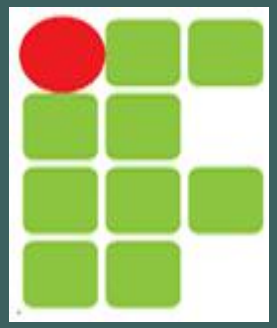


Aula 04

Análise e Projeto de Software

Métodos e Técnicas em Engenharia de Software

Profa. Alexandra

Prof. Thiago

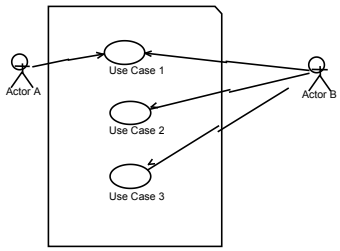
Pós-Graduação em Gestão de Sistemas de Informação

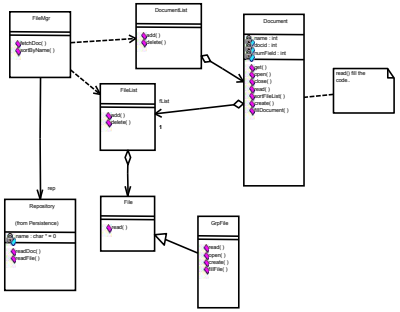
Objetivos

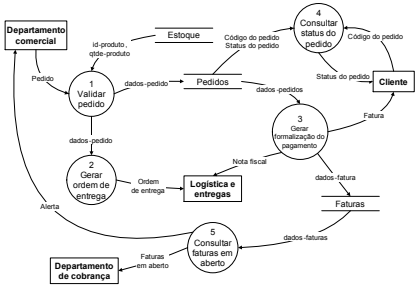
- Após esta aula, você deverá ser capaz de:
  - Compreender o objetivo das atividades de Análise e Projeto de Software
  - Descrever a diferença entre os paradigmas de Análise
  - Definir arquitetura de software e conceituar estilos e padrões arquiteturais

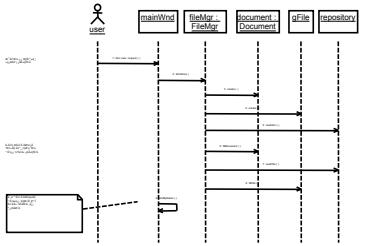
Análise e Projeto

As atividades de **Análise** estão relacionadas à obtenção de uma **visão detalhada e precisa** sobre os **requisitos do sistema** e sua representação por meio de **modelos** (gráficos ou não) da **estrutura e comportamento** do sistema pretendido









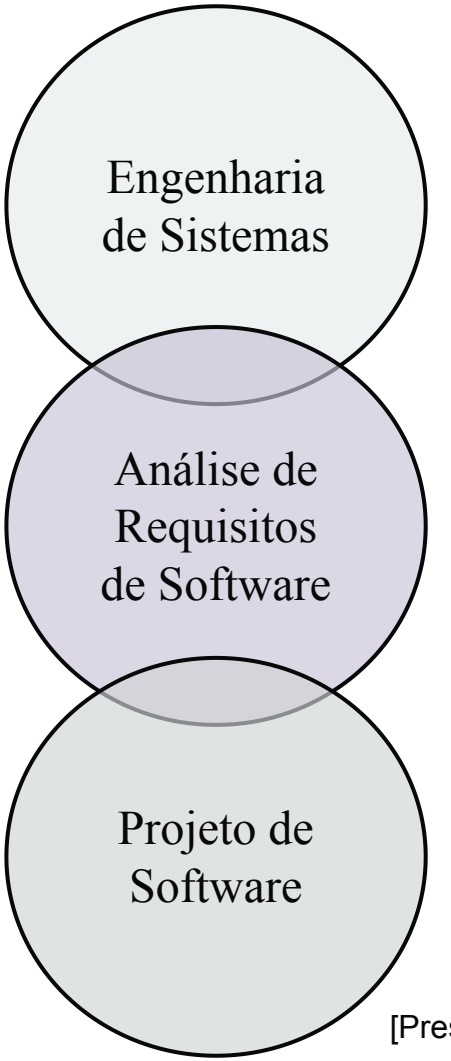
Análise e Projeto

No entanto, **não seria desejável** que essa representação do sistema pretendido **já considerasse todos os aspectos técnicos** da solução (por exemplo, hardware, restrições de comunicação, segurança ou processamento)

Por isso **a solução fornecida como resultado da Análise abstrai tais aspectos – pode-se dizer que ela traz uma solução "abstrata" para o problema**

A atividade de **Projeto** (ou Design) é alimentada pelo resultado da Análise e tem como objetivo **adequar o modelo do sistema às restrições técnicas e "do mundo real"** da aplicação

Análise e Projeto



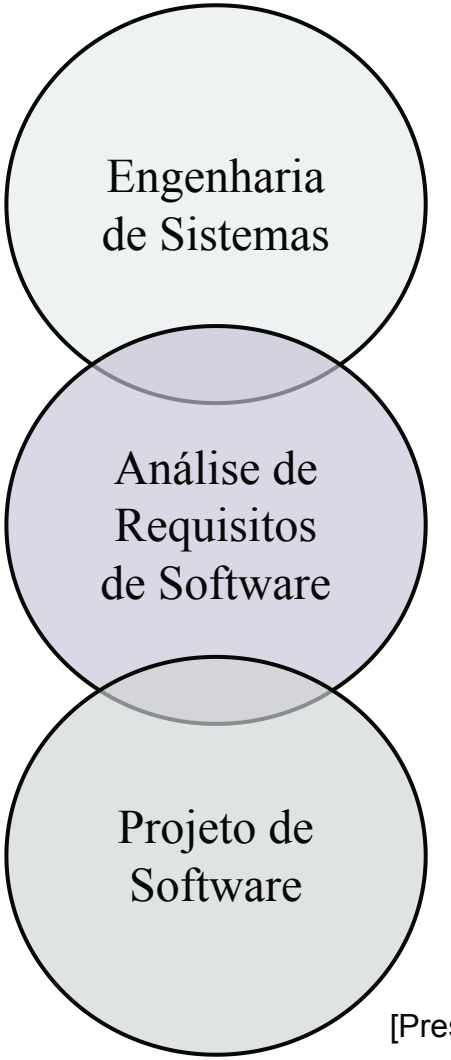
Foco nos processos e na estrutura organizacional

