modificar um artefato

# Quando? Disciplina

- A ***disciplina*** descreve as sequências das atividades que produzem algum resultado significativo e mostra as interações entre os participantes
- São realizadas a qualquer momento durante o ciclo de desenvolvimento (Fases do PU)
- Requisitos, Análise, Projeto, Implementação e Teste

# Disciplinas do PU

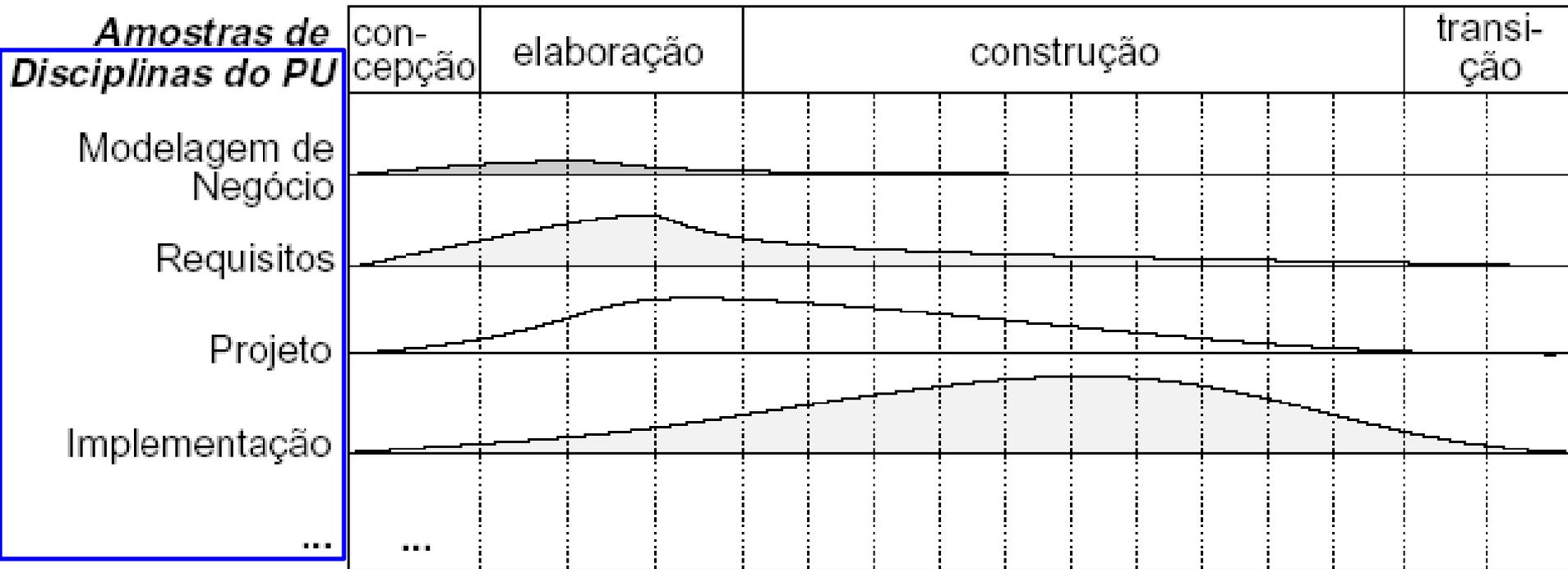


Figura extraída de Larman, 2004

# Princípios básicos do PU

- Desenvolvimento iterativo
- Baseado em casos de uso
- Centrado na arquitetura

# Desenvolvimento iterativo

- O desenvolvimento de um software é dividido em vários ciclos de iteração, cada qual produzindo um sistema testado, integrado e executável
- Em cada ciclo ocorrem as atividades de **análise de requisitos, projeto, implementação e teste**, bem como a integração dos artefatos produzidos com os artefatos já existentes

# Desenvolvimento iterativo

- Planejar quantos ciclos de desenvolvimento serão necessários para alcançar os objetivos do sistema
- As partes mais importantes (requisitos) devem ser priorizadas e alocadas nos primeiros ciclos de acordo com os seguintes critérios:
  - (1) possuem alto valor de negócio
  - (2) possuem alto risco (ex. devem poder manipular 500 transações concorrentes)
  - (3) são arquiteturalmente significativas

# Desenvolvimento iterativo

- Na primeira iteração estabeleça os principais riscos e o escopo inicial do projeto, de acordo com a funcionalidade principal do sistema
- As partes mais complexas do sistema devem ser atacadas já no primeiro ciclo, pois são elas que apresentam maior risco de inviabilizar o projeto

# Desenvolvimento iterativo

- O tamanho de cada ciclo pode variar de uma empresa para outra e conforme o tamanho do sistema
- Por exemplo, uma empresa pode desejar ciclos de 3 semanas, outra pode preferir 6 semanas
  - recomenda-se que a duração de uma iteração seja entre **duas** e **seis** semanas

# Desenvolvimento iterativo

**CIDADÃO!**

- Uma iteração muito longa perde o objetivo do desenvolvimento iterativo!

