



Arquitetura de Software II – Requisitos Arquiteturais

Tópicos Avançados de
Programação Orientada a Objetos
MAC 413 / 5715

Prof. Fabio
IME / USP

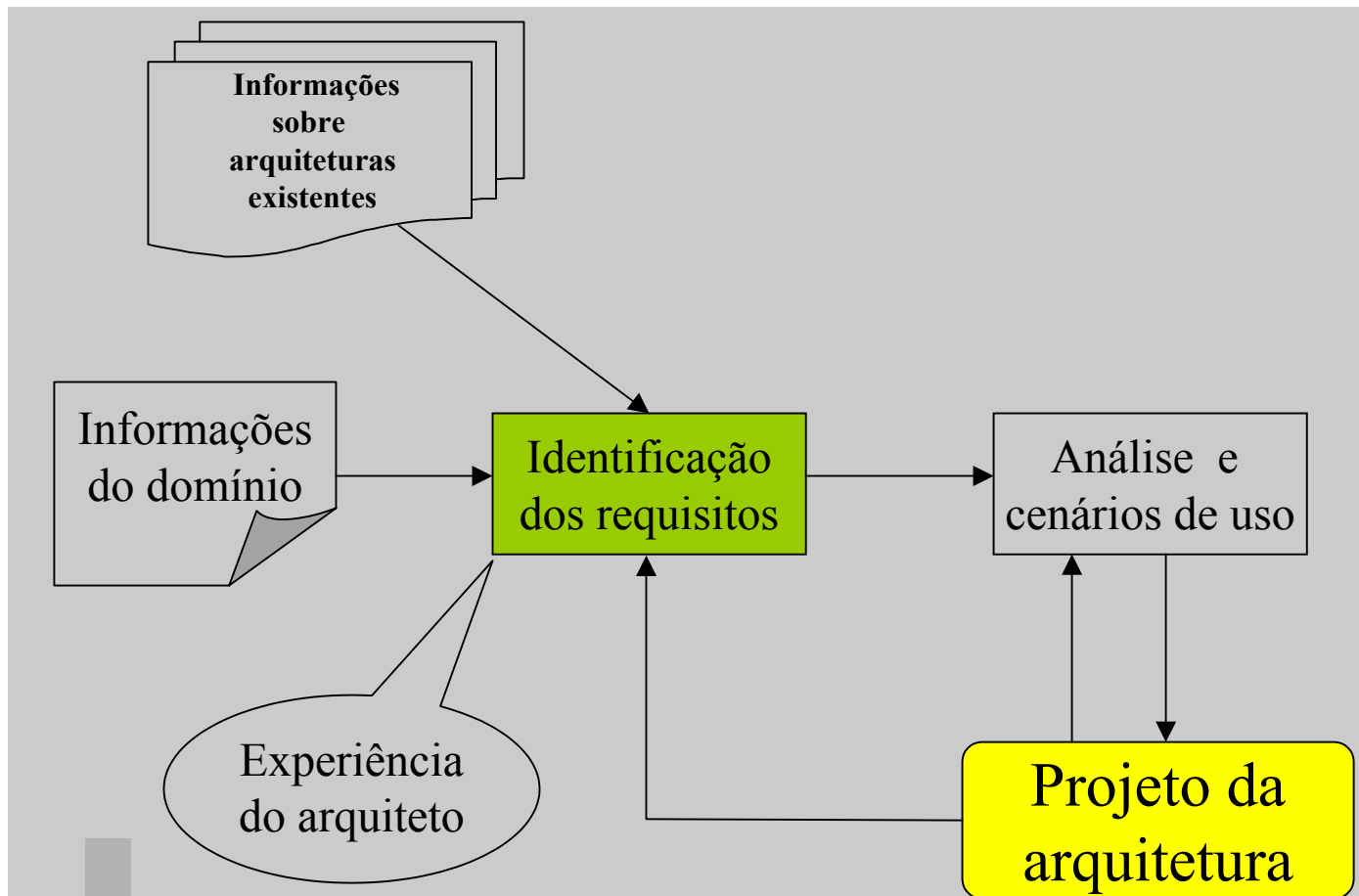


Requisitos Arquiteturais

Antes de projetar uma Arquitetura:

- ◆ modelagem do problema
- ◆ identificação dos requisitos
 - funcionais
 - não-funcionais

Identificação de Requisitos Arquiteturais





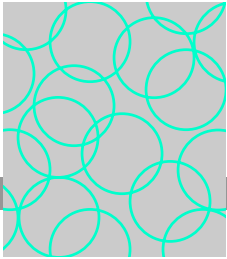
Atributos de Projeto

- ◆ Norteiam o processo de desenvolvimento de software
- ◆ Visam satisfazer os requisitos arquiteturais
- ◆ São qualidades desejáveis em sistemas de software:
 - Separação de Interesses
 - Abstração
 - Modularidade
 - Compartilhamento de Recursos



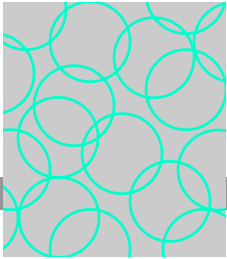
Separação de Interesses

- ◆ *Separation of Concerns.*
- ◆ Consiste em separar/isolar os diferentes aspectos de um sistema.
- ◆ Se todos os aspectos estão muito inter-relacionados, fica difícil tratar deles independentemente.
- ◆ Exemplos de aspectos:
 - segurança, confiabilidade, desempenho,...



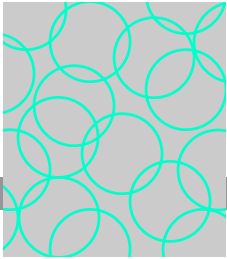
Modularidade

- ◆ Sistemas de grande porte precisam ser divididos em módulos; caso contrário, tornam-se intratáveis.
- ◆ Ajuda em duas situações:
 1. podemos lidar com os detalhes internos de cada módulo sem se preocupar com o mundo exterior;
 2. quando lidamos com os inter-relacionamentos entre os vários módulos de um sistema.



Modularidade

- ◆ A modularidade tenta atingir três metas:
 1. capacidade de decompor um sistema complexo ou de grande porte;
 2. capacidade de compor um sistema a partir de um conjunto de módulos;
 3. facilitar a compreensão de sistemas complexos.



Abstração

- ◆ Uma forma poderosa de se lidar com complexidade.
- ◆ Abstração consiste em um processo no qual
 - identificamos os aspectos essenciais de um fenômeno e
 - ignoramos os detalhes ou aspectos não relevantes.
- ◆ Foi o caminho percorrido pela pintura ao longo dos séculos XIX e XX.