Foco em especificar e modelar uma solução computacional abstrata para o problema

Foco em adequar a solução às restrições técnicas e do negócio

[Pressman, 2001]

Análise e Projeto

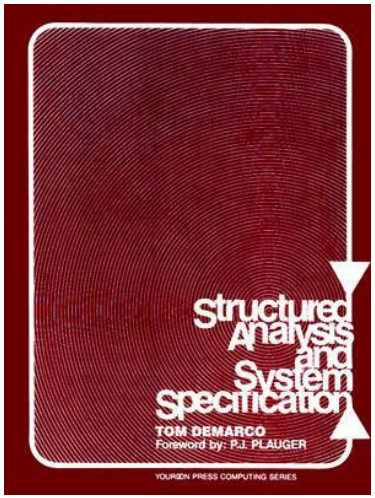


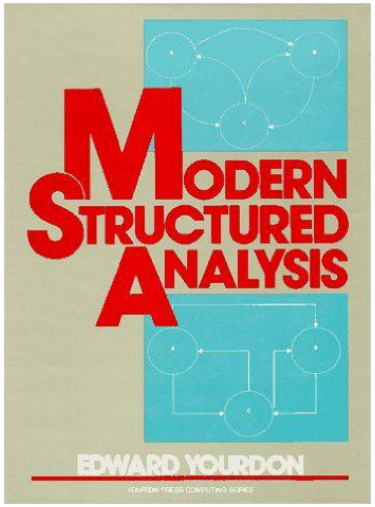
[Pressman, 2001]

É possível afirmar que a Análise atua como uma "ponte" entre a Engenharia de Sistemas e o Projeto de Software

Análise – um breve histórico

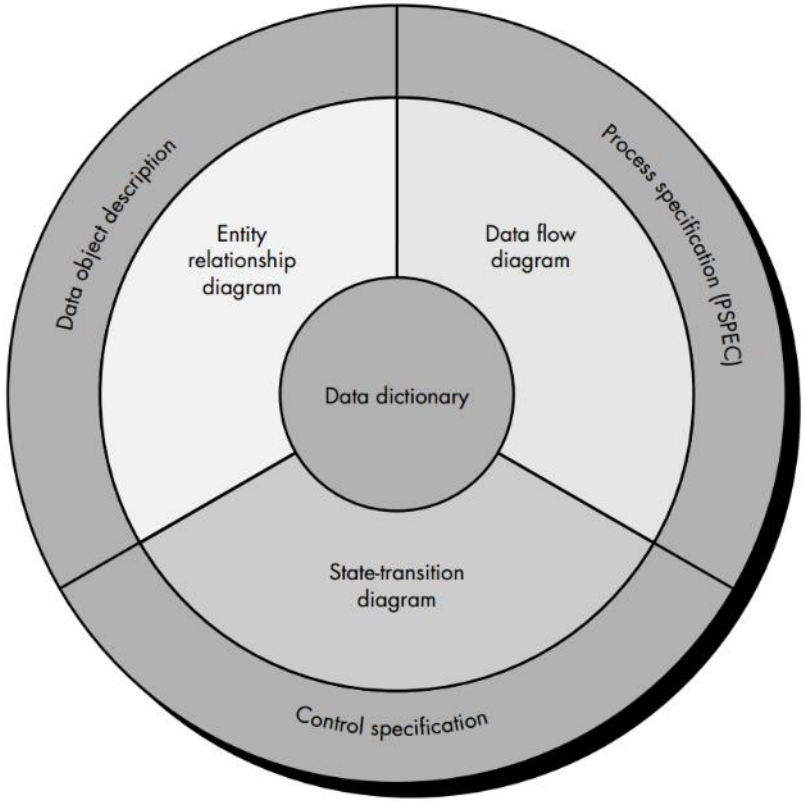
- O primeiro paradigma para as atividades de Análise é denominado **Análise Estruturada**
- O termo foi criado por Douglas Ross e popularizado por DeMarco em seu livro de 1979
- 1984 – A Análise Estruturada é estendida
  - McMenamin e Palmer - Essential Structured Analysis
- 1989 – Análise Estruturada atinge seu pico
  - Yourdon publica *Análise Estruturada Moderna*, e incorpora os Diagramas Entidade-Relacionamento de Chen
- 1991 – Yourdon muda para a Análise Orientada a Objetos
- 1995 – 38% das organizações utilizavam Análise Estruturada





Análise Estruturada

- A Análise Estruturada se baseia em três representações do sistema:
  - Especificação de processos
  - Especificação de controle
  - Descrição de objetos de dados



Análise Estruturada

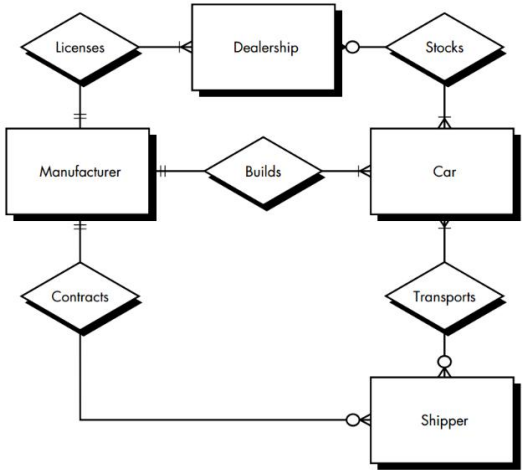
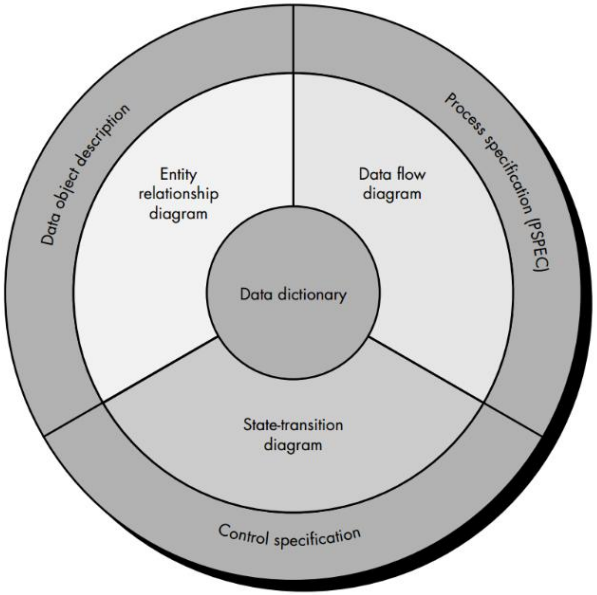


