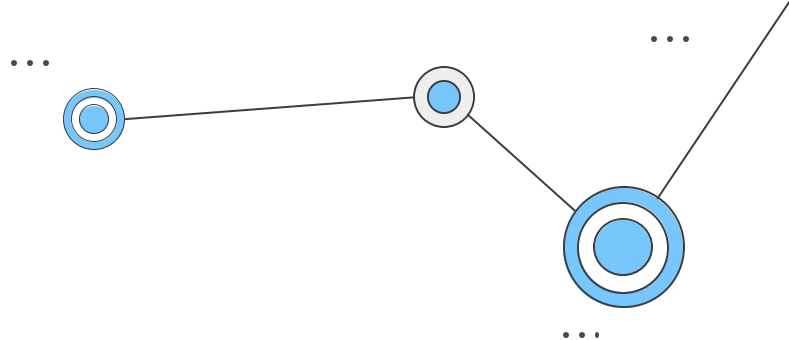




PUC Minas



# Projeto de Software

Prof. Dr. João Paulo Aramuni



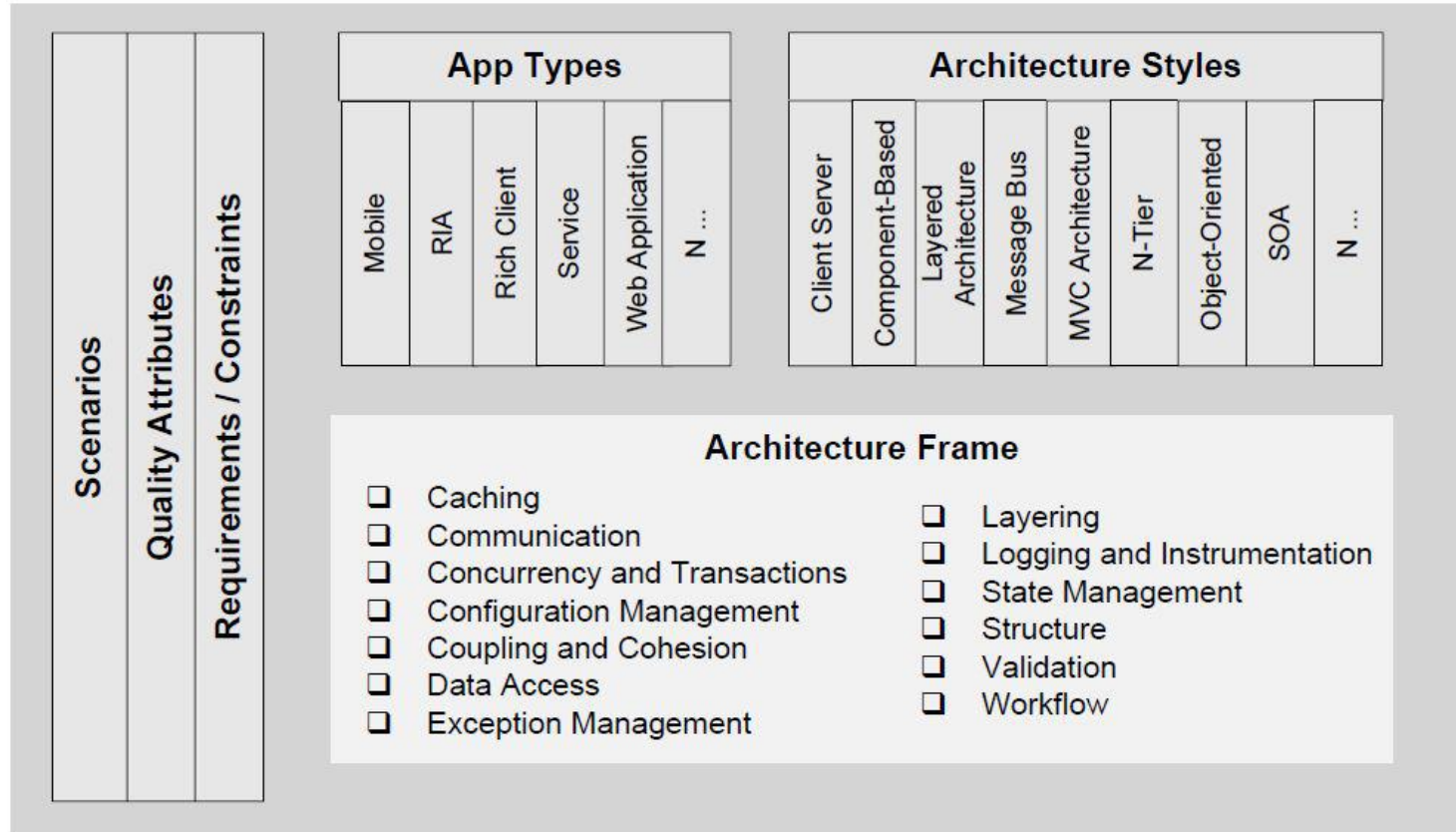
# Unidade 2

## Arquitetura de Software

PDS - Manhã /Noite



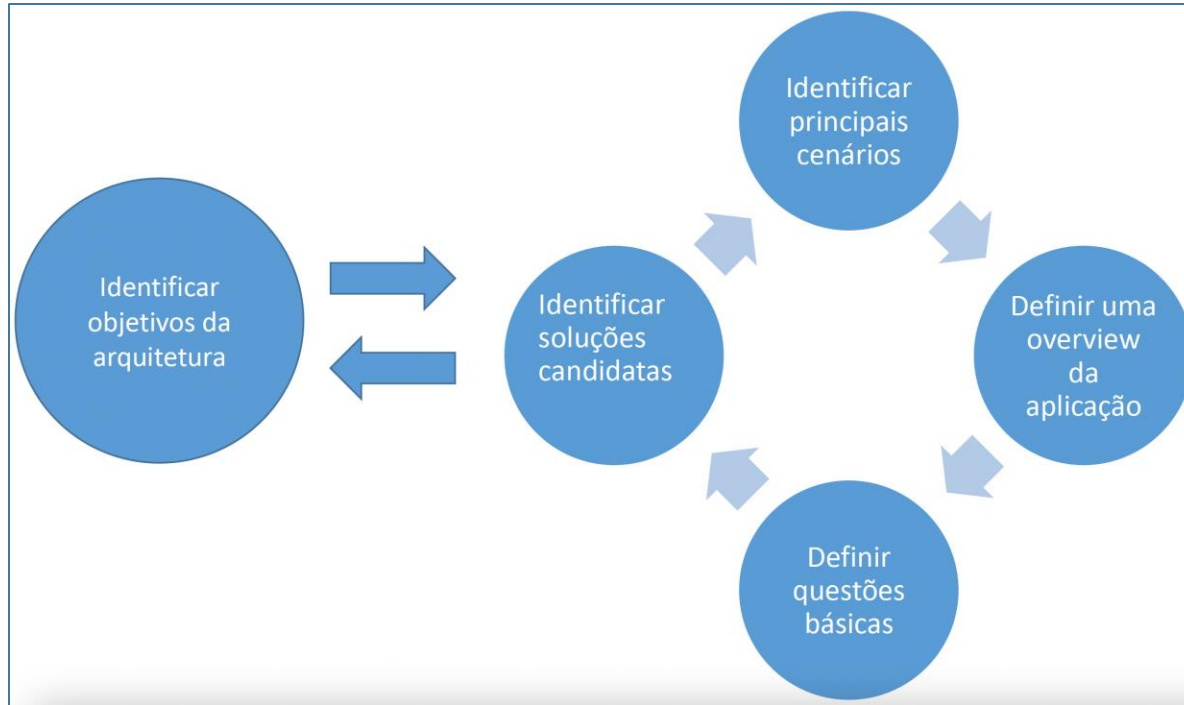
# Arquitetura de Software: meta frame



# Arquitetura de Software – Definição

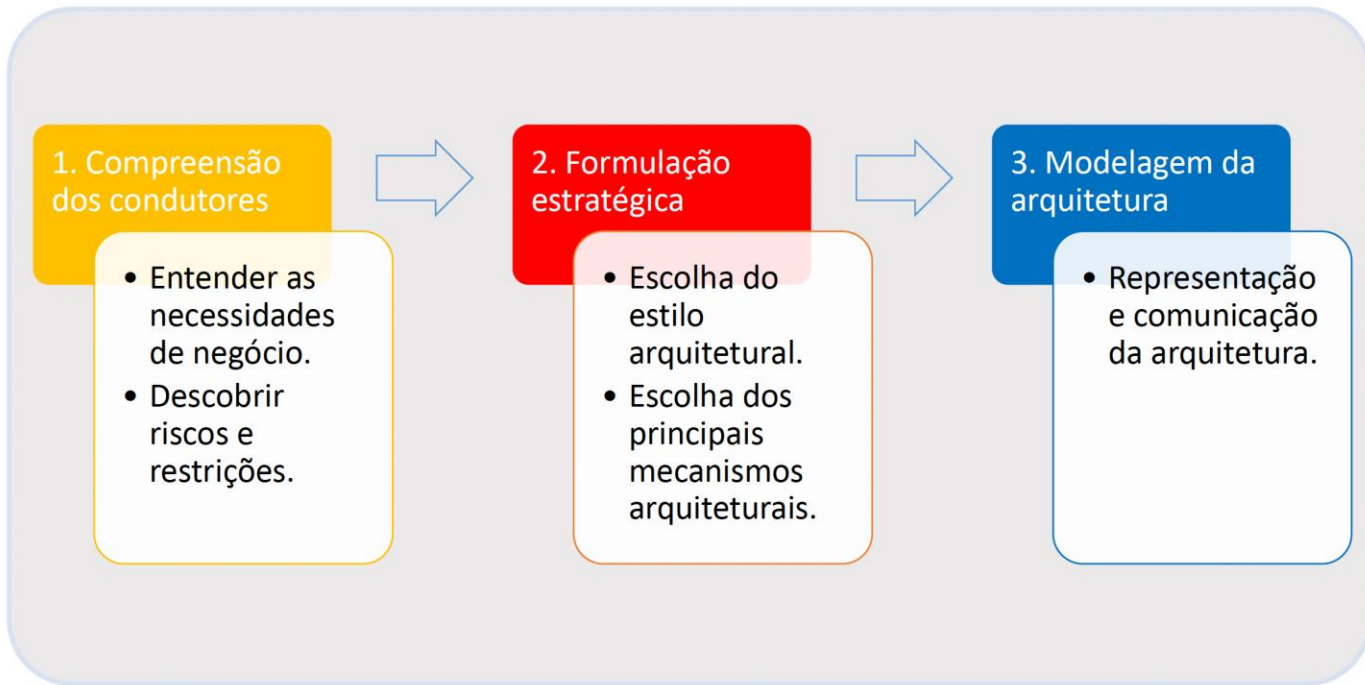
Arquitetura de software: passos básicos

- Ciclo básico para definição da arquitetura



# Arquitetura de Software – Definição

Método didático para definir uma arquitetura

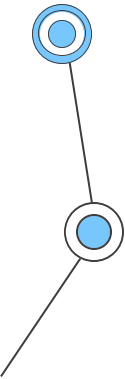
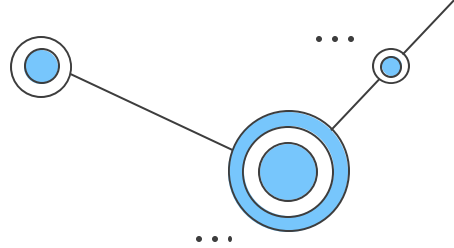


# Arquitetura de Software – Definição

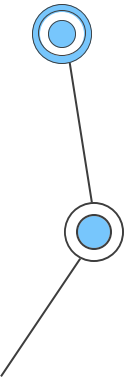
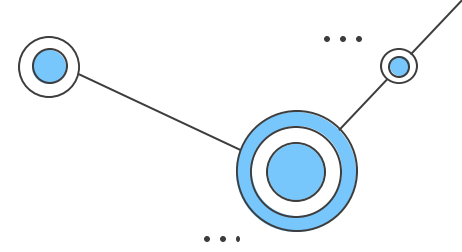
O que é entender as necessidades de negócio?  
[1. Compreensão dos condutores]

Vamos ver uma analogia com o processo de compra de um novo automóvel.

- Geralmente escolhe-se o carro com base em questões emocionais.
- Nossas escolhas na arquitetura não podem ser dessa maneira.



# Abordagem adequada para comprar um carro

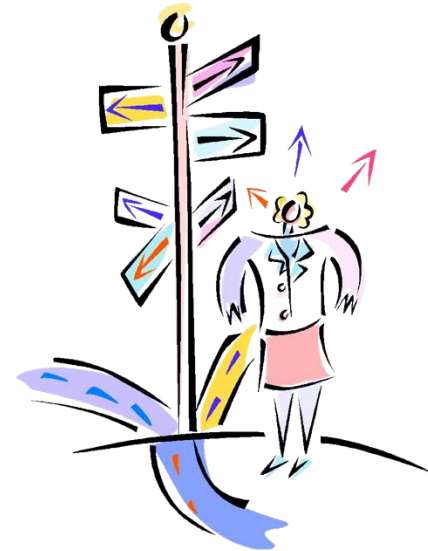
