




# Processo Unificado

# Processo Unificado

- ➔ É um processo genérico de software que pode ser especializado
- ➔ Utiliza a UML (Unified Modeling Language) como notação.
- ➔ Características:
  - Direcionado por Caso de Uso
  - Centrado na Arquitetura
  - Iterativo e Incremental




# Processo Unificado

- ➔ Direcionado por Caso de Uso
- Os casos de uso descrevem a funcionalidade completa do sistema.
- Direcionam o projeto (design), implementação e teste.

# Processo Unificado

## ➔ Centrado na Arquitetura

- A arquitetura define os aspectos estáticos e dinâmicos mais significantes do sistema.
- A arquitetura é influenciada por:
  - plataforma do sw (e.g. arquitetura do computador, sistema operacional, SGBD, protocolos para comunicação);
  - blocos que serão reusáveis (e.g. framework para interface gráfica);
  - sistemas legados; requisitos não-funcionais, etc..
- Inicialmente é definido um esboço da arquitetura; em seguida os casos de uso mais importantes são especificados em termos de subsistemas e classes; e a medida que outros casos de uso são especificados, a arquitetura é melhor definida.



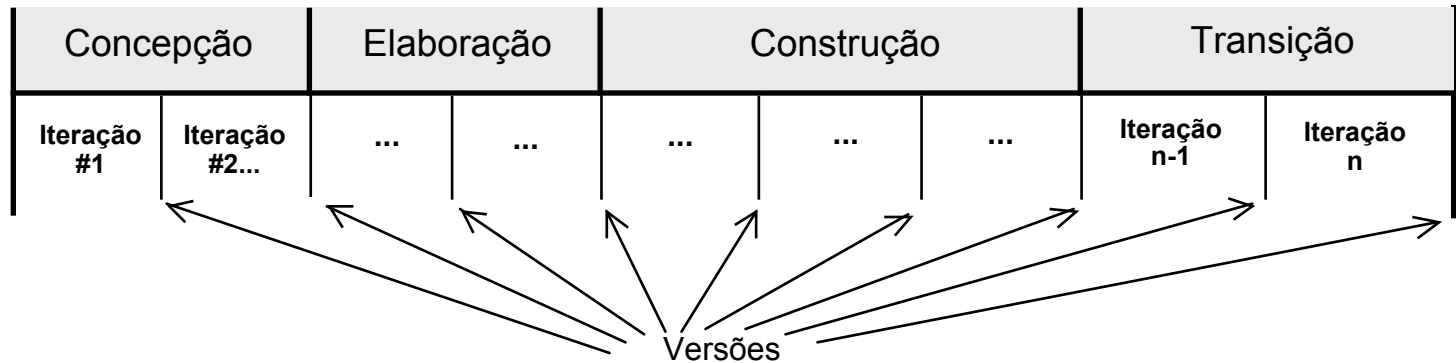
# Processo Unificado

## ➔ Iterativo e Incremental

- O desenvolvimento é dividido em mini-projetos. Cada mini-projeto é uma iteração que resulta em um incremento.

# Processo Unificado

- Uma série de ciclos são repetidos durante a vida do sistema.
- Cada ciclo consiste de quatro fases: concepção, elaboração, construção e transição.





# Fases do Processo Unificado

## Fase de Concepção (Inception Phase)

Estudo rápido do sistema proposto a partir de uma visão do produto final. (Continuar ou não o projeto?)

## Fase de Elaboração

A maioria dos casos de uso são especificados detalhadamente e a arquitetura do sistema é projetada (modelo de casos de uso, modelo de análise, modelo de projeto, etc.).

## Fase de Construção

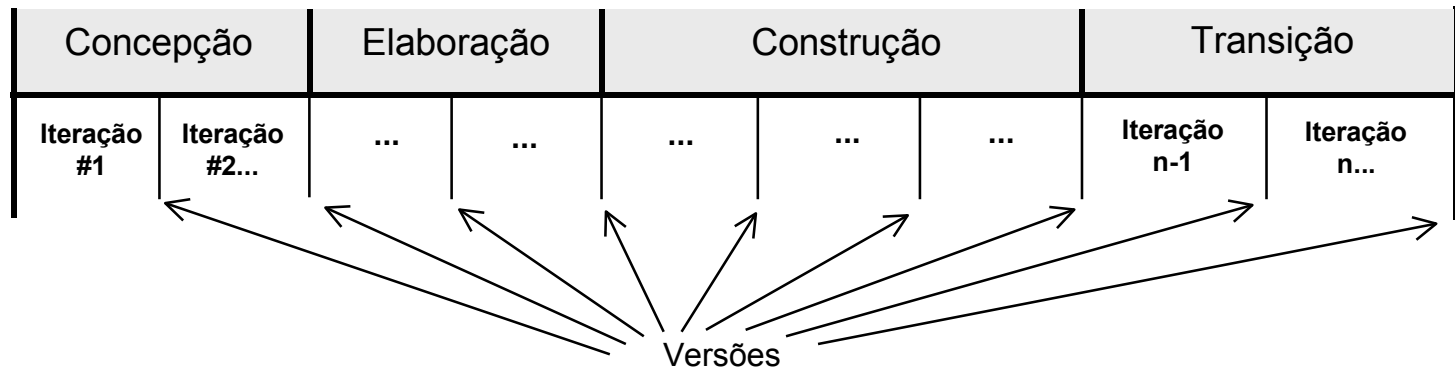
O produto é construído e torna-se operacional.

## Fase de Transição

O produto entra na fase de teste beta: um número reduzido de usuários experientes utilizam o produto e identificam defeitos e deficiências.