Diagrama Entidade-Relacionamento



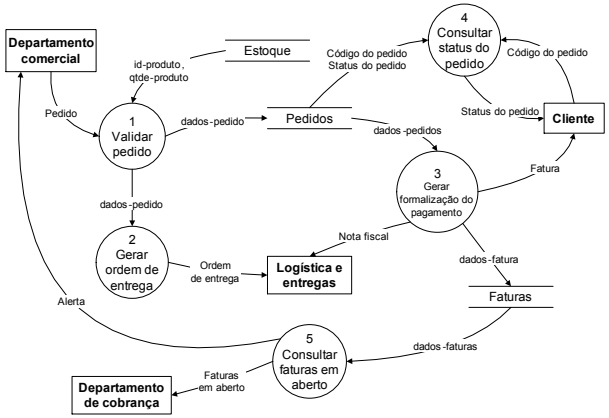


Diagrama de Fluxo de Dados

Diagrama de Transição de Estados

Análise Estruturada

- O foco dos modelos está nos **processos de transformação da informação**

- Já está ultrapassada?
- Não exatamente…
  - A manutenção de sistemas legados exige atualizar modelos feitos segundo A.E.
  - A visão de fluxo de dados da A.E. tem sido empregada, com atualizações, para representar a **modelagem de negócio** (BPMN, IDEF0)
  - São visões complementares de um sistema moderno

Análise – um breve histórico

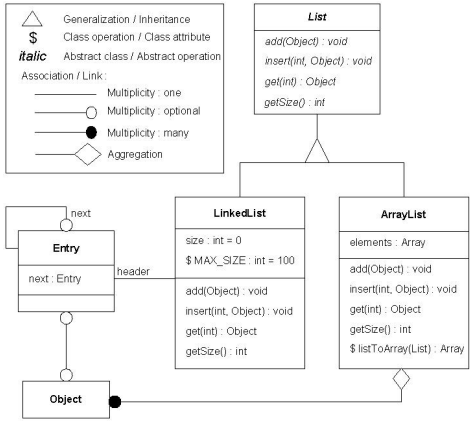
- Os anos 1970 e 1980 viram o surgimento e desenvolvimento das **linguagens orientadas a objeto** e sua aplicação ao desenvolvimento de sistemas reais

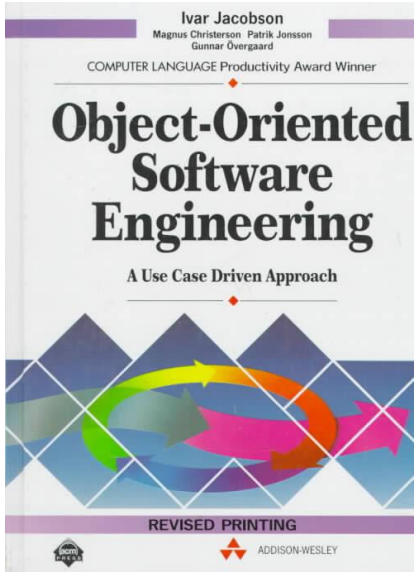
- Quais as vantagens (potenciais)?
  - Maior poder de expressão das linguagens;
  - Maior potencial de manutenção e reutilização de código

Análise – um breve histórico

- A Engenharia de Software acompanhou o desenvolvimento das tecnologias OO, desenvolvendo técnicas para dar suporte ao processo de desenvolvimento com essas tecnologias

  - 1990: Abordagem Coad / Yourdon
  - 1991: Rumbaugh et al. desenvolvem a OMT (Object Modelling Technique)
  - 1992: Ivar Jacobson desenvolve a metodologia OOSE na empresa Objectory AB, que introduz o conceito de casos de uso
  - 1994: Abordagem de Grady Booch, precursora de técnicas que seriam incorporadas ao RUP



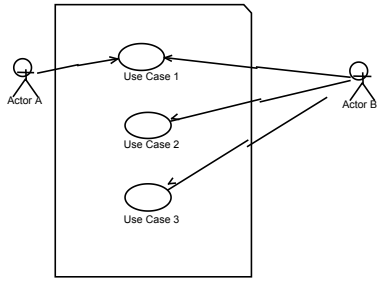


Análise – um breve histórico

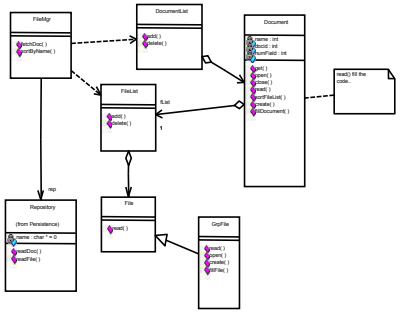
- 1994-1997
  - Com a compra da Objectory AB pela Rational Software, Booch, Rumbaugh e Jacobson começam a trabalhar na definição de uma linguagem unificada para modelagem de sistemas OO
  - O resultado desse trabalho é a **UML ➔ Unified Modelling Language**
  - Desde 1997 é adotada como padrão pelo OMG (Object Management Group)

Análise Orientada a Objeto

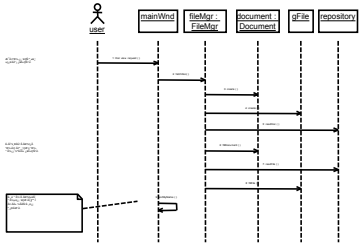
- **Visão dos requisitos**: baseada em casos de uso, uma representação da interação entre usuário e sistema



- **Visão estrutural**: representa classes, sua estrutura e suas relações

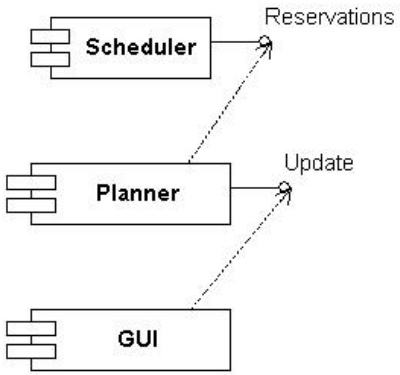


- **Visão comportamental**: mostra a interação dinâmica entre objetos em tempo de execução