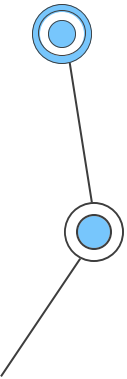
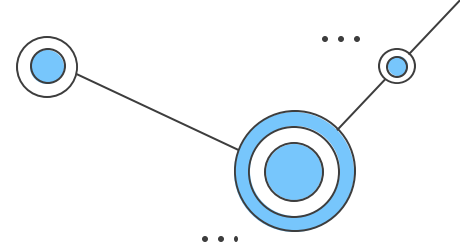


# Arquitetura de Software – Definição

Decisão estratégica: escolha de um carro

Antes de escolher um carro, deve-se compreender a real necessidade.

É importante definir quais critérios irão conduzir a escolha.

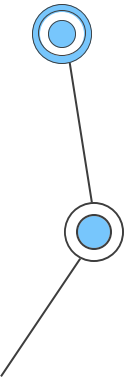
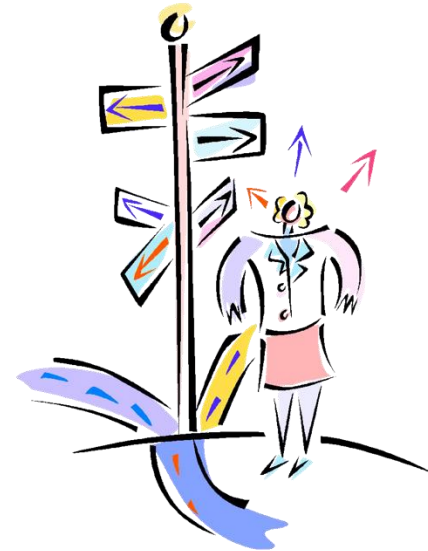
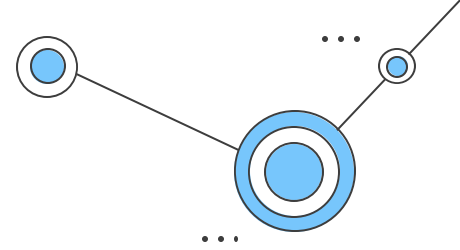




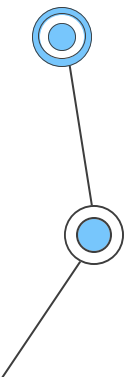
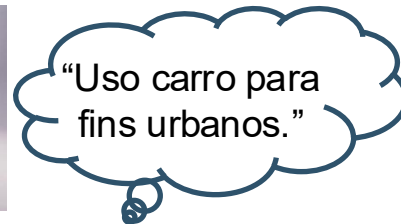
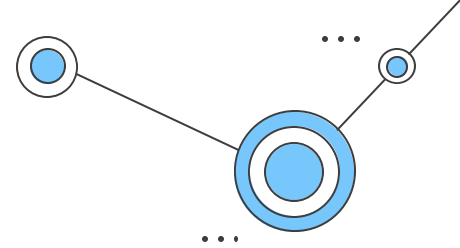
# Arquitetura de Software – Definição

Necessidades:

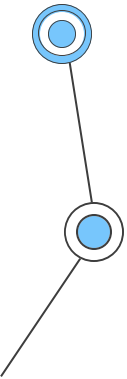
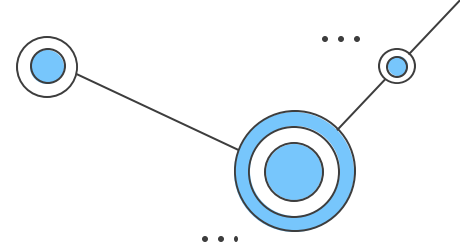
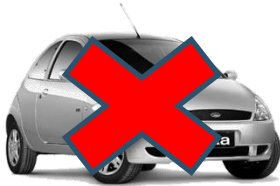
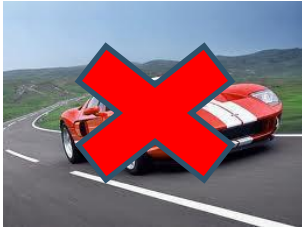
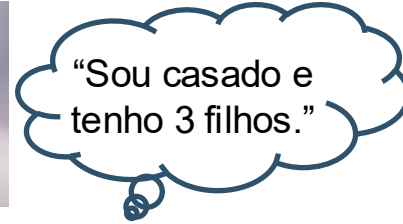
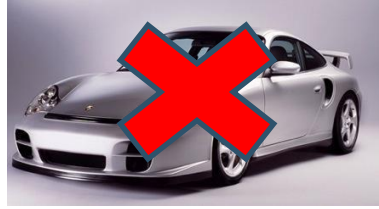
- “Uso carro para fins urbanos.”
- “Sou casado e tenho 3 filhos.”
- “Faço pequenas viagens e minha esposa carrega muitas bagagens.”
- “Meu orçamento é limitado a 65 mil Reais”
- “Moro em uma cidade quente.”