# Fases do Processo Unificado

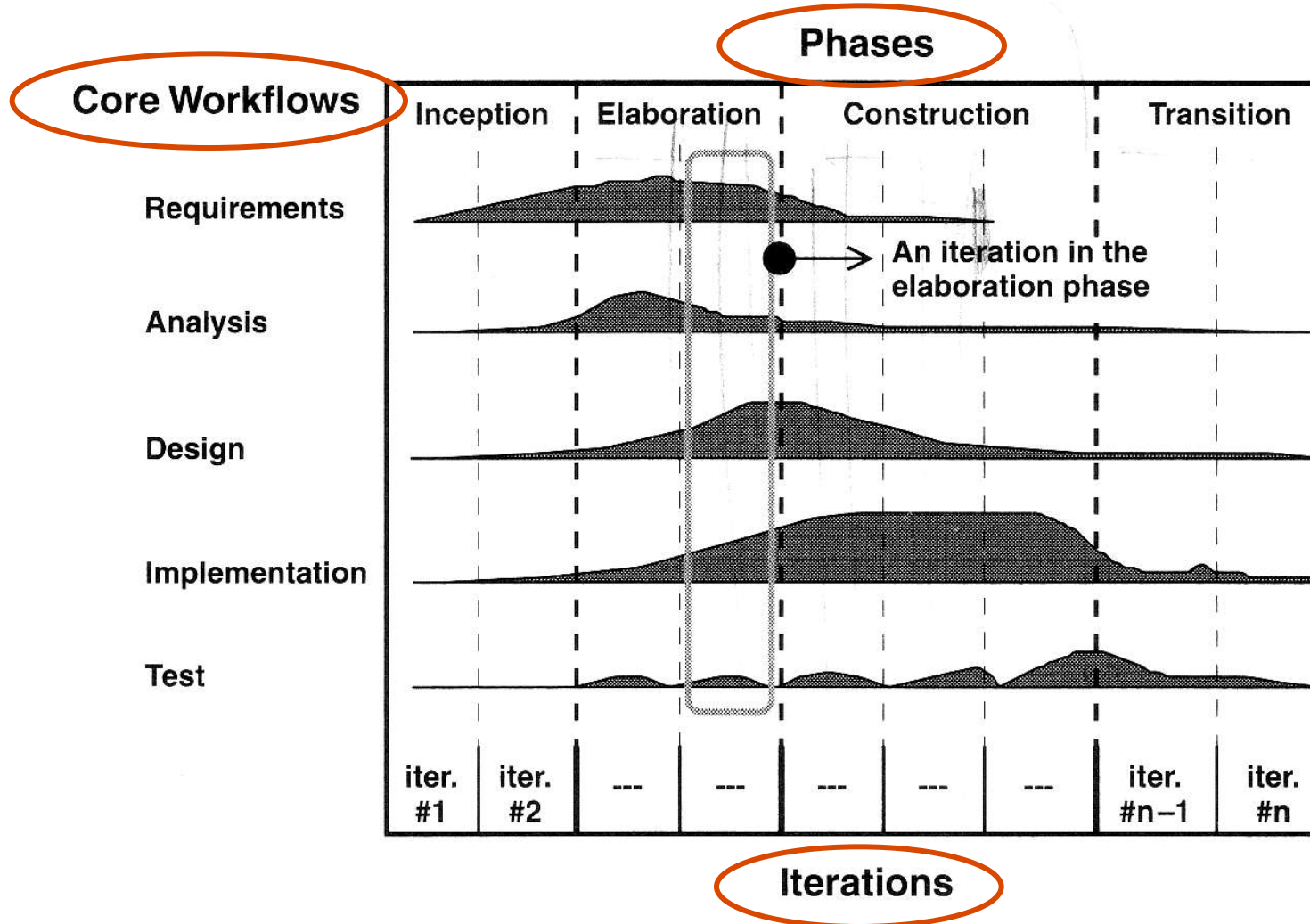
- Cada fase é subdividida em iterações.



- Cada iteração apresenta os workflows de requisitos, análise, projeto (design), implementação e testes.



# Fases do Processo Unificado





# Workflows do Processo Unificado

1. Workflow de Requisitos
2. Workflow de Análise
3. Workflow de Projeto
4. Workflow de Implementação
5. Workflow de Teste



# Workflow de Requisitos



# Atividades do Workflow de Requisitos

Objetivo: planejar o desenvolvimento em direção ao sistema correto.

Alguns passos normalmente realizados no levantamento de requisitos:

1. Entender o contexto do sistema
2. Enumerar os requisitos candidatos
3. Capturar os requisitos funcionais
4. Capturar os requisitos não-funcionais

# Atividades do Workflow de Requisitos

Alguns passos normalmente realizados no levantamento de requisitos:

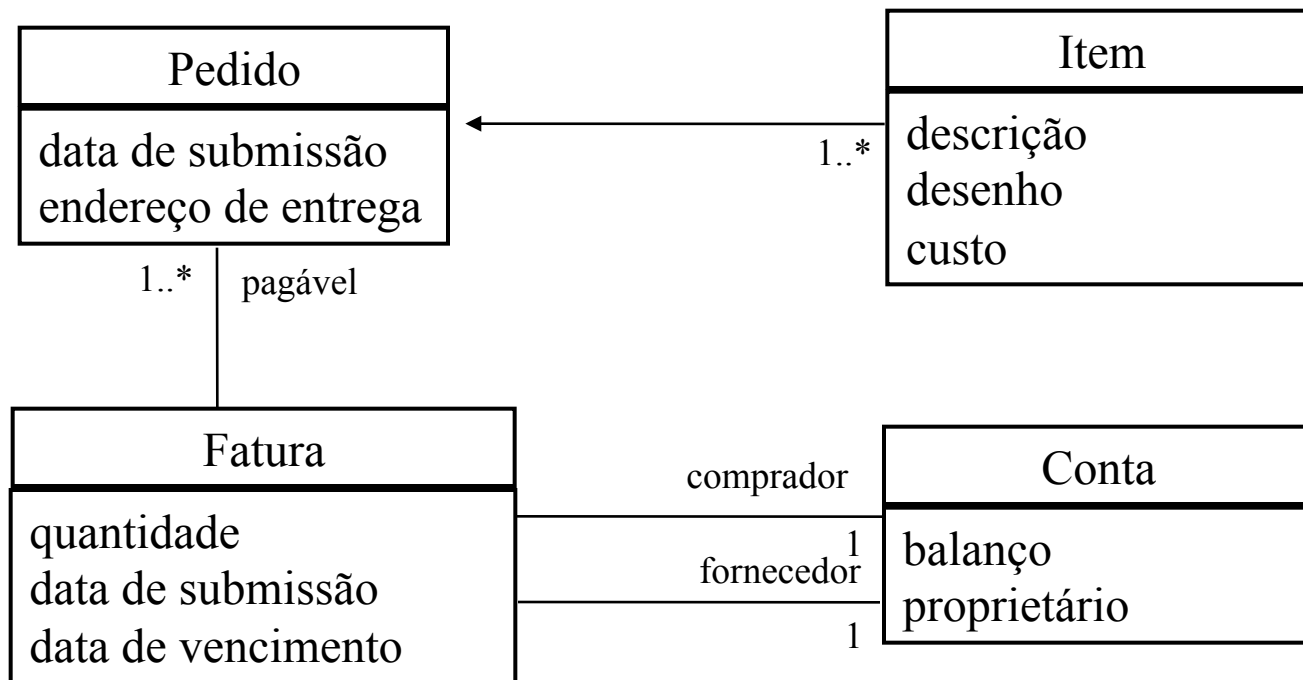
## 1. Entender o contexto do sistema

Para expressar o contexto de um sistema, o gerente de projeto e o arquiteto decidem o que será definido:

- um modelo do domínio: descreve os conceitos importantes do contexto como objetos do domínio e as ligações entre estes objetos;
- um modelo do negócio (casos de uso e objetos do negócio): especifica quais os processos do negócio serão suportados pelo sistema e as competências de cada processo (participantes, suas responsabilidades e operações executadas por eles);
- nenhum deles.

# Atividades do Workflow de Requisitos

**Modelo do domínio:** descreve os conceitos importantes do contexto como objetos do domínio e as ligações entre estes objetos.



- Em domínios pequenos, ao invés de definir um modelo do domínio, pode ser definido um glossário



# Atividades do Workflow de Requisitos

Alguns passos normalmente realizados no levantamento de requisitos:

## 1. Entender o contexto do sistema

## 2. Enumerar os requisitos candidatos

- Ideias mencionadas pelos clientes, usuários, analistas e desenvolvedores são enumeradas como requisitos candidatos.
- Cada requisito candidato possui um nome e uma breve explicação.



# Atividades do Workflow de Requisitos

Alguns passos normalmente realizados no levantamento de requisitos:

- 1. Entender o contexto do sistema**
- 2. Enumerar os requisitos candidatos**
- 3. Capturar os requisitos funcionais** (através de casos de uso)



# Atividades do Workflow de Requisitos

Alguns passos normalmente realizados no levantamento de requisitos:

## **3. Capturar os requisitos funcionais** (através de casos de uso)

### 3.1. Identificar os atores e casos de uso

- incluindo breve descrição dos casos de uso e diagrama de casos de uso

### 3.2. Dar prioridades aos casos de uso

### 3.3. Detalhar os casos de uso

- o detalhamento de um caso de uso complexo pode incluir diagrama de statechart, de atividade ou de interação

### 3.4. Definir a interface com o usuário

- projeto da interface lógica: elementos que representam os atributos dos casos de uso
- projeto e protótipo da interface física: sketches com os elementos identificados na interface lógica. Protótipos executáveis são construídos para as partes mais importantes e onde a usabilidade deve ser avaliada.



# Workflow de Requisitos

Alguns passos normalmente realizados no levantamento de requisitos:

**1. Entender o contexto do sistema**

**2. Enumerar os requisitos candidatos**


**3. Capturar os requisitos funcionais**

**4. Capturar os requisitos não-funcionais**

- Os requisitos não-funcionais específicos de um determinado caso de uso são conectados a ele.
- Os requisitos não-funcionais genéricos são especificados em uma lista de requisitos suplementares.



# Workflow de Análise



# Workflow de Análise

Durante o Workflow de Análise, os requisitos levantados no Workflow de Requisitos são refinados e estruturados.

## Objetivos:

- obter um entendimento mais preciso dos requisitos;
- obter uma descrição dos requisitos que seja fácil de manter e que ajude a estruturar o sistema como um todo.