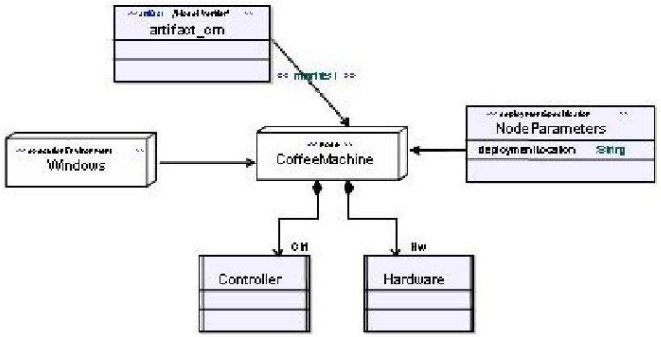


Análise Orientada a Objeto

- **Visão de implementação**: aspectos estruturais e comportamentais do sistema da forma como serão construídos



- **Visão do ambiente**: aspectos estruturais e comportamentais do ambiente no qual o sistema será implementado e representado



Princípios de Análise

Segundo Pressman (2001), a atividade de análise é guiada pelos seguintes princípios, independentemente do paradigma adotado:

❶ O **domínio de informação** de um problema deve ser compreendido e representado

❷ As **funcionalidades** do software a ser implementado devem ser definidas

❸ O **comportamento** do software (como consequência de eventos externos) deve ser representado

❹ Os modelos que apresentam informação, função e comportamento devem ser particionados de forma que detalhes são revelados em **camadas ou de forma hierárquica**

❺ O processo de análise parte **de informações essenciais em direção a detalhes** de implementação

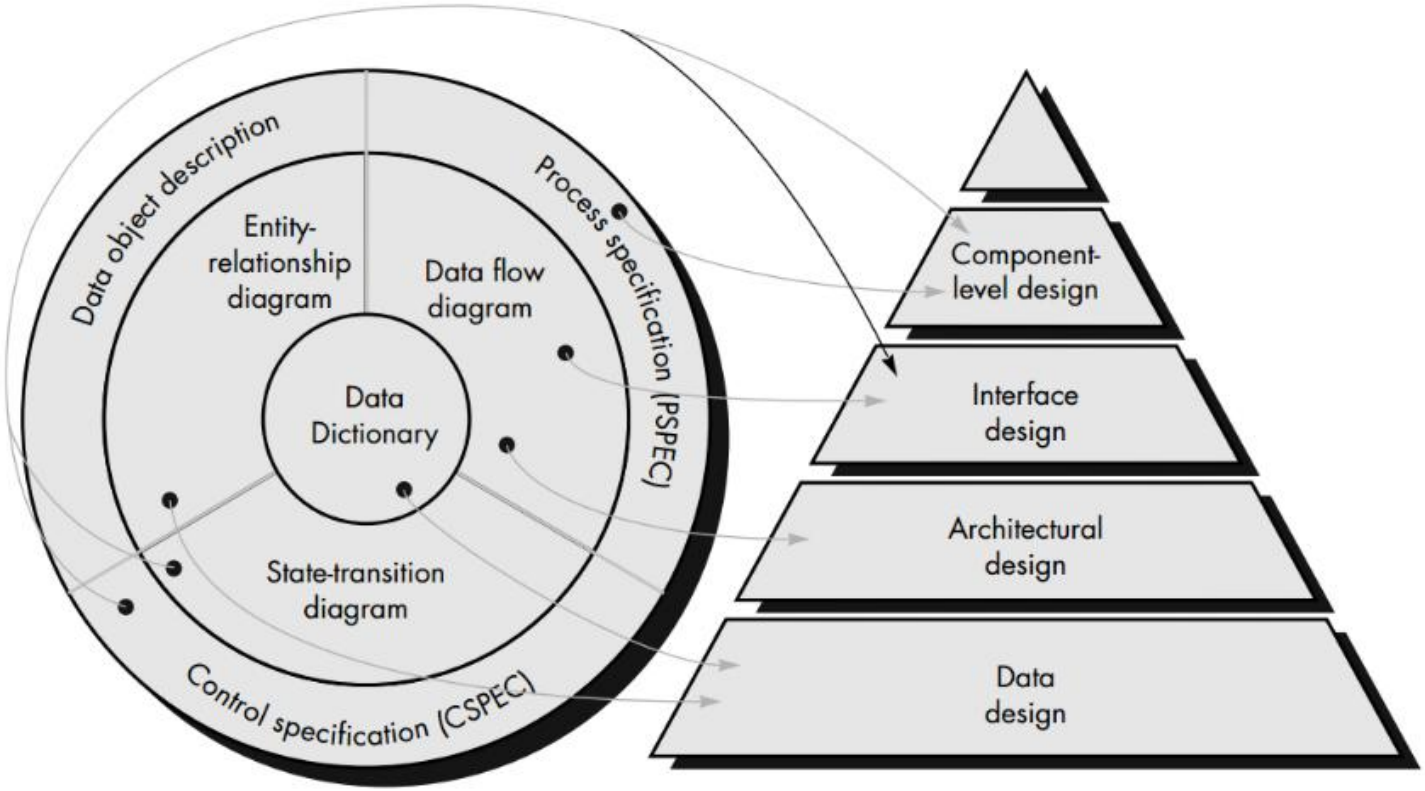
Projeto de Software

- Tem como objetivo transformar o modelo de Análise em uma representação que seja implementada usando uma linguagem de programação

- Pode ser considerada a primeira de três atividades técnicas – projeto, implementação e teste