# Abordagem adequada para comprar um carro



# Abordagem adequada para comprar um carro



# Abordagem adequada para comprar um carro

“Faço pequenas viagens  
e minha esposa carrega  
muitas bagagens.”



# Abordagem adequada para comprar um carro

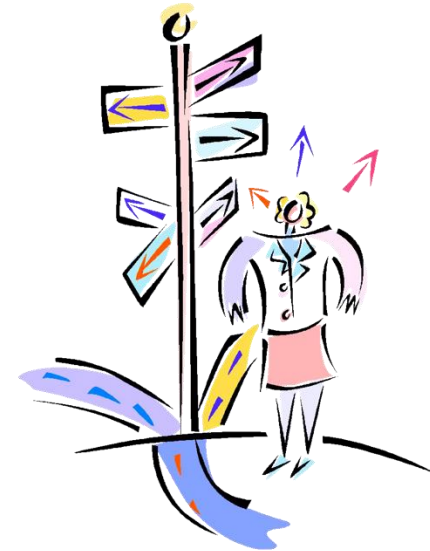
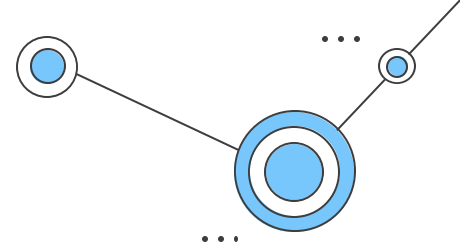
“Meu orçamento é limitado a 65 mil reais.”



# Arquitetura de Software – Definição

Decisões baseadas em escolhas racionais:

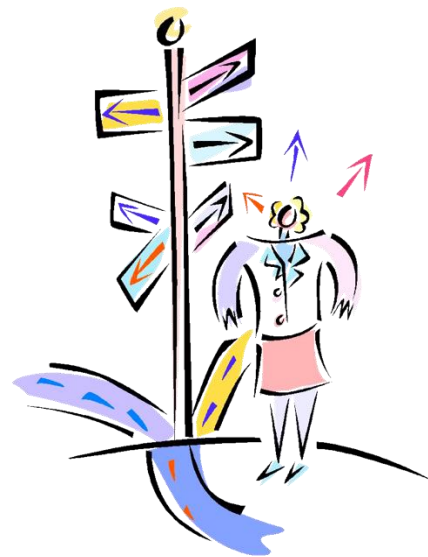
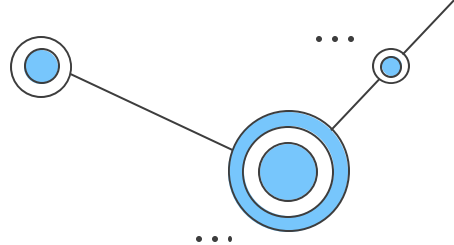
- Podemos não ter gostado da escolha realizada, mas devemos admitir que foi uma escolha racional baseada em decisões lógicas.



# Arquitetura de Software – Definição

Decisões baseadas em escolhas racionais:

- “Moro em uma cidade quente.”
  - Carro com ar-condicionado!
- Apesar de não ter definido a escolha do carro, é também uma decisão relevante a ser considerada.





# Arquitetura de Software – Definição

Sem uma estratégia baseada em decisões racionais, podemos acabar com uma arquitetura inadequada...



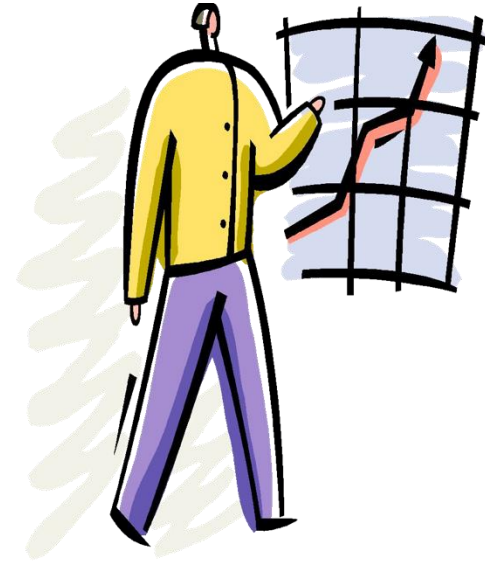
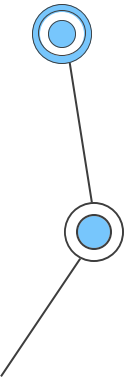
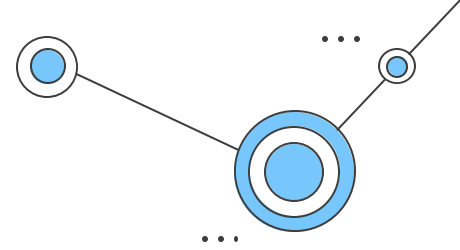


# Arquitetura de Software – Definição

Condutores arquiteturais: necessidades do negócio  
[1. Compreensão dos condutores]

Necessidades de negócio

- Quais são as necessidades de negócio endereçadas pela arquitetura?
- As mais importantes são condutores arquiteturais.



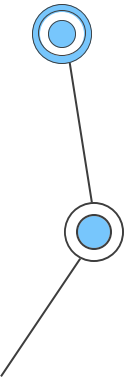
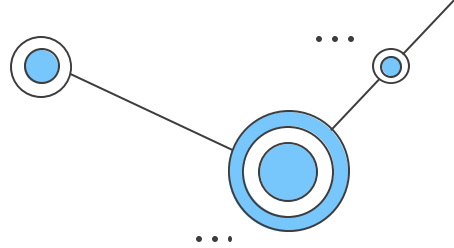
# Arquitetura de Software – Definição

Condutores arquiteturais: identificar riscos  
[1. Compreensão dos condutores]

Riscos técnicos

Quais riscos podem causar impacto na arquitetura?

Os riscos mais impactantes são condutores arquiteturais.



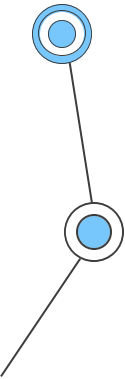
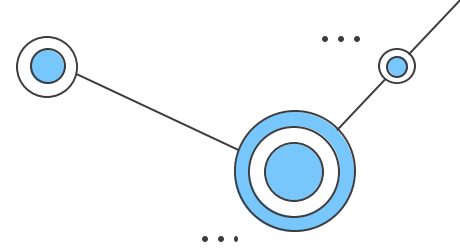
# Arquitetura de Software – Definição

Condutores arquiteturais: identificar restrições  
[1. Compreensão dos condutores]

Restrições

Existem restrições (tecnológicas, de projeto ou corporativas) importantes para a arquitetura?

Restrições mais impactantes são condutores arquiteturais.



# Arquitetura de Software – Definição

Condutores arquiteturais: requisitos de qualidade  
[1. Compreensão dos condutores]



Desempenho	Escalabilidade	Usabilidade
Testabilidade	Reusabilidade	Confiabilidade
Disponibilidade	Segurança	Facilidade de estender
Facilidade de manter	Facilidade de gerenciar	Etc...

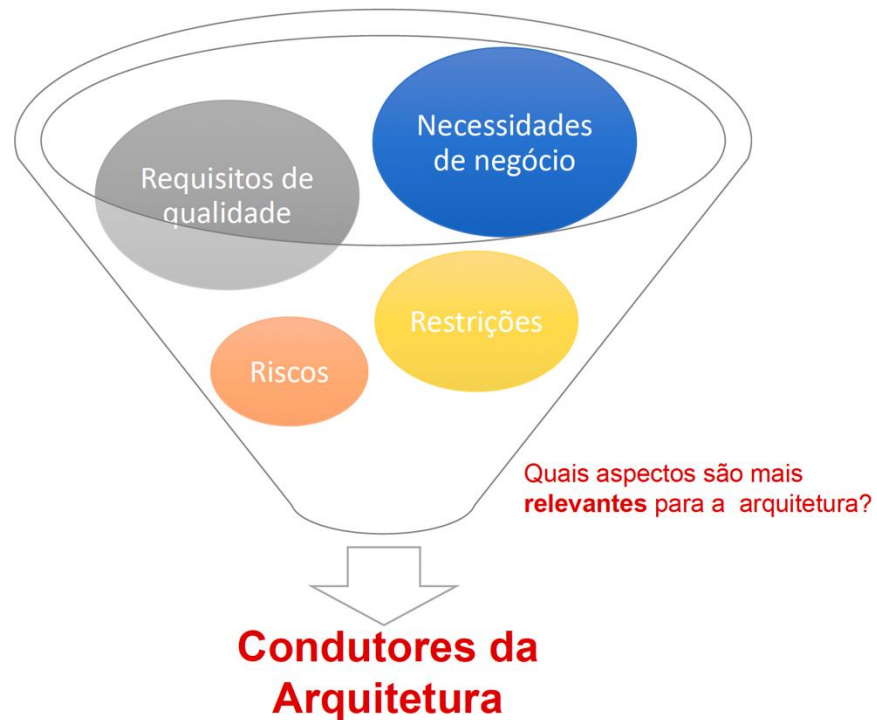


Exemplos de requisitos de qualidade impactados pela distribuição de um sistema.