# Workflow de Análise

Comparação entre o Modelo de Casos de Uso e o Modelo de Análise:

## Modelo de Casos de Uso

- Descrito usando a linguagem do cliente
- Visão externa do sistema
- Estruturado por casos de uso; define uma estrutura para a visão externa
- Usado como um contrato entre o cliente e os desenvolvedores que especifica o que o sistema deverá e não deverá fazer

## Modelo de Análise

- Descrito usando a linguagem do desenvolvedor
- Visão interna do sistema
- Estruturado por classes estereotipadas e pacotes; define uma estrutura para a visão interna
- Usado pelos desenvolvedores para entender como o sistema deverá ser projetado e implementado

# Workflow de Análise

## Modelo de Casos de Uso

- Pode conter redundâncias, inconsistências, etc.
- Captura a funcionalidade do sistema
- Define os casos de uso que serão analisados no modelo de análise

## Modelo de Análise

- Não deve conter redundâncias, inconsistências, etc.
- Define como realizar a funcionalidade dentro do sistema; serve como primeira aproximação do projeto
- Define as realizações dos casos de uso do modelo de casos de uso

# Artefatos do Workflow de Análise

- ➔ Classes de Análise
  - Classes de Interface (Boundary Classes)
  - Classes de Entidade
  - Classes de Controle

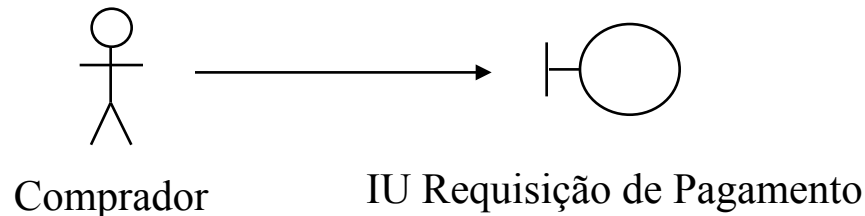
# Artefatos do Workflow de Análise

## Classes de Interface (Boundary Classes)

Usadas para modelar a interação entre o sistema e seus atores.

Essa interação, geralmente, envolve o recebimento e apresentação de informações e requisições dos atores e para os atores.

Representam abstrações de janelas, formulários, interfaces de comunicação, sensores, terminais, etc.



- A classe IU Requisição de Pagamento é usada para dar suporte à interação entre o Comprador e o caso de uso Pagar Fatura.



# Artefatos do Workflow de Análise

## Classes de Entidade

Usadas para modelar as informações que tem vida longa e geralmente são persistentes.

Podem ser derivadas das classes do modelo de negócio ou das classes do modelo de domínio.



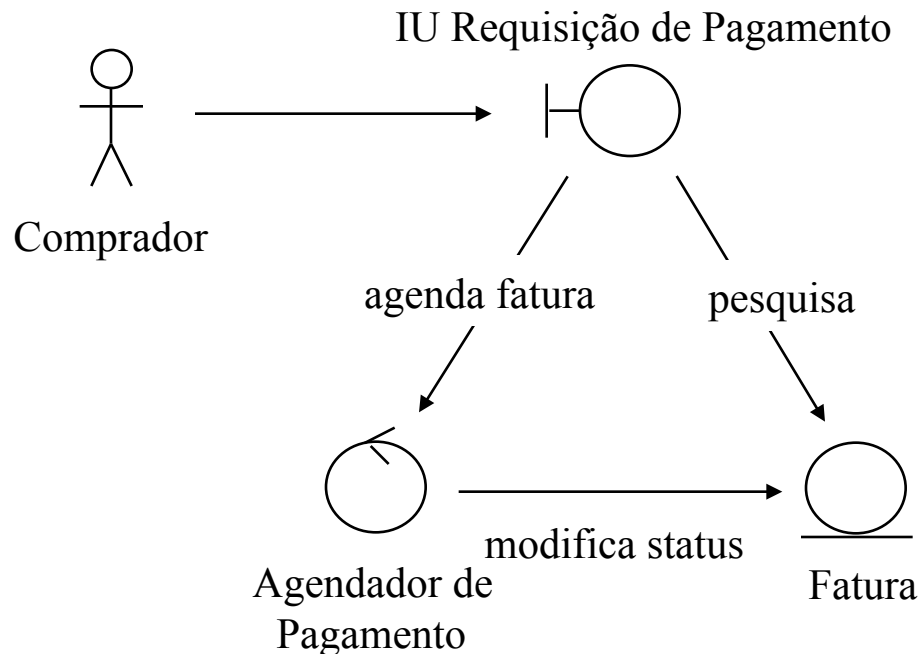
- A classe Fatura é usada para representar as faturas.

# Artefatos do Workflow de Análise

## Classes de Controle

Representam coordenação, sequência e controle de outros objetos.

São usados para encapsular o controle relacionado a um caso de uso específico.



- A classe Agendador de Pagamento é responsável pela coordenação entre IU Requisição de Pagamento e Fatura.

# Artefatos do Workflow de Análise


## → Classes de Análise

- Classes de Interface (Boundary Classes)
- Classes de Entidade
- Classes de Controle

## → Realização do Caso de Uso – Análise


- Diagramas de Classe
- Diagramas de Interação

## → Pacote de Análise



# Atividades do Workflow de Análise

1. Analisar a arquitetura
2. Analisar um caso de uso
3. Analisar uma classe
4. Analisar um pacote



# Atividades do Workflow de Análise

## 1. Analisar a arquitetura (arquiteto)

- Identificar os pacotes de análise
- Identificar as classes de entidade óbvias
- Identificar os requisitos especiais comuns que surgem durante a análise (ex: distribuição e concorrência, características de segurança)