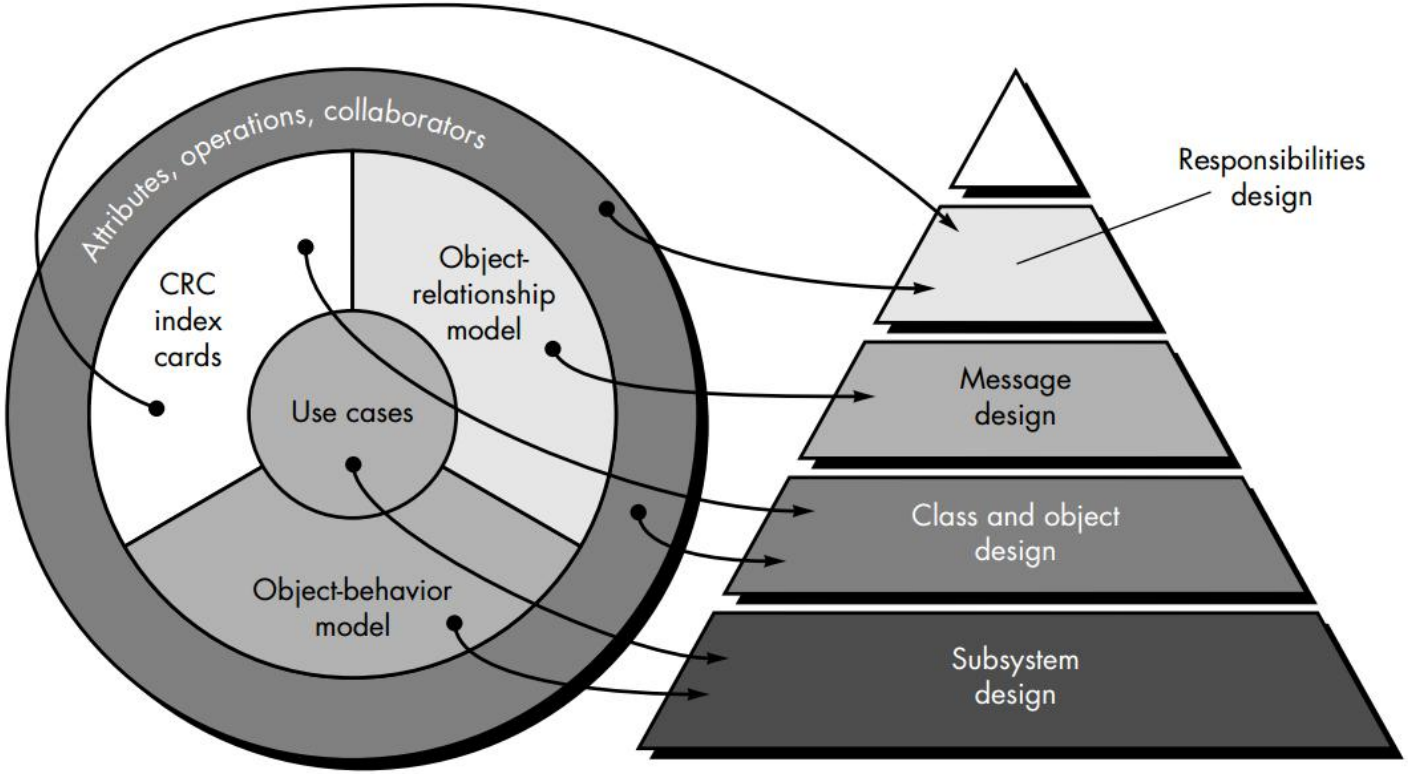
Projeto estruturado

O projeto vinculado ao paradigma de Análise Estruturada se inicia a partir dos artefatos produzidos naquela etapa



Projeto orientado a objeto

O projeto vinculado ao paradigma da Análise Orientada a Objeto segue os mesmos procedimentos



Projeto arquitetural

- Chamamos de **projeto arquitetural** o processo que identifica quais subsistemas compõem o sistema e qual é o *framework* de comunicação entre eles

- Entrada
  - Os modelos do sistema gerados na fase de análise

- Saída
  - Descrição da arquitetura do software, composta de uma série de representações gráficas dos modelos do sistema, associadas a texto descritivo

Vantagens

Algumas vantagens da definição explícita de uma arquitetura do sistema:

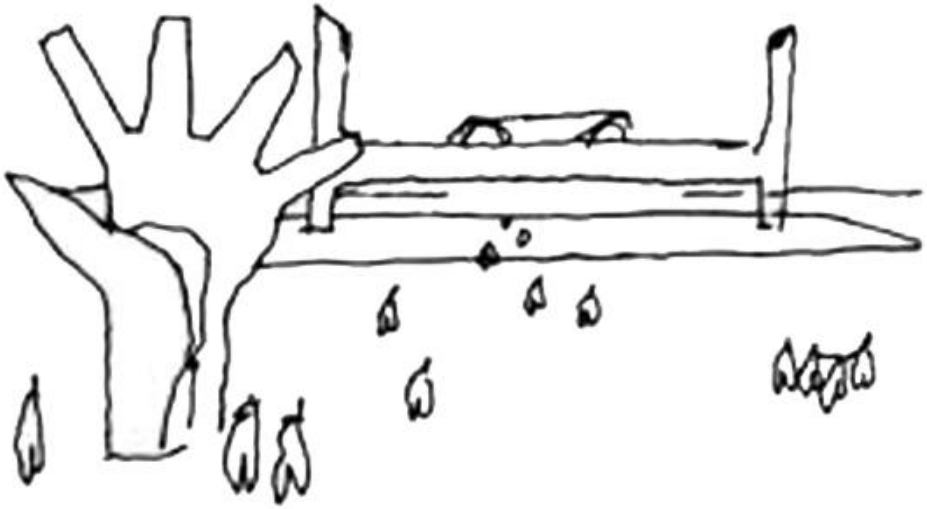
- Facilita a comunicação com o cliente, pois é uma representação de alto nível do sistema
- Permite a verificação de alguns requisitos não-funcionais, tais como desempenho, confiabilidade e facilidade de manutenção
- A arquitetura do sistema pode ser **reutilizada** em sistema com requisitos similares

Definição alternativa

Arquitetura é a visão da modelagem na qual você não pode eliminar mais nada sem deixar de entender o propósito do software

Adaptado de Phillipe Krutchen – The Rational Unified Process: An Introduction

Analogia com a arquitetura “tradicional”



Esboços de Oscar Niemeyer para o Memorial da América Latina – São Paulo – SP

O processo de projeto arquitetural

- Sommerville define três atividades no processo de projeto arquitetural:

  - **Estruturação do sistema**
    - Decompor o sistema em um conjunto de subsistemas e definir a comunicação entre eles

  - **Modelos de controle**
    - Estabelecer um modelo do fluxo de controle entre as diferentes partes do sistema

  - **Decomposição em módulos**
    - Os subsistemas identificados são decompostos em módulos

Subsistemas e módulos

- Apesar de, em muitos contextos, não existir uma distinção clara entre sub-sistema e módulo, utilizaremos a definição de (Sommervile, 2004):

  - **Subsistema**: parte do sistema principal que pode ser considerada um sistema por si só e não depende dos serviços providos por outros sub-sistemas. São compostos por módulos e tem interfaces definidas de comunicação com outros subsistemas.
  - **Módulo**: componente do sistema que provê um ou mais serviços para outros módulos mas não pode ser considerado um sistema independente.