Requisitos não funcionais

# Arquitetura de Software – Definição

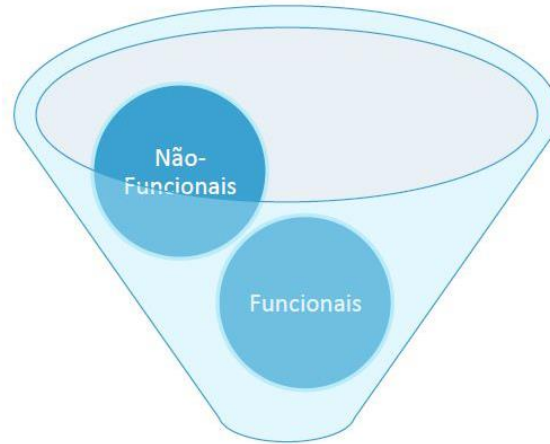
Funil dos condutores arquiteturais  
[1. Compreensão dos condutores]



# Arquitetura de Software – Definição

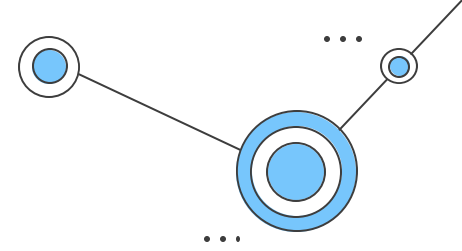
Requisitos arquiteturais

[1. Compreensão dos condutores]

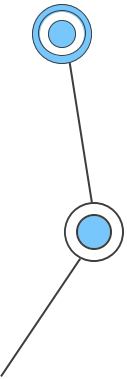


*O funil da relevância!*

Requisitos  
Arquiteturais



Um requisito pode ser chamado de Requisito Arquitetural sempre que for significativo para o Desenvolvimento de uma arquitetura.

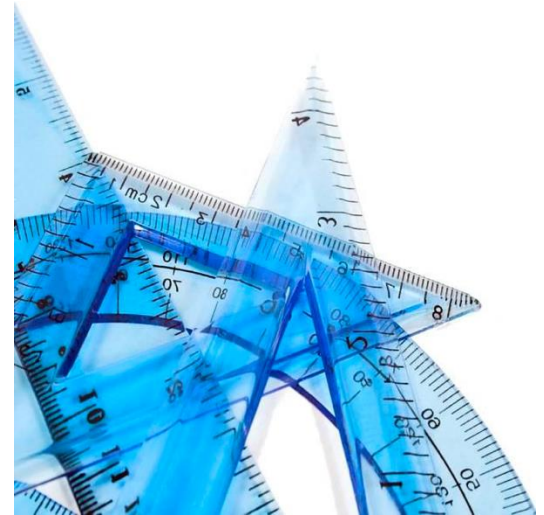
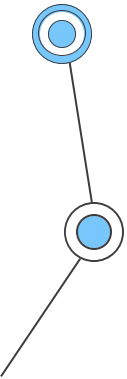
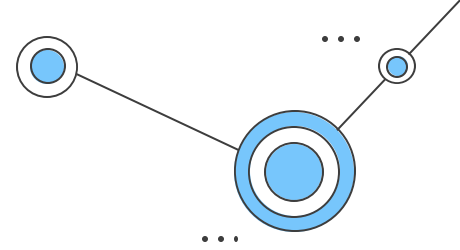


# Arquitetura de Software – Definição

Requisitos

[1. Compreensão dos condutores]

Um Requisito é uma condição ou capacidade (funcional ou não funcional) com a qual um Sistema deve estar conforme.



# Requisitos: Casos revisitados

## Caso 1

- Uma equipe de TI no governo descobriu na semana da entrega que o envio de email não funcionava devido a **topologia física desconhecida e restrições de segurança (Certificados digitais e HTTPS)**

## Caso 2

- Uma outra equipe teve que refazer toda a camada visual de Javascript ao perceber em produção que **o sistema não atendia o Firefox**

## Caso 3

- Uma terceira equipe de uma grande empresa de minas descobriu em produção que o tempo médio dos **casos de uso era de 40 segundos**

## Caso 4

- Uma equipe descobriu que a aplicação **tratava o banco de dados em produção periodicamente (deadlocks) e parava toda o lançamento de notas de uma universidade.**



# Qualidade dos requisitos [1. Compreensão dos condutores]

Qual o problema com os requisitos abaixo?

"O sistema deve ser rápido e capaz de processar grandes quantidades de requisições simultâneas."

"O sistema deve ter uma interface amigável."

"O sistema deve ter um baixo número de defeitos."

"O sistema deve ser altamente parametrizável."

"O sistema deve ser seguro."

"O sistema deve oferecer suporte a procedimentos como pagamento de contas, emissão de extrato, transferência, etc."

# Arquitetura de Software – Definição

Requisitos tolos!

[1. Compreensão dos condutores]

Utilização de termos genéricos:

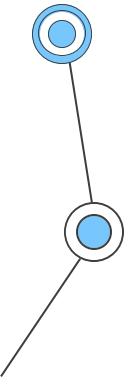
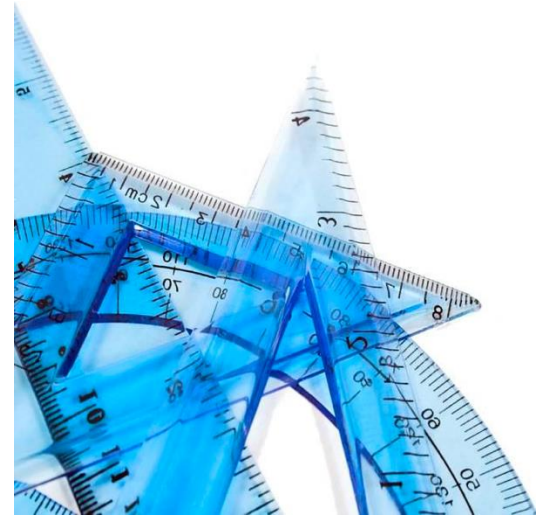
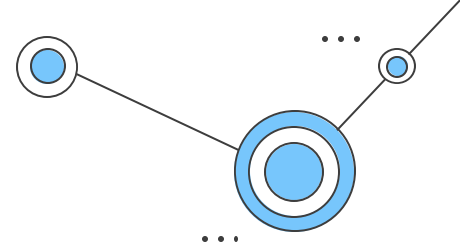
- Geralmente, usualmente, frequentemente, normalmente, tipicamente, aproximadamente.

Termos relativos:

- Rápido, flexível, intuitivo, amigável, grande número de requisições, baixo tempo de resposta, etc.

Termos furtivos:

- Poderia, deveria, talvez, pode, provavelmente, etc.



# Arquitetura de Software – Definição

Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

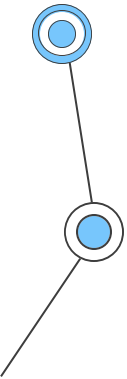
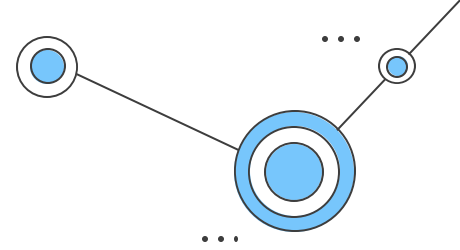
Princípio SMART

- **S**pecific (Específicos), **M**ensurable (Mensurável), **A**ttainable (Atingível) **R**ealizable (Realizável), **T**raceable (Rastreável).

O princípio SMART permite que um requisito seja escrito com menor ambiguidade.

Exemplo, considere o requisito “A interface do sistema deve ser amigável”.

- O uso de palavras genéricas como sistema e advérbios como amigável não transmite certeza sobre como devemos resolver este problema e implementar o código.