# Atividades do Workflow de Análise

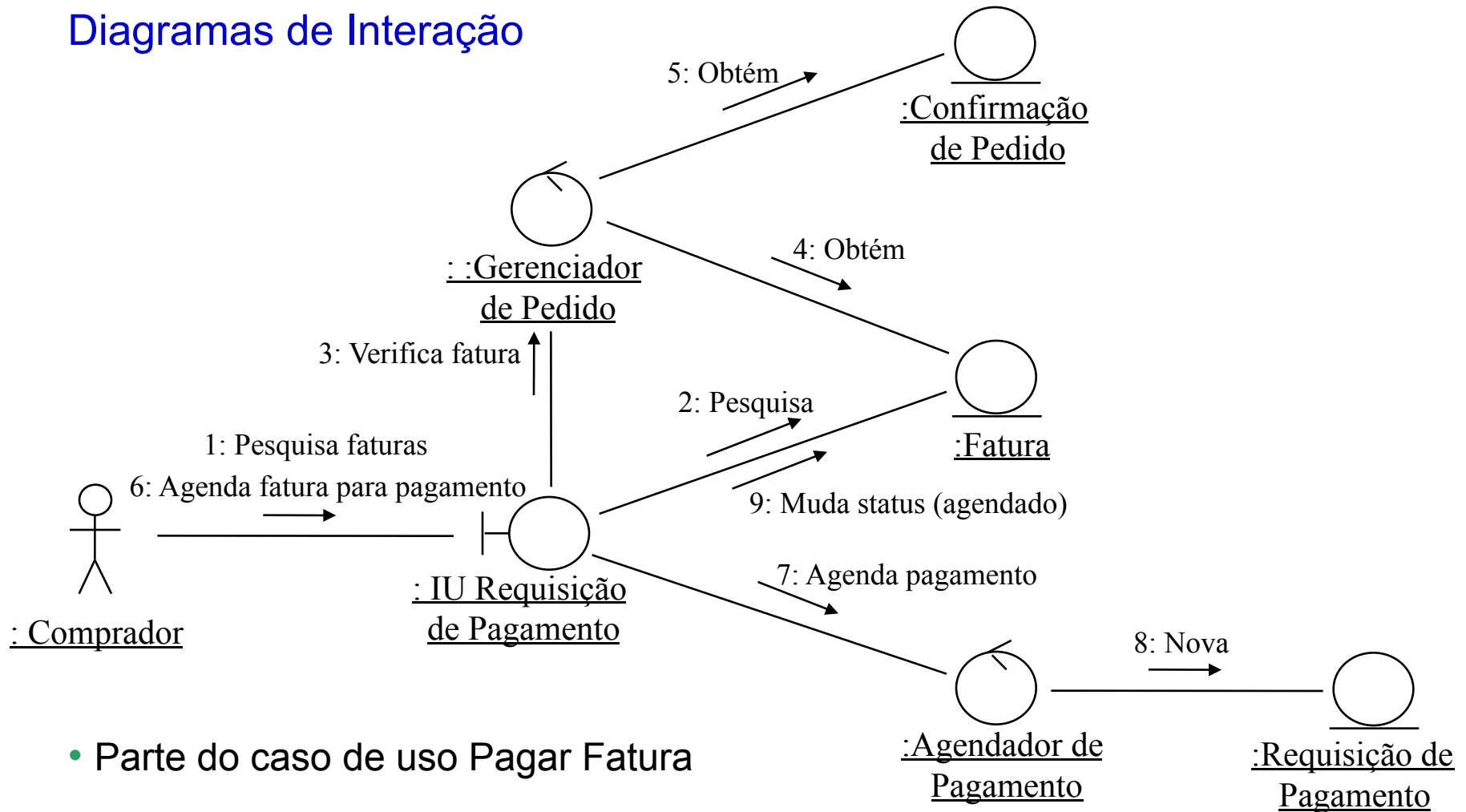
## 2. Analisar um caso de uso (engenheiro de caso de uso)

- Identificar as classes de análise do caso de uso
- Descrever as interações entre os objetos de análise

A descrição de como os objetos da análise interagem é feita através de diagramas de comunicação que contêm as instâncias dos atores participantes, os objetos de análise e seus links.

# Workflow de Análise

## Diagramas de Interação



- Parte do caso de uso Pagar Fatura

# Atividades do Workflow de Análise

## 2. Analisar um caso de uso (engenheiro de caso de uso)

- Identificar as classes de análise do caso de uso
- Descrever as interações entre os objetos de análise

A descrição de como os objetos da análise interagem é feita através de diagramas de comunicação que contêm as instâncias dos atores participantes, os objetos de análise e seus links.

- Capturar os requisitos especiais que surgem durante a realização do caso de uso (ex. a classe Fatura deve ser persistente)




# Atividades do Workflow de Análise

## 3. Analisar uma classe (engenheiro de componente)

- Identificar as responsabilidades através dos papéis que ela desempenha nas realizações de todos os casos de uso nos quais ela participa.

Exemplo: Responsabilidades da classe Agendador de Pagamento:


- Criar uma requisição de pagamento.
- Monitorar os pagamentos que foram agendados e enviar uma notificação quando o pagamento foi efetivado ou cancelado.
- Identificar os atributos, associações e agregações, generalizações
- Capturar os requisitos especiais



# Atividades do Workflow de Análise

## 4. Analisar um pacote (engenheiro de componente)

- Inclui a descrição das dependências.



# Workflow de Análise

## Diferentes maneiras de aplicar a análise:

1. O modelo de análise é utilizado para descrever os resultados da análise e é mantida a consistência dele durante todo o ciclo de vida do software.
2. O modelo de análise é utilizado para descrever os resultados da análise, mas ele é visto como uma ferramenta temporária e intermediária. Durante as fases de projeto (design) e implementação, o modelo de análise não é mais mantido.
3. O modelo de análise não é utilizado para descrever os resultados da análise.



# Workflow de Projeto

# Workflow de Projeto

Durante o Workflow de Projeto, é criado um plano detalhado (blueprint) para o modelo de implementação.