Arquitetura e requisitos não-funcionais

Decisões tomadas no projeto da arquitetura são fortemente influenciadas por requisitos não-funcionais do sistema:

- **Desempenho**
  - Operações importantes devem ficar dentro do menor número possível de subsistemas para minimizar o *overhead* da comunicação
- **Proteção**
  - Usar arquitetura em camadas com as operações relacionadas com a segurança nas camadas mais internas
- **Segurança**
  - Concentrar os componentes relacionados com a segurança

Arquitetura e requisitos não-funcionais

- **Disponibilidade**
  - Incluir componentes redundantes que permitam substituição e atualização sem a interrupção do sistema

- **Facilidade de manutenção**
  - Criar componentes de menor granularidade que possam ser modificados rapidamente

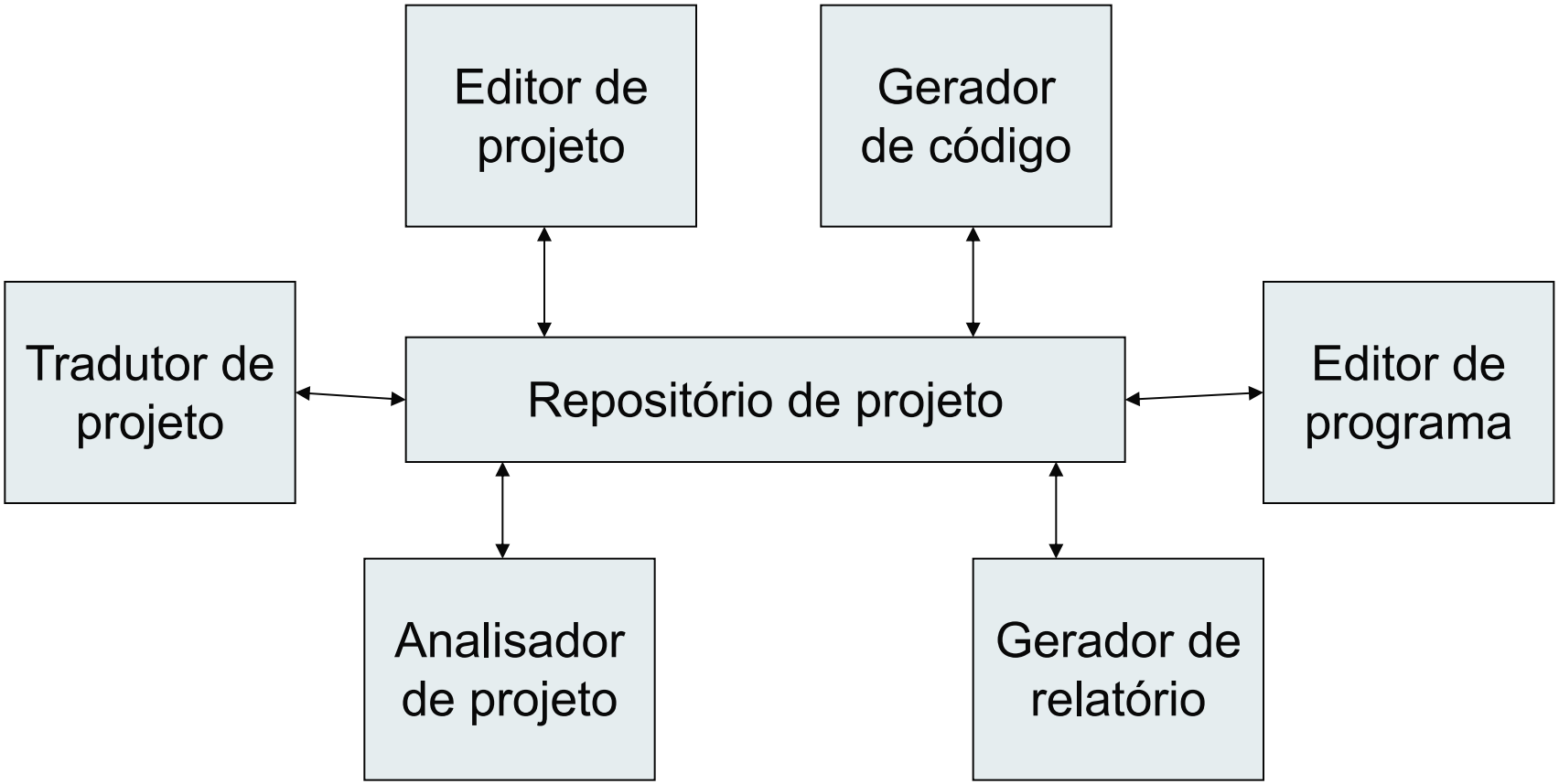
Estilos arquiteturais

Uma forma de orientar a definição da arquitetura é seguir um estilo arquitetural pré-definido (Pressman, 2001):

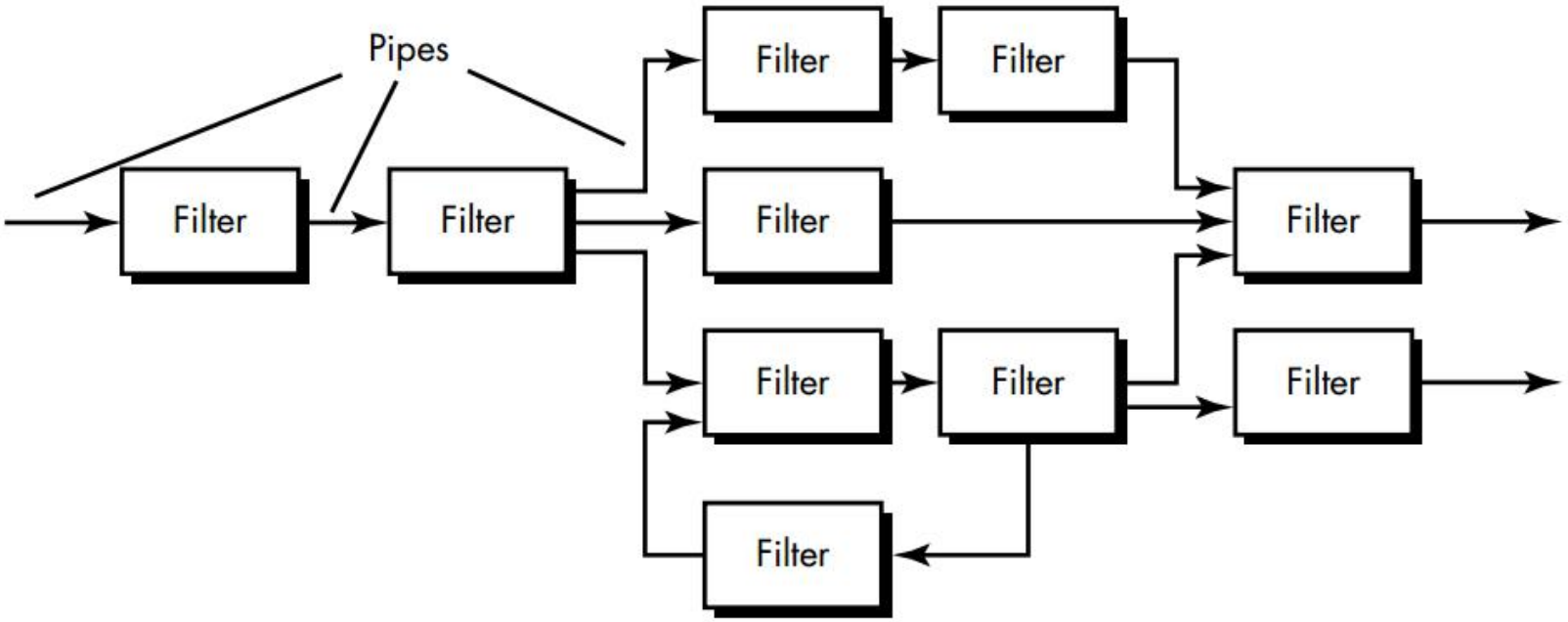
- **Arquitetura centrada em dados**
- **Arquitetura de fluxo de dados**
- **Arquitetura baseada em chamada e retorno**
- **Arquitetura em camadas**