# Desenvolvimento iterativo

- Se o projetista observar que será difícil cumprir o prazo da iteração, é recomendável remover tarefas ou requisitos da iteração atual e incluí-las em uma iteração futura, em vez de não cumprir o prazo

# Desenvolvimento iterativo

- Cada iteração exige a escolha de um pequeno subconjunto de requisitos, além da sua rápida construção e teste
  - casos de uso complexos: os vários cenários do mesmo caso de uso podem ser tratados ao longo de várias iterações
  - casos de uso simples e curtos: podem ser completados em uma iteração
- Nas iterações iniciais o resultado pode não ser exatamente o que é desejado em última instância

# Desenvolvimento iterativo

- No entanto, o ato de executar rapidamente um pequeno passo antes de finalizar todos os requisitos leva a uma realimentação rápida
- Essa realimentação é muito importante:
  - ao invés de especular sobre os requisitos corretos e completos, a equipe procura a realimentação a partir de uma construção e teste realístico de alguma coisa buscando uma percepção prática do sistema

# Desenvolvimento iterativo

- Os usuários têm a oportunidade de ver um sistema e dizer:

*“sim! Foi isso que eu pedi, mas agora que o experimentei, o que eu realmente quero é algo ligeiramente diferente!”*

- Esse “sim! Mas...” não é um sinal de erro. É um modo hábil de progredir e descobrir o que é de real valor no sistema para os interessados (*stakeholders*)

# Baseado em Casos de Uso

- Um caso de uso é uma sequência de ações, executadas por um ou mais atores e pelo próprio sistema, que produz um ou mais resultados de valor para um ou mais atores
- O PU é dirigido por casos de uso, pois os utiliza para dirigir todo o trabalho de desenvolvimento, desde a captação inicial e negociação dos requisitos até a aceitação do código (testes)

# Baseado em Casos de Uso

- Os casos de uso são centrais ao PU e outros métodos iterativos, pois:
  - os requisitos funcionais são registrados preferencialmente por meio deles
  - o planejamento do desenvolvimento é feito em função dos casos de uso identificados, tratando-se prioritariamente os mais complexos
  - o teste é baseado neles

# Centrado na Arquitetura

- Arquitetura é a organização fundamental do sistema como um todo
- Arquitetura descreve o sistema em termos de sua organização em camadas, pacotes, classes, interfaces e subsistemas

# Centrado na Arquitetura

- A análise arquitetural está envolvida na identificação e na resolução dos requisitos não-funcionais (ex., qualidade) do sistema.
- A arquitetura se refere a questões como:
  - desempenho, escalabilidade, reuso e restrições econômicas e tecnológicas

# Centrado na Arquitetura

- O PU prioriza a construção de uma arquitetura de sistema que permita a realização dos requisitos
- Essa arquitetura baseia-se na identificação de uma estrutura de classes, produzida a partir de um modelo conceitual
- A cada ciclo de trabalho realizado, novas características são adicionadas à arquitetura do sistema, deixando-a mais completa e mais próxima do sistema final