Comparação entre o Modelo de Análise e o Modelo de Projeto:

## Modelo de Análise


- Modelo conceitual: abstração do sistema e sem questões de implem.
- Projeto genérico (aplicável a vários projetos)
- Três estereótipos nas classes: controle, entidade e boundary
- Menos formal
- Pode não ser mantido durante o ciclo de vida completo do software

## Modelo de Projeto

- Modelo físico, porque é um plano detalhado da implementação
- Não é genérico, e sim específico para uma implementação
- Qualquer número de estereótipos, dependendo da ling. de implementação
- Mais formal
- Deverá ser mantido durante o ciclo de vida completo do software


# Artefatos do Workflow de Projeto

- ➔ Classes de Projeto
- ➔ Realização do Caso de Uso - Projeto
  - Diagramas de Classe
  - Diagramas de Interação
  - Fluxos de Eventos – Projeto (descrição textual para explicar os diagramas)
- ➔ Subsistemas de Projeto
- ➔ Modelo de Deployment



# Atividades do Workflow de Projeto

1. Projetar a arquitetura
2. Projetar um caso de uso
3. Projetar uma classe
4. Projetar um subsistema



# Atividades do Workflow de Projeto

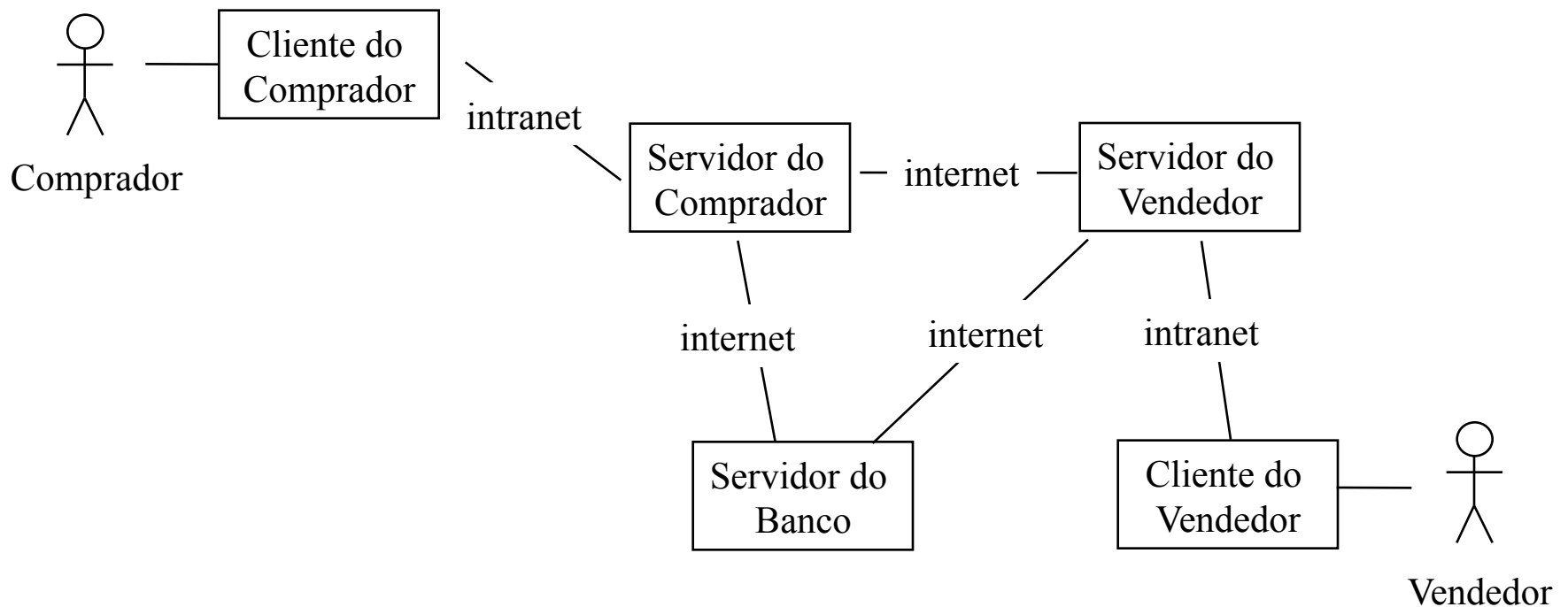
## 1. **Projetar a arquitetura** (arquiteto)


- Identificar os nodos e suas configurações de rede



# Workflow de Projeto

Identificação dos nodos e configurações de rede (Modelo de Deployment)