Arquitetura centrada em dados

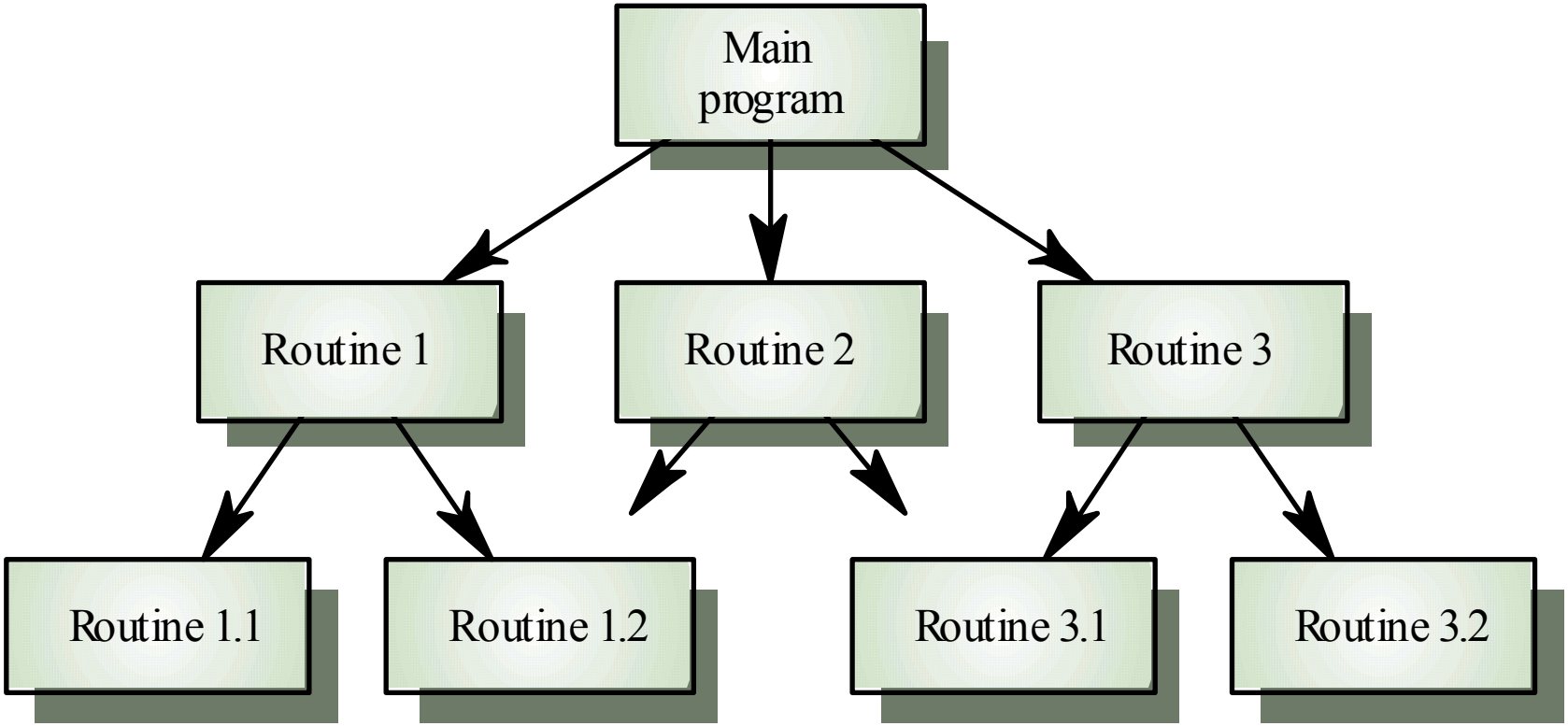
- Exemplo: sistema de ferramentas CASE



Arquitetura de fluxo de dados



Arquitetura baseada em chamada e retorno



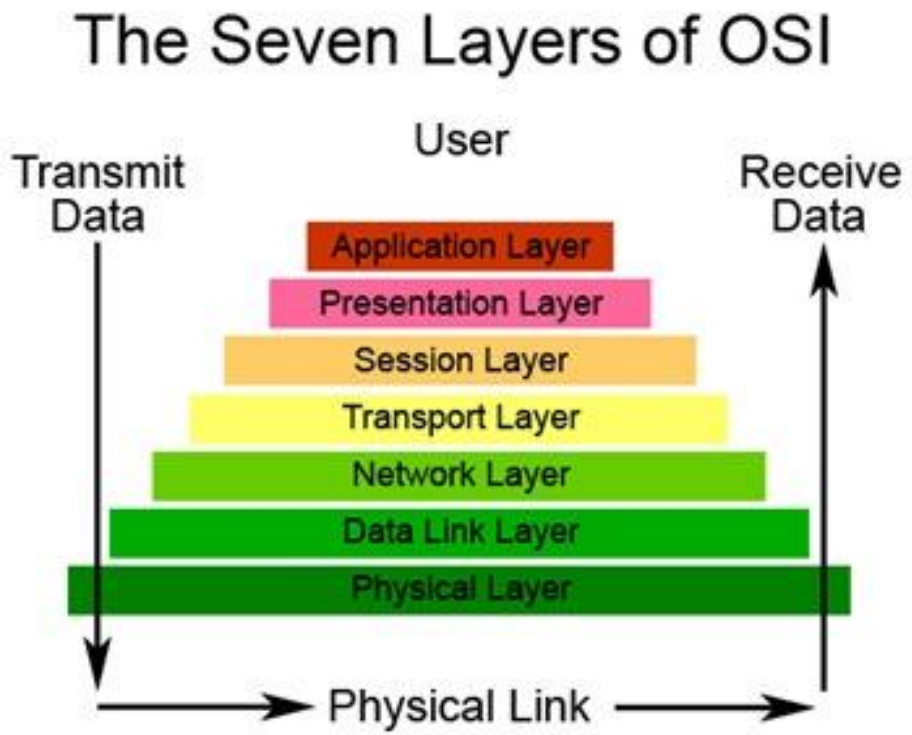
Arquitetura em camadas

Camada de apresentação ou aplicação

Camada de negócio (ou de regras de negócio)

Middleware

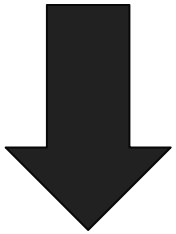
Camada de Banco de Dados



Tendências em Análise e Projeto de Software

Conceito de padrão

Padrão

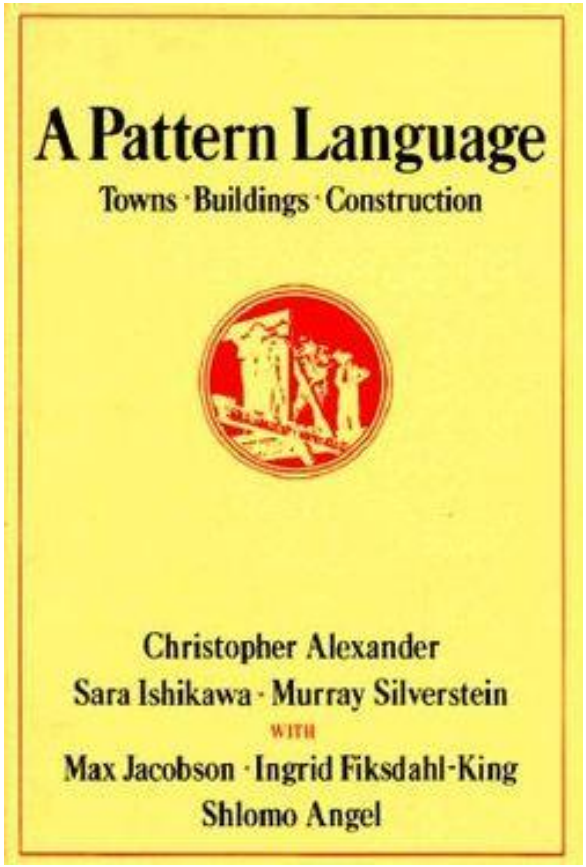


Descrição de um problema que ocorre muitas vezes, e de uma solução genérica que pode ser aplicada toda vez que o problema ocorre

Histórico

- A idéia de padrões foi inicialmente desenvolvida na área de arquitetura

- Christopher Alexander escreveu em 1977 o livro *A Pattern Language: Towns, Buildings, and Construction*, que serviu de inspiração para os desenvolvedores de software



Exemplos de padrões na arquitetura

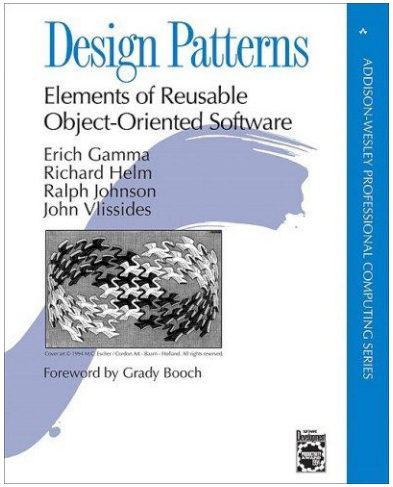
- **Problema**: Em lugares nos quais há ação e movimento de pessoas, os lugares mais convidativos são aqueles altos o suficiente para dar um ponto de vantagem, e baixos o suficiente para colocar as pessoas em ação

- **Solução:** escadas unindo níveis, diretamente acessíveis para as pessoas

http://vasarhelyi.eu/books/A_pattern_language_book

Padrões em Análise e Projeto

- **Design Patterns**