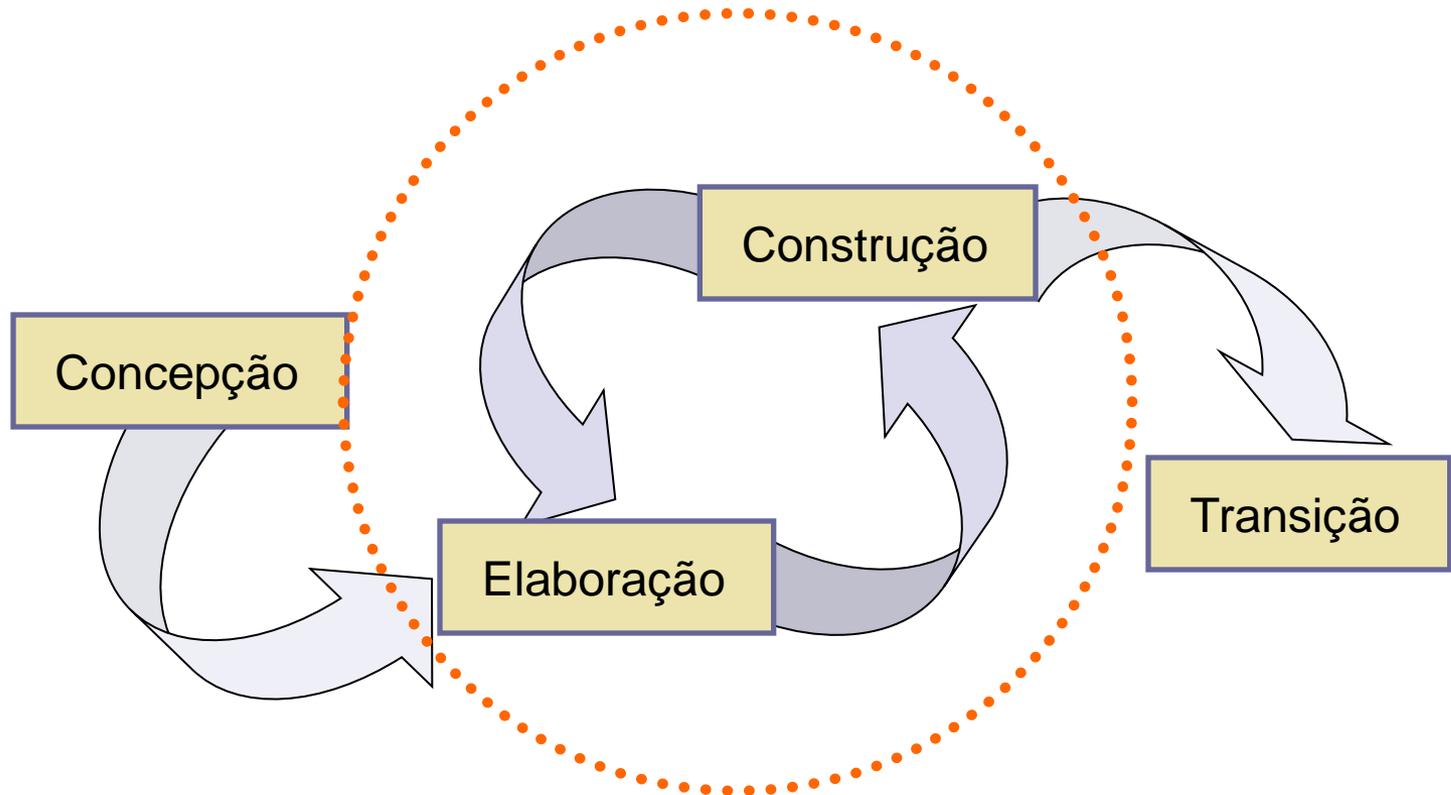
# Centrado na Arquitetura

- No PU, a arquitetura do sistema em construção é o alicerce fundamental sobre o qual ele se erguerá
- Deve ser uma das preocupações da equipe de projeto
- A arquitetura, juntamente com os casos de uso, deve orientar a exploração de todos os aspectos do sistema

# Fases do PU

- O PU é dividido em quatro fases:
  - Concepção
  - Elaboração
  - Construção
  - Transição