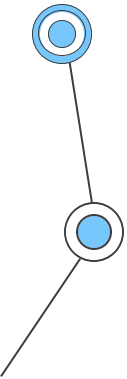
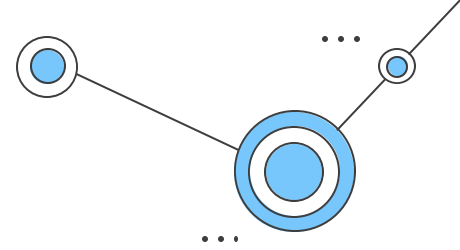
# Arquitetura de Software – Definição

Requisitos “Espertos” – Princípios SMART  
[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

## 1- Specific (Específicos).

- Um requisito deve ser específico. Exemplos de requisitos específicos indicam o caso de uso, tela ou módulo a que eles se referem.
- Como exemplo do requisito fornecido anteriormente poderíamos dizer que a interface da tela de cadastro de estudantes deve ser amigável. O requisito foi tornado específico.



# Arquitetura de Software – Definição

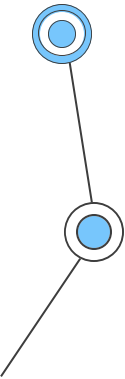
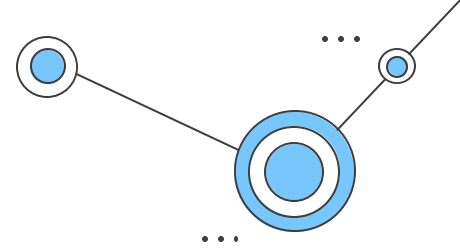
Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

2- Mensurable (Mensurável).

- Todo arquitetural deve ser mensurável.
  - ✓ é possível verificar objetivamente que o requisito foi atendido.
- Exemplo: Um usuário novato deve conseguir operar a tela de interface de cadastro de aluno com um treinamento não superior a 30 minutos e com no máximo um erro ou advertência fornecido pelo sistema.



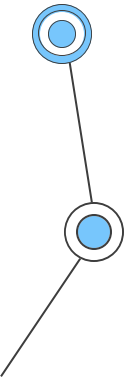
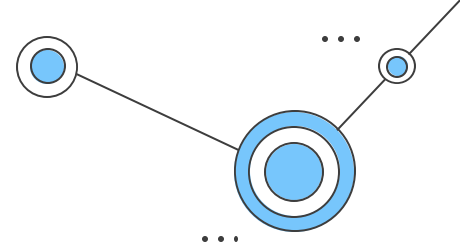
# Arquitetura de Software – Definição

Requisitos “Espertos” – Princípios SMART  
[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

## 3 - Attainable (Atingível)

- o requisito é tecnicamente viável.
- Pode parecer que todo requisito é virtualmente atingível mas existem contraexemplos. Se escrevêssemos que a tela de cadastro de alunos precisa ter disponibilidade de 100%, estaríamos prometendo algo que não pode ser cumprido.
- Todo programa computacional opera sobre máquinas, que apresentam tempos médios de falhas em suas especificações.



# Arquitetura de Software – Definição

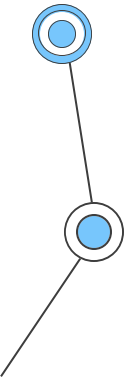
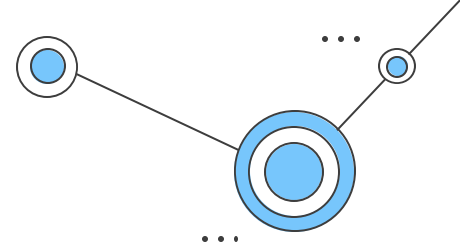
Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

## 4 - Realizable (Realizável)

- Muitos requisitos são atingíveis, mas nem todos o são no contexto de um projeto. Ele é realista, tendo em conta os recursos e o prazo do projeto.
- Projetos tem times limitados, tempo limitado e recursos físicos e financeiros limitados.



# Arquitetura de Software – Definição

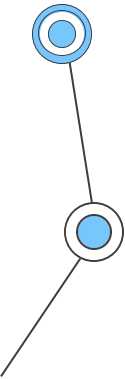
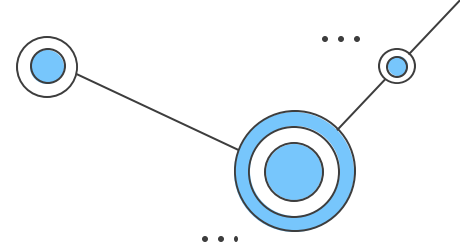
Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

## 4 - Realizable (Realizável)

- Se, por exemplo, buscássemos que o a tela de cadastro de alunos operasse em um ambiente tolerante a falhas mas não tivéssemos orçamento para buscar máquinas de redundância, este requisito não seria realizável.
- Se um requisito não é realizável, ele deve ser relaxado ou modificado.





# Arquitetura de Software – Definição

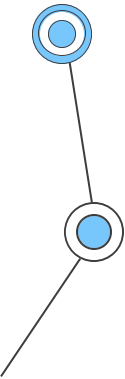
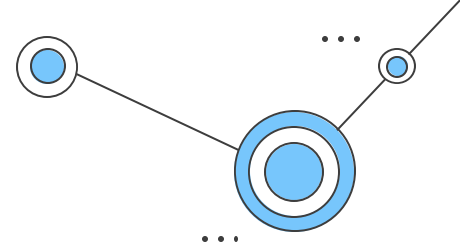
Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

Requisitos SMART são aqueles que atendem a cinco critérios:

## 5 - Traceable (Rastreável).

- O Standish Group fez um levantamento sobre requisitos e verificou que quase 50% dos requisitos de softwares em produção nunca são usados ou são usados raramente.
- Um dos motivos para isso é a introdução de requisitos nos software as vezes ocorre de forma indisciplinada e as vezes pelos próprios desenvolvedores, sem autorização dos usuários.
- Desta forma, é importante saber a origem de cada requisito. Todo requisito, portanto, deve ser rastreável.



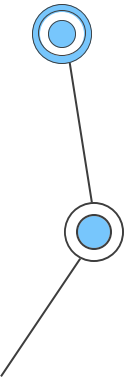
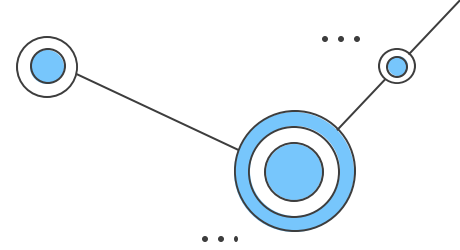
# Arquitetura de Software – Definição

Requisitos “Espertos” – Princípios SMART

[1. Compreensão dos condutores]

Exemplo

- **Requisito ambíguo:** “O sistema deve ser rápido e capaz de processar grandes quantidades de requisições simultâneas.”
- **Requisito SMART:** “A tela de cadastro de usuários deve possuir um tempo de resposta menor que 8 segundos e suportar 20 usuários simultâneos em horários de pico (15:00 às 19:00). [Requisito elicitado e aprovado com usuário Paulo Silva]”

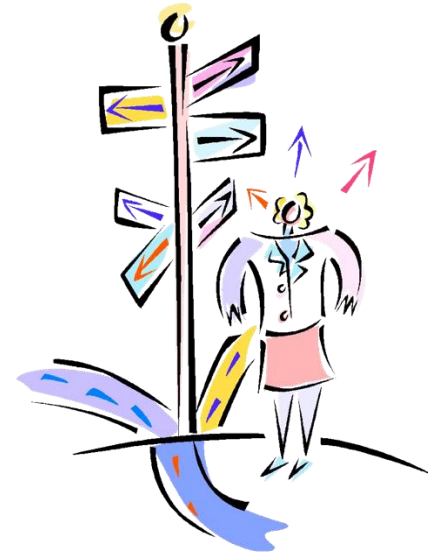
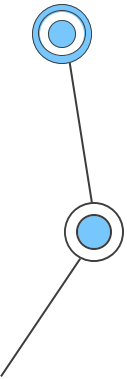
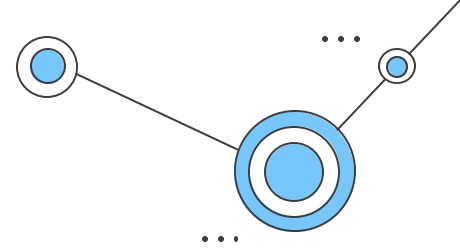


# Arquitetura de Software – Definição

Mecanismos Arquiteturais  
[2. Formulação estratégica]

Escolha de Tecnologias

- Como escolher tecnologias de uma arquitetura?



# Arquitetura de Software – Definição

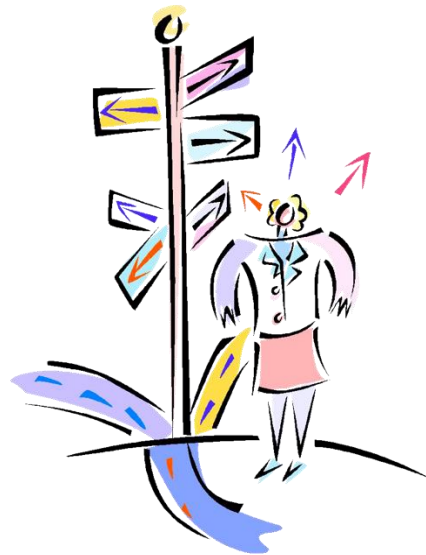
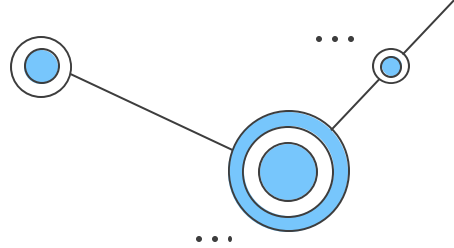
## Mecanismos Arquiteturais [2. Formulação estratégica]

Mecanismo arquitetural representa uma solução comum para um problema recorrente.