Pittura figurativa



Nicolas Poussin

1594-1665

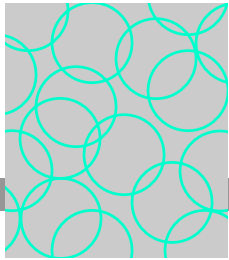
L'été ou Ruth et Booz

Vers 1660-1664

Ronda Noturna Rembrandt
1642



Arcângelo Ianelli - fase figurativa

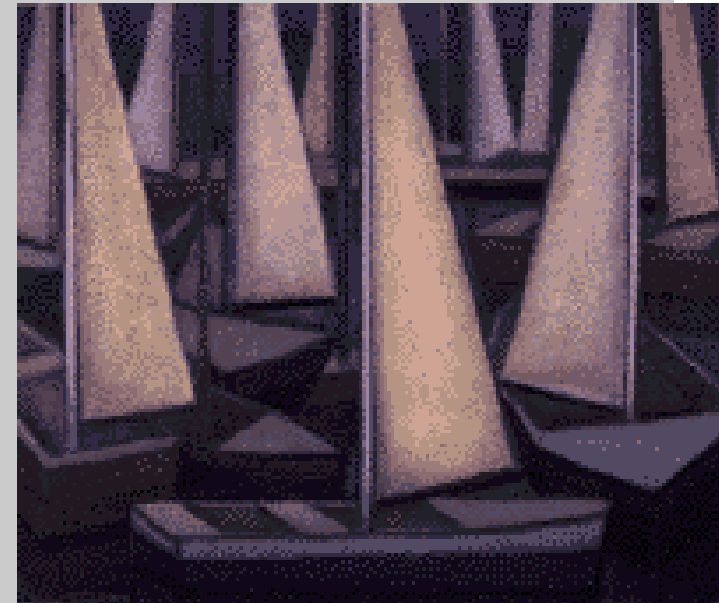


Katia 1956

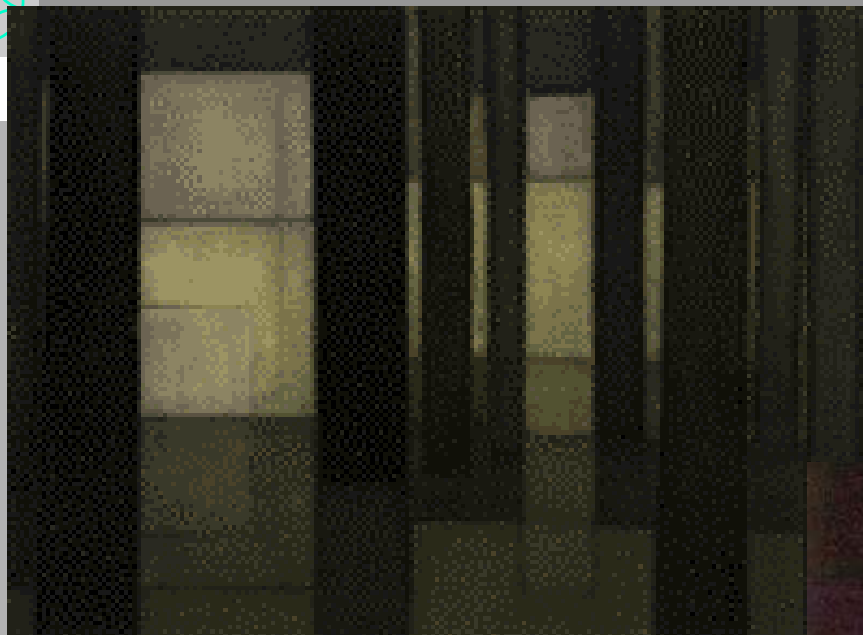
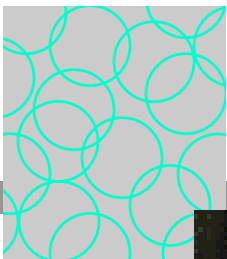
Marinha
1952