  - Aplicados em um nível de granularidade mais “fino” (classes)
  - Muito influenciados pelo trabalho dos padrões GoF (Gang of Four)



- **Padrões arquiteturais**
  - Semelhantes aos estilos arquiteturais mencionados anteriormente
  - Soluções para domínios específicos (ex: Model-View-Controller) ou tecnologias específicas (ex: Business Intelligence, J2EE,

Modelagem ágil

- Ambler (2001) propôs uma abordagem às atividades de análise e projeto denominada Modelagem Ágil

- Baseada nos valores e princípios das metodologias ágeis, propõe práticas colaborativas e integradas ao desenvolvimento

Modelagem ágil

Valores

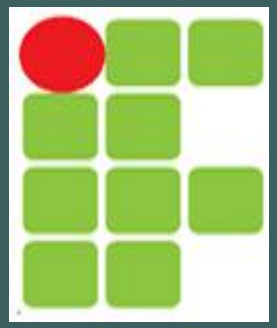
- Comunicação
- Simplicidade
- Feedback
- Coragem
- Humildade

Princípios suplementares

- Conteúdo é mais importante que a representação
- Todos podem aprender com os outros
- Conheça seus modelos
- Adaptação local
- Comunicação aberta e honesta
- Trabalhe com os instintos das pessoas

Princípios-chave

- Software é seu objetivo principal
- Organizar o próximo esforço é seu objetivo secundário
- Carregue pouco peso em sua jornada
- Assuma a simplicidade
- Acolha as mudanças
- Faça mudanças incrementais
- Modele com um propósito
- Use múltiplos modelos
- Trabalhe com qualidade
- Maximize o investimento dos stakeholders



Obrigado!

Profa. Alexandra Aparecida de Souza

Prof. Thiago Schumacher Barcelos