# Fases do PU



Pag. 24 -Wazlawick

# Fases e Disciplinas do PU

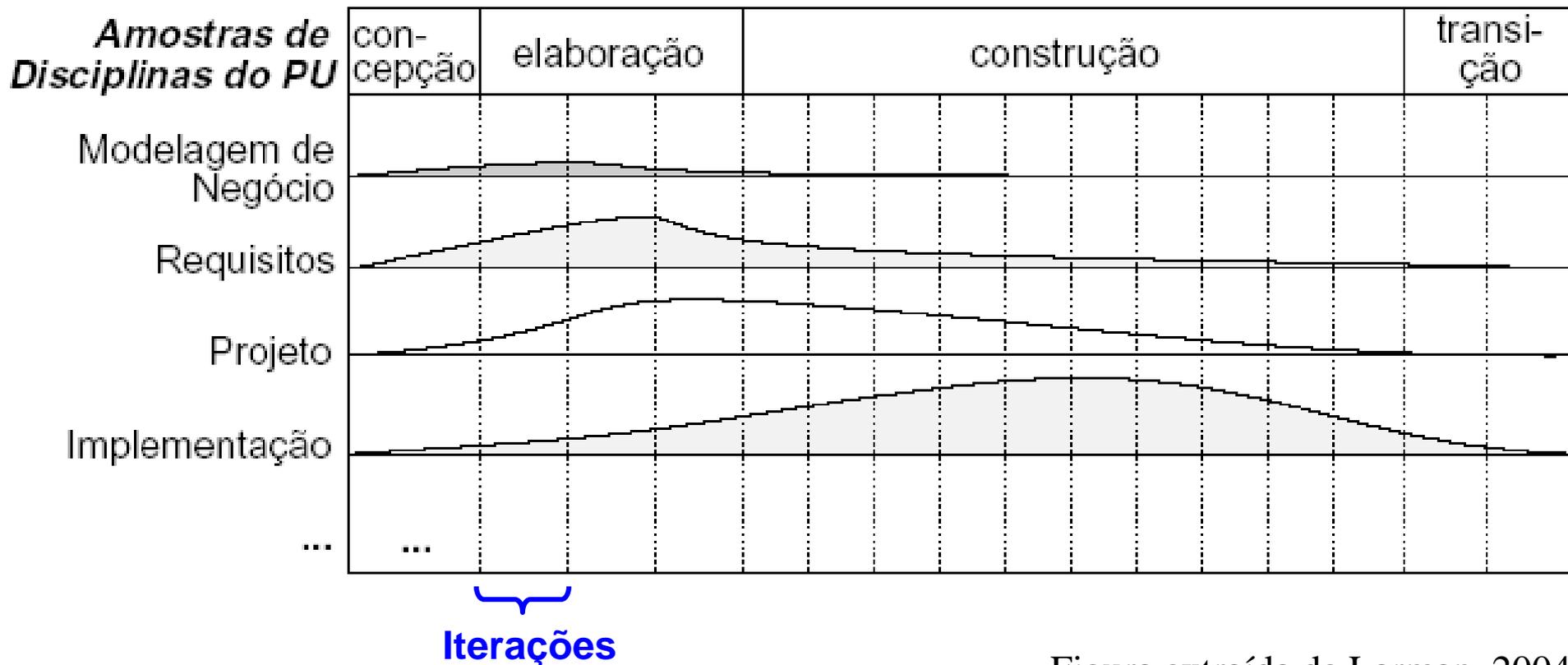


Figura extraída de Larman, 2004

# Fases do PU: Concepção

- *Inception* em inglês
- Estuda a viabilidade de implantação do sistema
- Definição do escopo do sistema
- Estimativas de custos e cronograma
- Identificação dos potenciais riscos que devem ser gerenciados ao longo do projeto
- Esboço da arquitetura do sistema, que servirá como alicerce para a sua construção

# Fases do PU: Elaboração

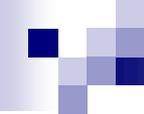
- Visão refinada do sistema, com a **definição dos requisitos funcionais**, **detalhamento da arquitetura** criada na fase anterior e **gerenciamento contínuo dos riscos** envolvidos
- Incorpora:
  - Maior parte da análise de requisitos
  - Projeto (*design*)

# Fases do PU: Construção

- O sistema é efetivamente desenvolvido e, em geral, tem condições de ser operado, mesmo que em ambiente de teste, pelos clientes
- Incorpora:
  - Implementação
  - Testes

# Fases do PU: Transição

- O sistema é entregue ao cliente para uso em produção
- Testes são realizados
- Um ou mais incrementos do sistema são implantados
- Defeitos são corrigidos, se necessário
- Incorpora:
  - Instalação e manutenção

- 
- Apesar de ser um processo prescritivo, o UP pode também ser realizado como um processo ágil, com poucos artefatos e burocracia, permitindo o desenvolvimento de software de forma eficiente, uma vez que o que interessa ao cliente é o software pronto e não uma pilha de documentos justificando porque não ficou pronto.

- Para que isso seja obtido, a **documentação** deve ser **dirigida à produção** do software.
- Cada atividade realizada pelo desenvolvedor deve ter um objetivo muito claro e uma utilização precisa visando sempre à produção de um código que atenda aos requisitos da melhor forma possível no menor tempo.

# Questões

- Discorra sobre as principais características do PU apresentando suas vantagens e desvantagens (se for o caso)

# Exercício de Casa - Leitura

- Wazlawick, *Análise e Projeto de Sistemas de Informação Orientados a Objetos*, 2004, editora Campus – **capítulos 1 e 2**