O valor na definição de mecanismos de arquitetura é que eles:

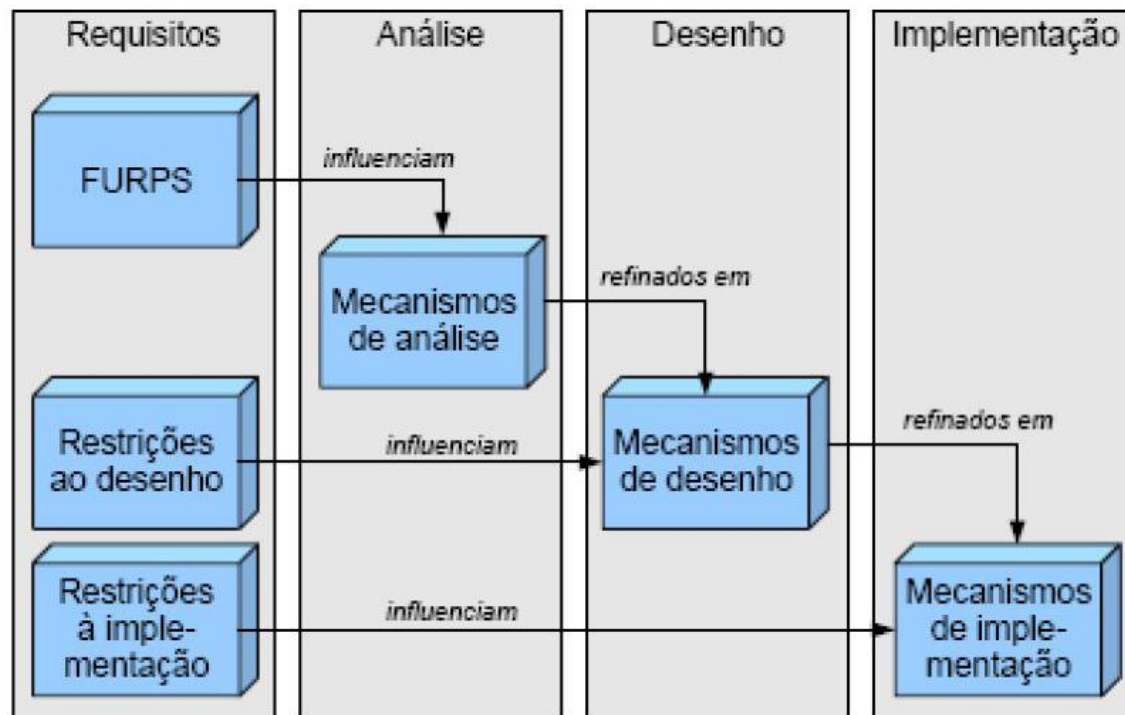
- Mostram explicitamente aspectos dos mecanismos da solução que são comuns em todo o sistema.
- Isto ajuda a planejar.
- Coloca marcadores para que os desenvolvedores construam os aspectos do sistema uma vez e reusem-nos. Isto reduz a carga de trabalho.

Um mecanismo arquitetural pode ter três estados: **análise**, **design** e **implementação**.



# Arquitetura de Software – Definição

Mecanismos Arquiteturais [2. Formulação estratégica]

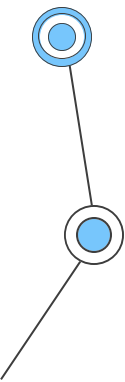
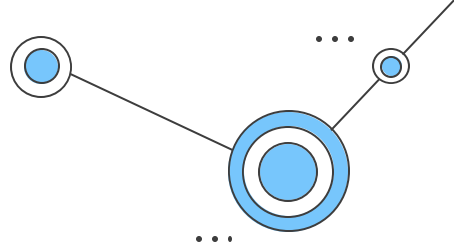


# Arquitetura de Software – Definição

Mecanismos Arquiteturais [2. Formulação estratégica]

Um exemplo complexo - segurança

- Requisitos arquiteturais:
  - Usuários devem ser corretamente identificados.
- Restrições:
  - A solução de segurança deve ser baseada em padrão aberto.
  - As senhas não devem trafegar sobre a rede.
  - A empresa já utiliza soluções Microsoft.



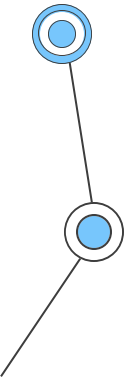
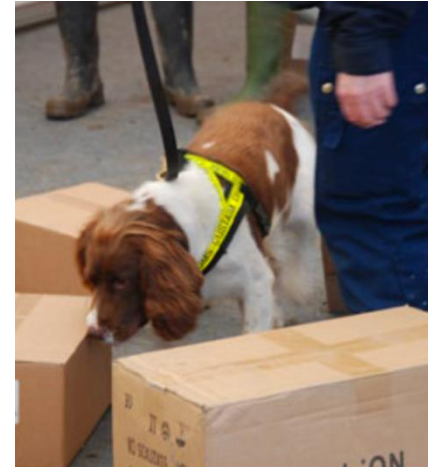
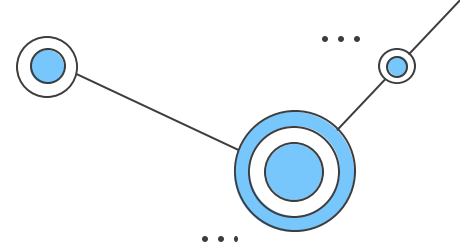
# Arquitetura de Software – Definição

Mecanismos Arquiteturais [2. Formulação estratégica]

Um exemplo complexo - A segurança do Hotel ACME

Mecanismo de análise:

- Os projetistas estudam materiais técnicos sobre autenticação.
- A autenticação é solução técnica para credenciar usuários de um sistema.



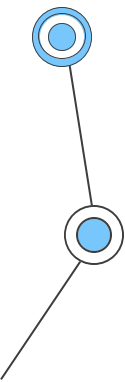
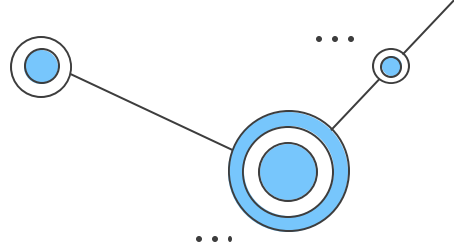
# Arquitetura de Software – Definição

Mecanismos Arquiteturais [2. Formulação estratégica]

Um exemplo complexo - A segurança do Hotel ACME

Mecanismo de desenho:

- As soluções Kerberos e LDAP são elencadas, pois são padrões abertos.
- O Kerberos é escolhido, pois não trafega senhas sobre a rede.





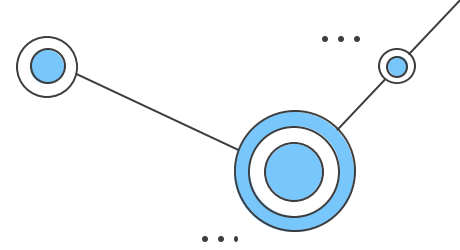
# Arquitetura de Software – Definição

Mecanismos Arquiteturais [2. Formulação estratégica]

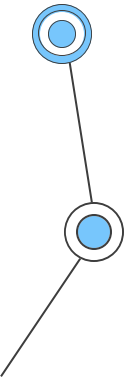
Um exemplo complexo - A segurança do Hotel ACME

Mecanismo de implementação:

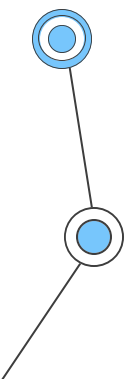
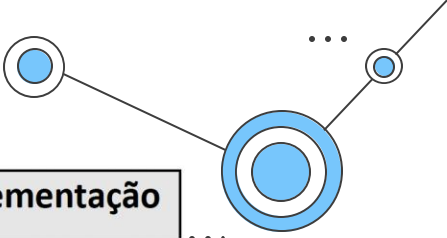
- Dado que a empresa já possui produtos Microsoft, escolhemos um produto Microsoft que suporta o protocolo Kerberos.
- Active Directory com Kerberos.