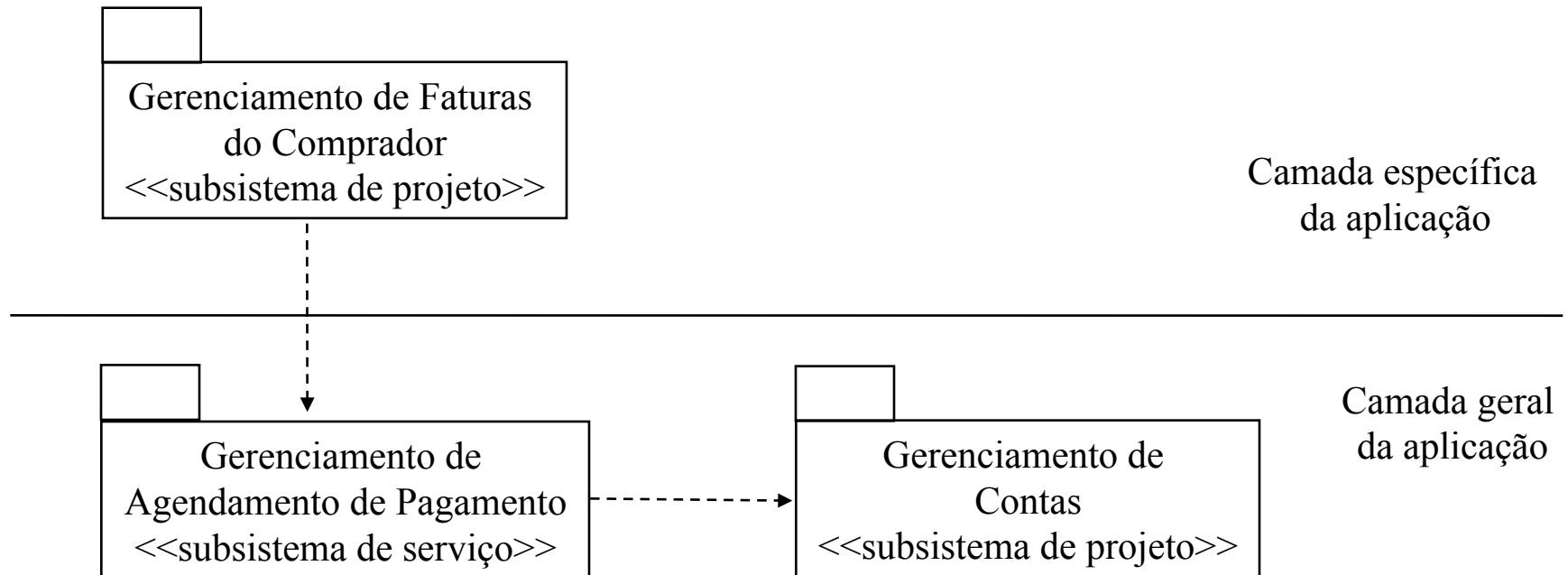
# Atividades do Workflow de Projeto

## 1. **Projetar a arquitetura** (arquiteto)

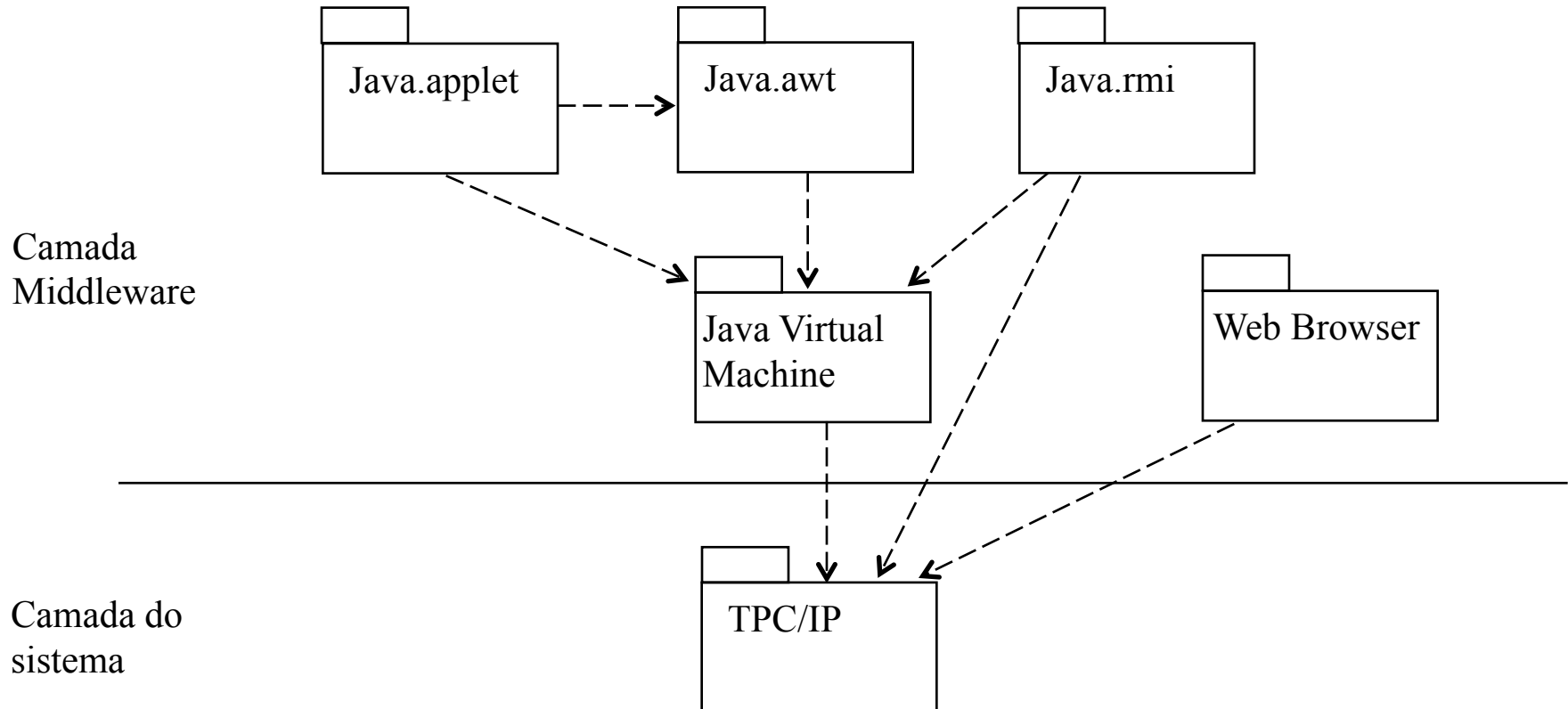
- Identificar os nodos e suas configurações de rede
- Identificar os subsistemas e suas interfaces