Barcos 1958



Arcângelo Ianelli – processo de abstração



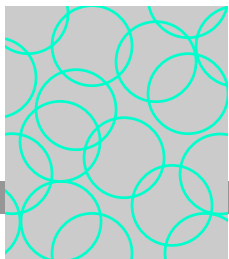
Árvores
1960



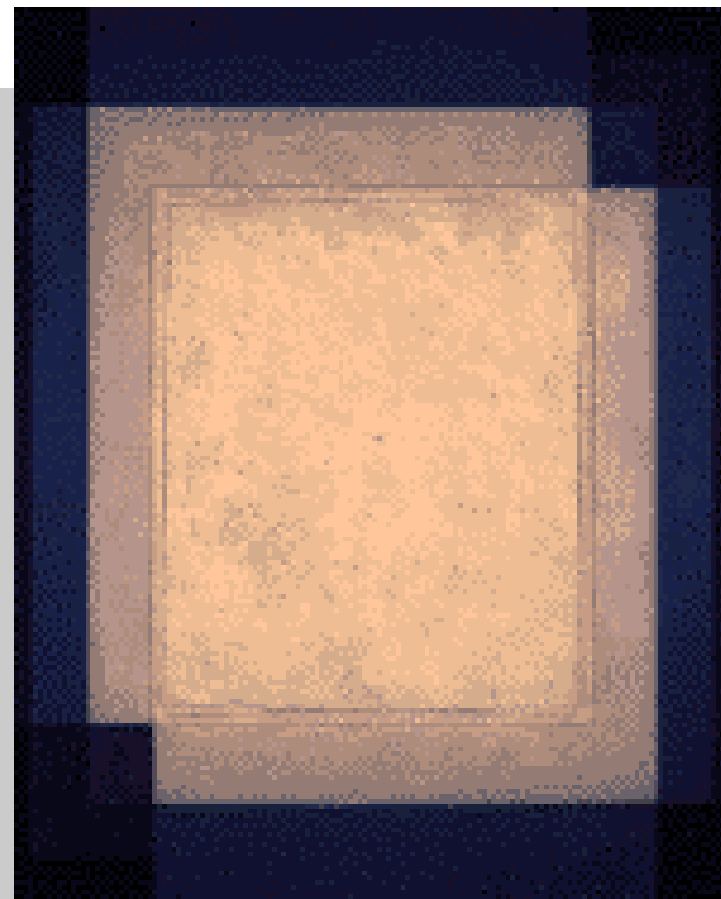
Natureza Morta
1960

24/09/02

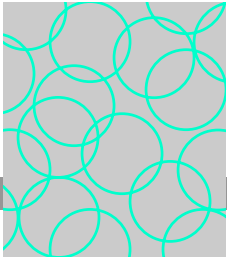
Arcângelo Ianelli – obras abstratas



Ruptura 1976



Superposição de Retângulos 1978



Abstração

- ◆ O bom projetista de software deve ser capaz de
 - efetuar abstrações mentais de elementos de um sistema complexo e
 - incluir abstrações bem elaboradas como parte do projeto arquitetural.
- ◆ Abstrações em SOs:
 - processo, memória virtual, RPC



Compartilhamento de Recursos

- ◆ Por um lado queremos um sistema modularizado
 - com baixo acoplamento entre os módulos e
 - com módulos bem coesos.
- ◆ Por outro lado, precisamos identificar quais recursos devem ser compartilhados entre os diversos módulos.
- ◆ Compartilhamento eficaz
 - evita duplicações
 - promove desacoplamento



Requisitos Não-Funcionais

- ◆ Também chamados de
 - atributos de qualidade
 - requisitos não comportamentais
- ◆ Descreve não *o que* um sistema faz,
- ◆ mas sim *como* ele faz.
- ◆ Padrão IEEE 830-1993 lista 13 requisitos não-funcionais para documentos de especificação de requisitos de software.



Requisitos Arquiteturais Não-Funcionais

◆ Usabilidade

- facilidade de aprender
- facilidade de usar

◆ Manutenibilidade

◆ Confiabilidade

◆ Desempenho

◆ Portabilidade

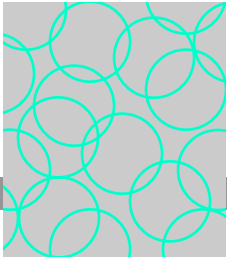
◆ Reusabilidade

◆ Segurança



Alguns Critérios para Medição de Usabilidade

- ◆ Tempo para realizar uma tarefa.
- ◆ Percentual de tarefa concluído / tempo.
- ◆ Taxa de sucessos / falhas.
- ◆ Tempo consumido com erros.
- ◆ Número de comandos utilizados.
- ◆ Número de comandos disponíveis não utilizados.
- ◆ Frequência de uso da documentação.



