

MBA⁺

**ARQUITETURA DE
SOLUÇÕES**

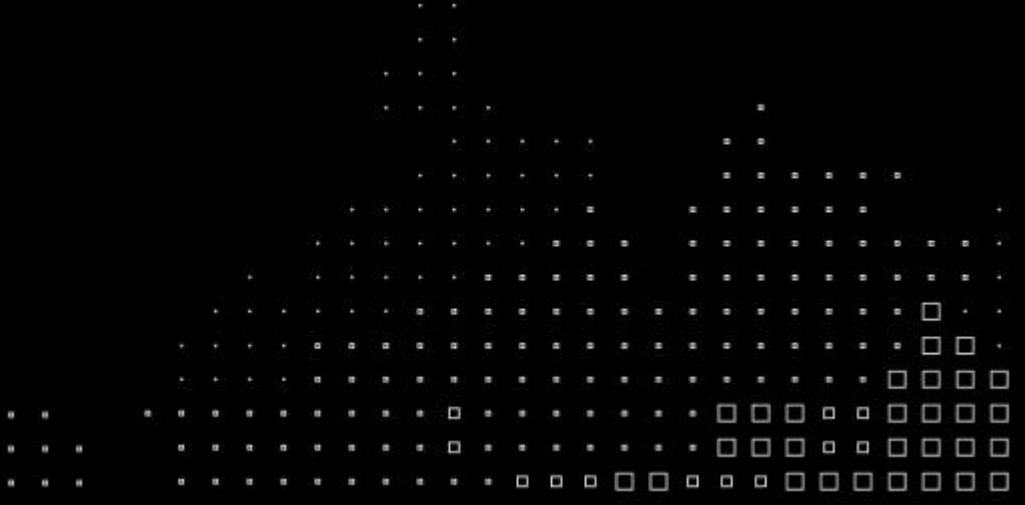
**MBA⁺**

IT Architecture Design & Styles

Aula 1

Prof. Leonardo Pinho

Email: profleonardo.pinho@fiap.com.br





Leonardo Pinho

Enterprise Architect Specialist - Banco Votorantim

Professor MBA FIAP

- Mestrado em Engenharia da Computação e Elétrica – Mackenzie (Em Curso)
- MBA Executivo – Administração e Marketing – Insper - (Em Curso)
- MBA Executivo – Finanças – Insper
- MBA - Arquitetura de Soluções – FIAP
- Redes de Computadores – Universidade Nove de Julho
- Técnico em Informática – Senac - SP

FIAP



LinkedIn



Professor MBA FIAP – Cursos:

- Arquitetura de Soluções
- Business Agility & Emerging Technologies Management



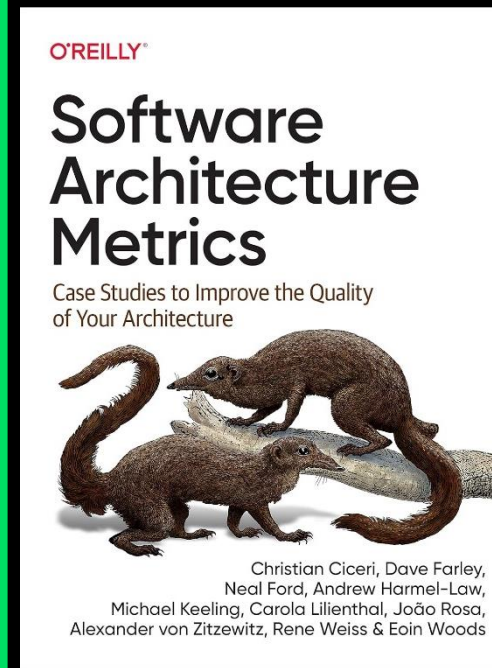
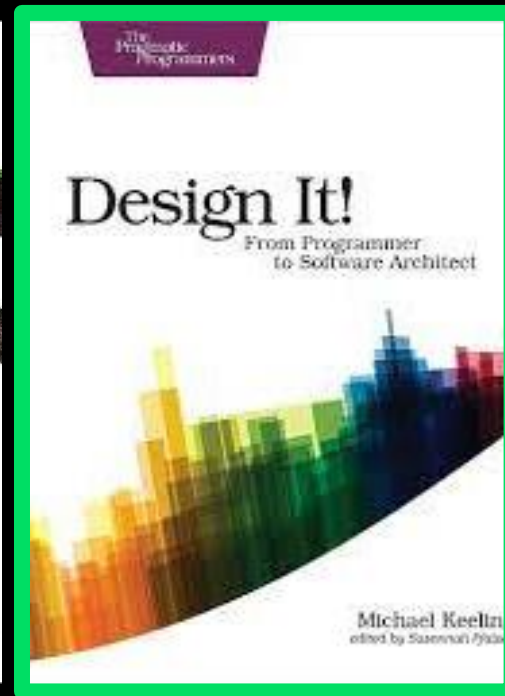
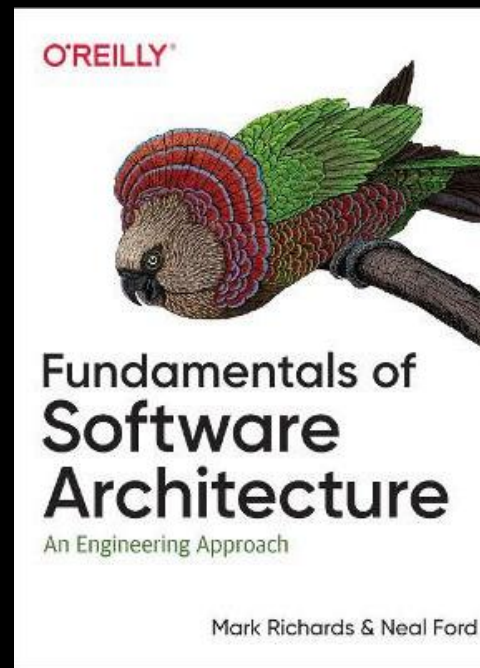
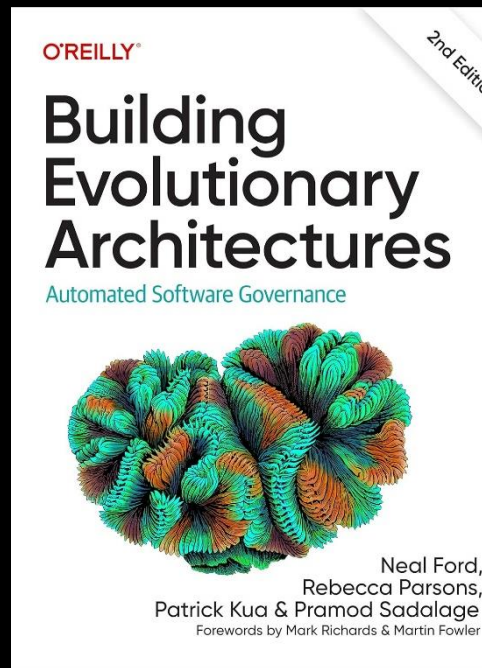
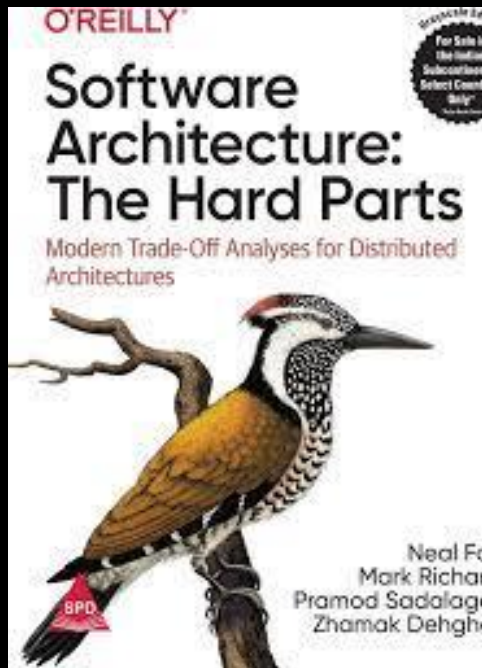
- Entusiasta da tecnologia por essência, extasiado por negócios e **corredor moderno**