*Kerberos, na mitologia grega, é o cão de três cabeças que guarda a porta do Hades (inferno).*



# Mecanismos Arquiteturais [2. Formulação estratégica]



Mecanismo de Análise	Mecanismo de Design	Mecanismo de Implementação
Front-End	Interface com Usuário	Bootstrap, Javascript, HTML5 , CSS
ESB	Integração entre serviços	MuleESB
Message Broker	Troca de mensagens	RabbitMQ
Comunicação entre processos	Framework SOA	WCF
Comunicação entre processos	Framework SOA	WebAPI
Persistência	Banco de dados relacional	Microsoft SQL Server
Persistência	Framework ORM	Entity Framework
Versionamento	Versionamento dos fontes da aplicação.	Visual Studio Team Foundation Services
Alta Disponibilidade	Load Balance de Serviços	Network Load Balancing
Alta Disponibilidade	Load Balance de ESB	MuleESB Cluster
Alta Disponibilidade	Load Balance de ESB	Microsoft SQL Server AlwaysOn

# Arquitetura de Software – Definição

Processo de modelagem resumido

Requisitos  
arquiteturais  
são coletados  
com os métodos

- SMART
- FURPS+



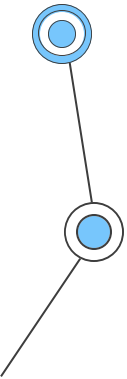
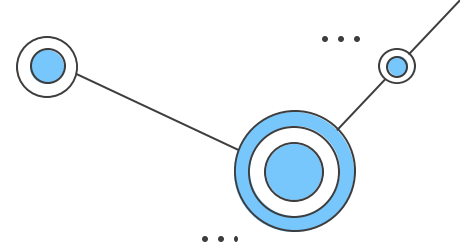
Mecanismos de  
análise que  
solucionam  
cada requisito  
arquitetural são  
escolhidos.



Para cada  
mecanismo de  
análise,  
mecanismos de  
desenho e  
implementação  
são estudados e  
escolhidos.



Modelos  
ajudam a  
visualizar os  
mecanismos  
trabalhados.

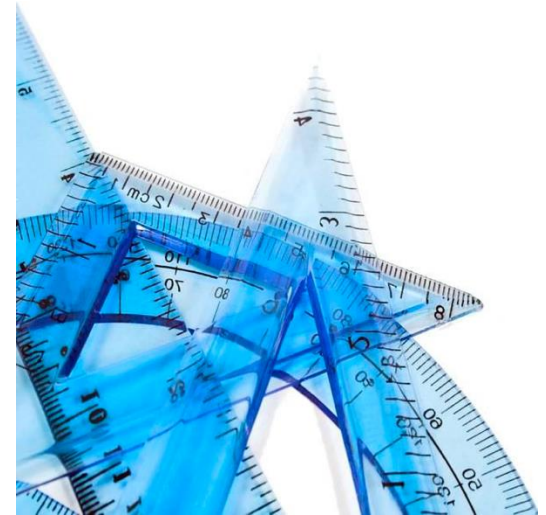
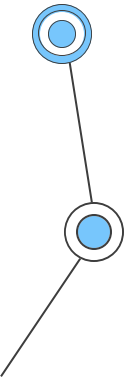
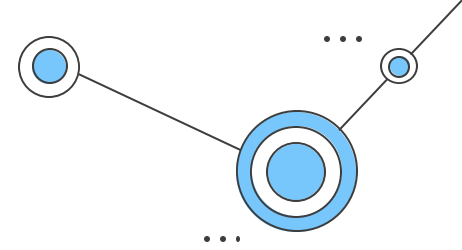


# Arquitetura de Software – Definição

Modelagem da arquitetura  
[3. Modelagem da arquitetura]

Representar a arquitetura:

- Modelos, diagramas, tabelas, descrições textuais;
- O objetivo é comunicar, entender e “validar” a arquitetura;



# Arquitetura de Software – Definição

Modelagem da arquitetura  
[3. Modelagem da arquitetura]

vocabulário  
funcionalidade

## Visão projeto

(lógica)

diagramas de estado  
diagrama de classes  
diagrama de  
interação

gerenciamento da configuração,  
montagem do sistema

## Visão implementação

diagramas de  
componentes

**Visão de caso  
de uso**  
diagramas caso  
de uso e sequência

## Visão processo

diagramas  
processos

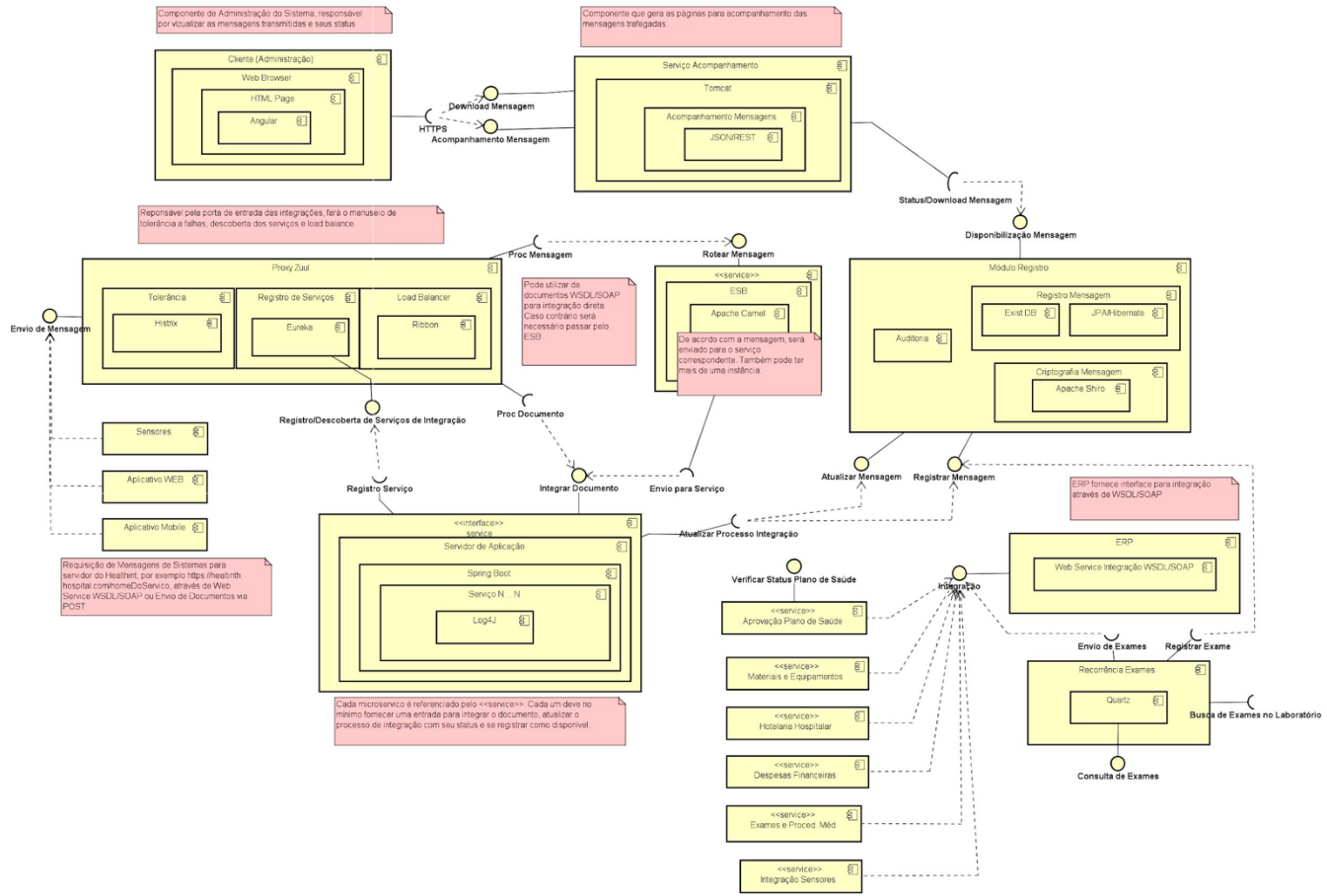
## Visão implantação

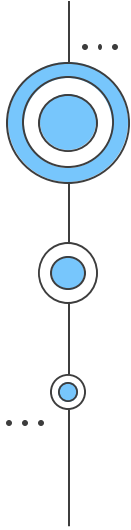
diagramas de  
implantação

Desempenho, escalabilidade, throughput

topologia do sistema, distribuição,  
Fornecimento, instalação

# Modelagem da arquitetura

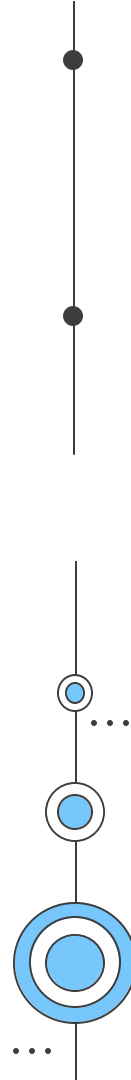




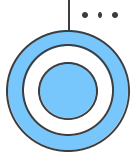
# Projeto de Software

## Referências básicas:

- **ACM TRANSACTIONS ON SOFTWARE ENGINEERING AND METHODOLOGY**. New York, N.Y., USA: Association for Computing Machinery, 1992-. Trimestral. ISSN 1049-331X. Disponível em: <https://dl.acm.org/toc/tosem/1992/1/2>. Acesso em: 19 jul. 2024. (Periódico On-line).
- LARMAN, Craig. **Utilizando UML e padrões**: uma introdução á análise e ao projeto orientados a objetos e desenvolvimento iterativo. 3. ed. Porto Alegre: Bookman, 2007. E-book. ISBN 9788577800476. (Livro Eletrônico).
- SILVEIRA, Paulo et al. **Introdução à arquitetura e design de software**: uma visão sobre a plataforma Java. Rio de Janeiro, RJ: Elsevier, Campus, 2012. xvi, 257 p. ISBN 9788535250299. (Disponível no Acervo).
- VERNON, Vaughn. **Implementando o Domain-Driven Design**. Rio de Janeiro, RJ: Alta Books, 2016. 628 p. ISBN 9788576089520. (Disponível no Acervo).