Manutenibilidade

- ◆ Facilidade de reparo de erros
- ◆ Facilidade de
 - evolução
 - alteração
- ◆ A manutenibilidade será boa se o sistema é
 - formado por módulos desacoplados e coesos
 - a arquitetura provê flexibilidade



Confiabilidade

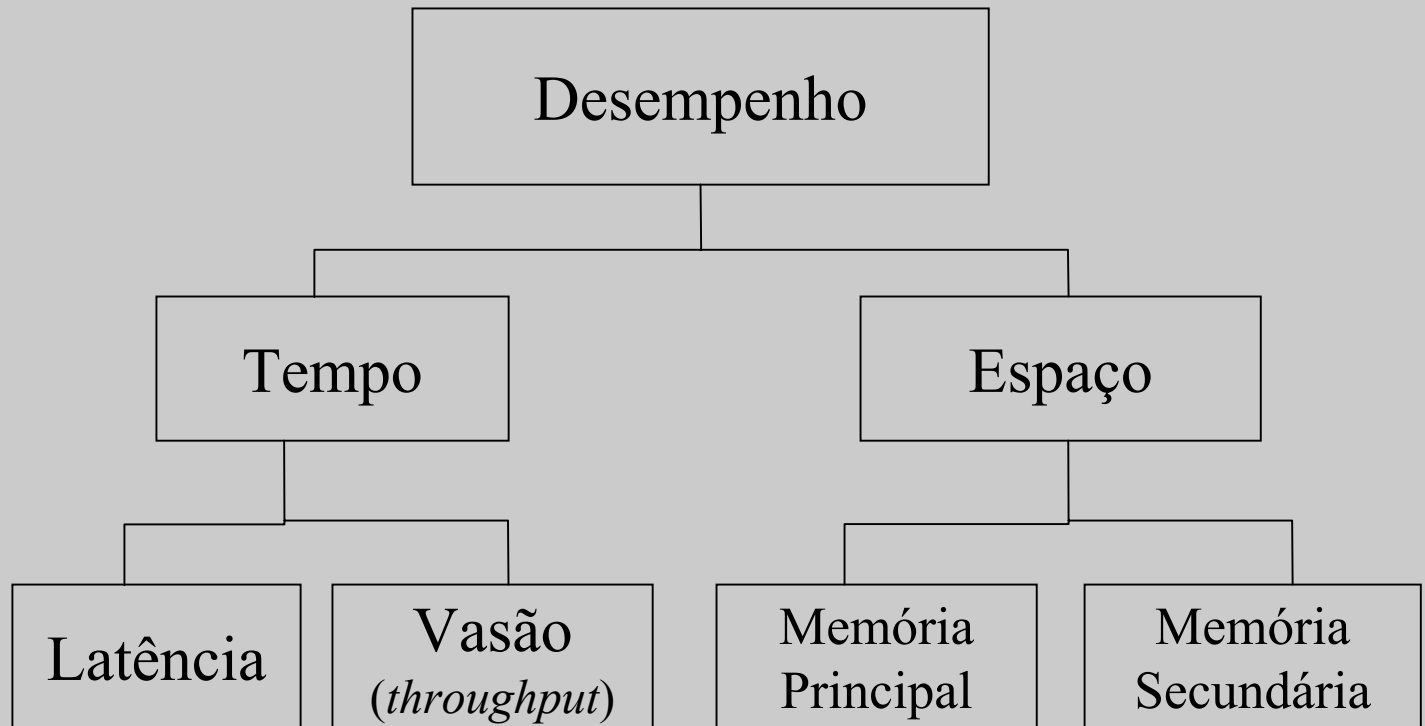
- ◆ Disponibilidade
 - p.ex. 99.999% do tempo ou das requisições
- ◆ Taxa de ocorrência de falhas
 - falha = ocorrência de comportamento inesperado, p.ex., $2+2=5$.
- ◆ Probabilidade de falha durante operação
- ◆ Tempo médio até ocorrência de falha
 - MTTF é usado com discos, por exemplo.