# Workflow de Projeto

Identificação dos subsistemas e suas interfaces:



# Workflow de Projeto






# Atividades do Workflow de Projeto

## 1. **Projetar a arquitetura** (arquiteto)

- Identificar os nodos e suas configurações de rede
- Identificar os subsistemas e suas interfaces
- Identificar as classes de projeto significantes (ex. identificação das classes de projeto a partir das classes de análise)



# Atividades do Workflow de Projeto


## 2. **Projetar um caso de uso** (engenheiro de caso de uso)

- Identificar as classes de projeto do caso de uso (diagrama de classes)
- Descrever as interações entre os objetos (diagramas de sequência)
- Identificar os subsistemas participantes que contêm classes de projeto que participam do caso de uso em questão
- Capturar os requisitos de implementação (ex. um objeto deverá ser capaz de manipular 10 clientes compradores sem um delay perceptível.)

# Atividades do Workflow de Projeto

## 3. **Projetar uma classe** (engenheiro de componente)

- Identificar as operações, os atributos, as associações e agregações, e as generalizações.
- Descrever os métodos: podem ser especificados usando linguagem natural ou pseudocódigo, mas na maioria das vezes, métodos não são especificados durante o projeto.
- Descrever os estados: para os objetos de projeto que são controlados pelo estado, é importante usar um diagrama de statechart.



# Atividades do Workflow de Projeto


## 4. **Projetar um subsistema** (engenheiro de componente)

- Assegurar que o subsistema satisfaz a realização das operações definidas pelas interfaces que ele provê.





# Workflow de Implementação



# Atividades do Workflow de Implementação

Durante o Workflow de Implementação, o sistema é implementado a partir do resultado do projeto.

Atividades:


## 1. **Arquitetar a Implementação** (arquiteto)

- As classes de projeto mais significantes para a arquitetura são identificadas e mapeadas para os nodos.

## 2. **Integrar o sistema** (integrador do sistema)

- Criação de um plano de integração que descreve os *builds* de uma iteração e como serão integrados.

## 3. **Implementar um subsistema** (engenheiro de componente)



# Atividades do Workflow de Implementação

**4. Implementar uma classe** (engenheiro de componente)

**5. Executar os testes de unidade** (engenheiro de componente)

- Teste de especificação (teste da caixa preta) e teste de estrutura (teste da caixa branca).
- Também são feitos testes de performance e capacidade.



# Workflow de Testes

# Atividades do Workflow de Teste

Durante o workflow de teste, o resultado da implementação é testado.

Atividades:

## 1. Planejar e Projetar o teste (projetista de teste)

- Planejamento dos testes de uma iteração: como rodá-los, quando rodá-los, quando terminar os testes, estimar os recursos humanos e de sistema necessários, agendar os testes.
- Os casos de testes são baseados nos diagramas de interação dos casos de uso.

## 2. Implementar o teste (engenheiro de componente)

## 3. Executar o teste de integração (testador de integração)

## 4. Executar o teste de sistema (testador de sistema)

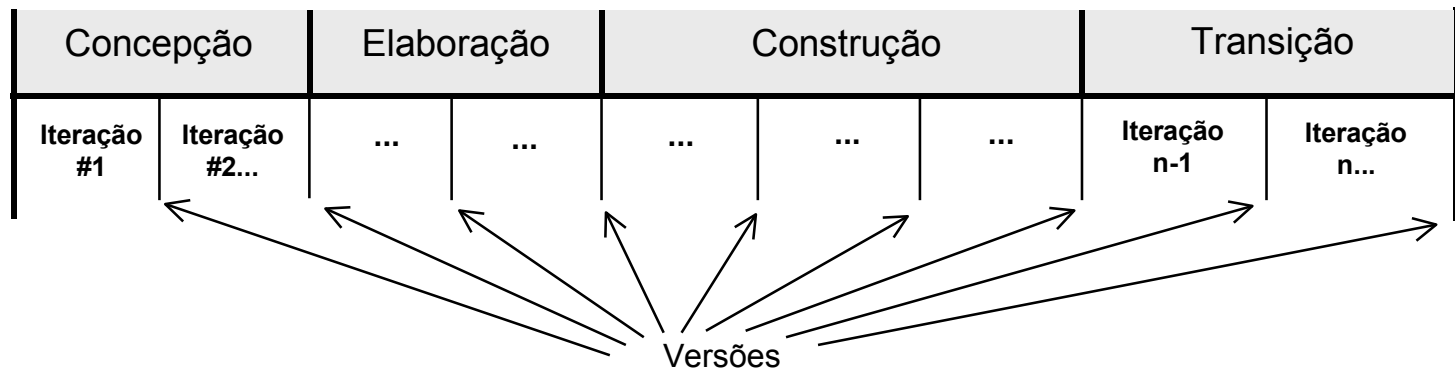
## 5. Avaliar o teste (projetista de teste)




# Rational Unified Process (RUP)

# RUP

- O RUP é uma instância específica do Processo Unificado.
- O RUP é um produto de processo. É desenvolvido e mantido pela Rational Software e integrado com suas ferramentas.
- O RUP também é um framework de processo que pode ser adaptado e estendido.





# Disciplinas do RUP

## Disciplinas de engenharia:

- Disciplina de Modelagem do Negócio
- Disciplina de Requisitos
- Disciplina de Análise e Projeto
- Disciplina de Implementação
- Disciplina de Teste
- Disciplina de Deployment

## Disciplinas de suporte:

- Disciplina de Gerenciamento de Projeto
- Disciplina de Gerenciamento de Configuração
- Disciplina de Ambiente