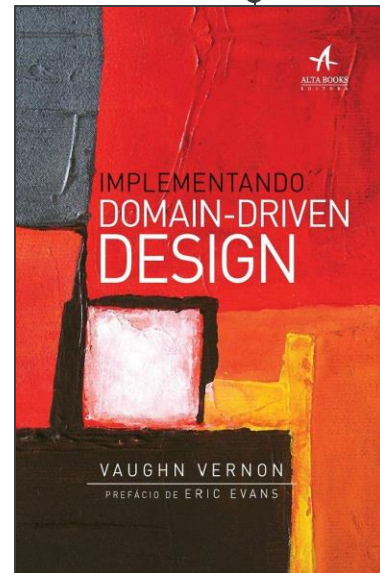
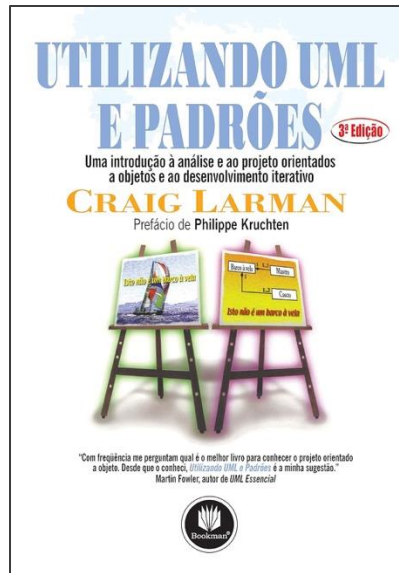
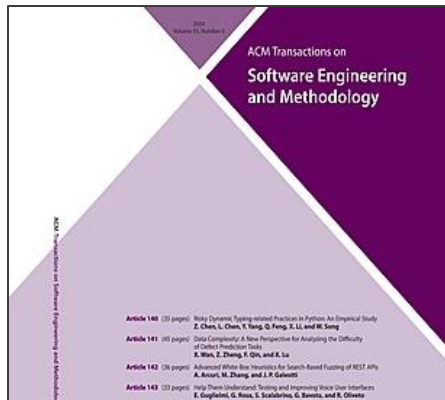






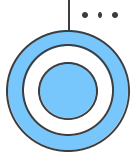
# Projeto de Software

## Referências básicas:



...

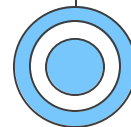


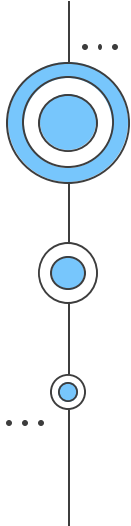


# Projeto de Software

## Referências complementares:

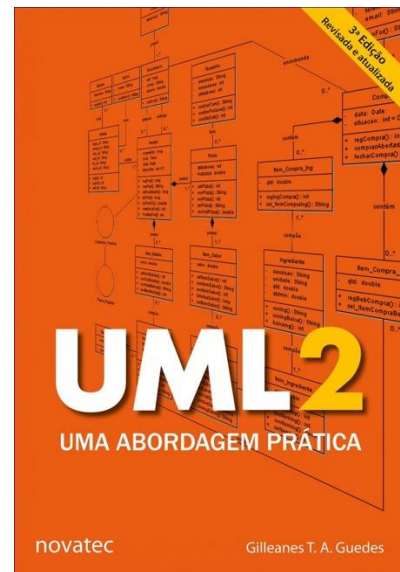
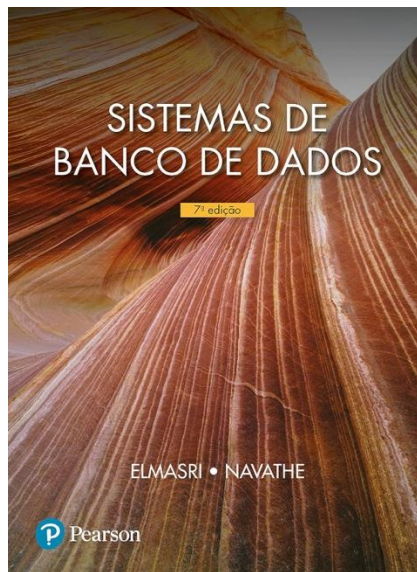
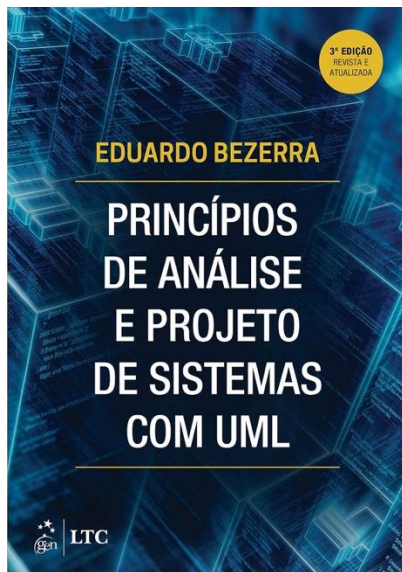
- BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas com UML**. 3. ed. rev. e atual. Rio de Janeiro: Elsevier, 2015. xvii, 398 p. ISBN 9788535226263. (Disponível no Acervo).
- ELMASRI, Ramez; Navathe, Shamkant B. **Sistemas de banco de dados**, 7ª ed. Editora Pearson 1152 ISBN 9788543025001. (Livro Eletrônico).
- GUEDES, Gilleanes T. A. **UML 2**: uma abordagem prática. 2. ed. São Paulo: Novatec, c2011. 484 p. ISBN 9788575222812. (Disponível no Acervo).
- **IEEE TRANSACTIONS ON SOFTWARE ENGINEERING**. New York: IEEE Computer Society, 1975-. Mensal,. ISSN 0098-5589. Disponível em: <https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=32>. Acesso em: 19 jul. 2024. (Periódico On-line).
- SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, c2019. xii, 756 p. ISBN 9788543024974. (Disponível no Acervo).
- WAZLAWICK, Raul Sidnei. **Análise e design orientados a objetos para sistemas de informação**: modelagem com UML, OCL e IFML. 3. ed. Rio de Janeiro, RJ: Elsevier, Campus, c2015. 462 p. ISBN 9788535279849. (Disponível no Acervo).



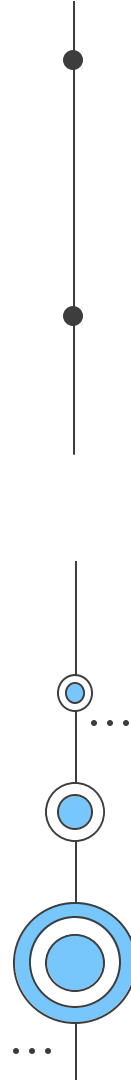


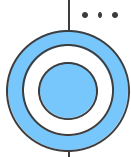
# Projeto de Software

## Referências complementares:



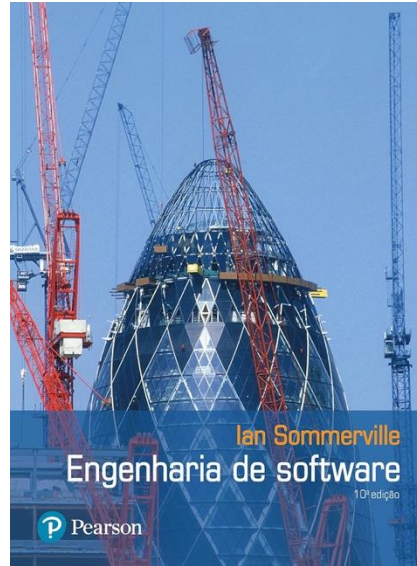
...





# Projeto de Software

## Referências complementares:



# Obrigado!

Dúvidas?

joaopauloaramuni@gmail.com



[GitHub](#)



[LinkedIn](#)



[Lattes](#)

...