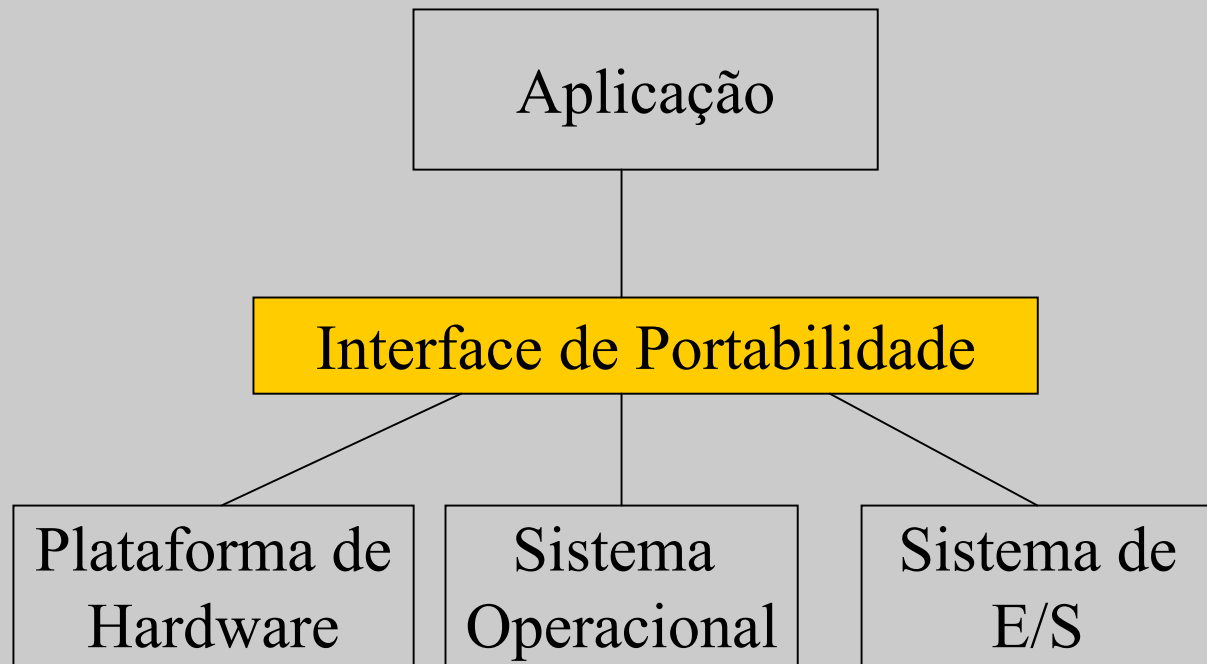
Confiabilidade

- ◆ *Mean Time Between Failures*
 - $MTBF = MTTF + MTTR$
 - onde MTTR é o tempo para reparo
- ◆ Disponibilidade = $MTTF / (MTTF + MTTR)$

Desempenho



Portabilidade

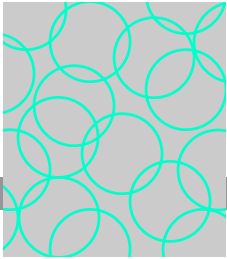




Reusabilidade

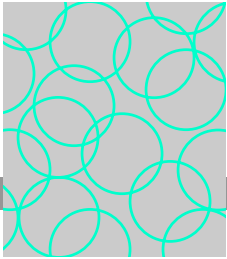
Reuso de:

- ◆ Aplicação (p.ex. em diferentes plataformas)
- ◆ Subsistemas
- ◆ Módulos
- ◆ Classes
- ◆ Funções



Segurança

- ◆ Identificação (de usuários e objetos)
- ◆ Autenticação
- ◆ Controle de Acesso
- ◆ Garantia de Integridade
- ◆ Garantia de Disponibilidade
- ◆ Auditoria de Segurança
- ◆ Detecção de Comportamento Suspeito



Bibliografia

- ◆ *Arquitetura de Software – Desenvolvimento orientado para arquitetura.* Antonio Mendes. Editora Campus, 2002.