- Mais de 15 anos de carreira com intensa experiência em:

- Metodologias ágeis;
- End-to-to end de SLDC's complexas;
- Liderança de times digitais;
 - Concepção, levantamento de requisitos, construção, lançamento, entrega, implantação e integrações de aplicações e produtos;
- Conhecimento em tecnologia e negócios em diversas áreas tais como:
 - Finanças, Telecomunicações, Varejo e Saúde.
 - Processos e Fluxos de Negócios, Frameworks de Arquitetura Corporativa, estilos



Você se considera um Bookworm? Segue Essa Bibliografia!



Nosso Foco!

Ou você é um PodcastAddict!



CONTEÚDO PROGRAMÁTICO

Fundamentos de Arquitetura de Software

Fundamento de Design Thinking;

Como Antecipar as mudanças em uma estratégia de Design;

Como prover o engajamento dos Stakeholders/Shareholders e parceiros;

Como definir Requisitos Significativos;

Como escolher uma arquitetura ideal para seu projeto;

Padrões de Arquitetura e Gerenciamento de Complexidade;

Escolhendo Design Adequado e Decisões de Design;

Avaliação da Arquitetura e Capacitação dos Arquitetos do seu time;

Estilos Arquiteturais + **Plus do Prof.**

CONTEÚDO PROGRAMÁTICO POR AULA

<p>Fundamentos de Arquitetura de Software</p> <p>Aula 01</p>	<p>Fundamentos de Design Think</p> <p>Aula 01</p>	<p>Antecipar as mudanças em uma estratégia de Design</p> <p>Aula 01</p>	<p>Hands On!</p> <p>Aula 01</p>
<p>Engajamento dos Parceiros</p> <p>Aula 02</p>	<p>Definindo requisitos</p> <p>Aula 02</p>	<p>Definindo a Arquitetura Ideal</p> <p>Aula 02</p>	<p>Hands On! / Kahoot I</p> <p>Aula 02</p>
<p>Padrões de Arquitetura de Gerenciamento de Complexidade</p> <p>Aula 03</p>	<p>Escolhendo Design Apropriado</p> <p>Aula 03</p>	<p>Decisões de Design</p> <p>Aula 03</p>	<p>Hands On!</p> <p>Aula 03</p>
<p>Avaliação da Arquitetura</p> <p>Aula 04</p>	<p>Outras Notações Arquiteturais</p> <p>Aula 04</p>	<p>Capacitação dos Arquitetos do seu time</p> <p>Notação C4</p> <p>Aula 04</p>	<p>Hands On! / Kahoot II</p> <p>Aula 04</p>

METODOLOGIA APLICADA

Aula expositiva para apresentação de conteúdo por meio da apresentação de slides com exemplos acadêmicos alinhado ao mercado;

Debates em sala de aula visando troca de experiências com presenças participativas e engajamento dos alunos;

Aplicação de estudos de casos e Hands On;

Kahoot;

Feedbacks são sempre bem vindos;

Não me limito a responder questões sobre a disciplina, caso eu não saiba responder algo, trarei a resposta na aula seguinte;

Interagir e mostrar pontos de experiências é sensacional, afinal não existe certo ou errado em arquitetura.

Meu objetivo com essa matéria é que vocês sejam mais analíticos. Mesmo que não concorde, pense (Think over) e reflita.

AVALIAÇÃO PARCIAL

Realizar a Entrega das atividades em cada aula.



Critérios	Pontuação
Entrega Aula 1 - I	1 Ponto
Entrega Aula 2 - II	1 Ponto
Entrega Aula 3 - III	1 Ponto
Entrega Aula 4 – I	1 Ponto
Ponto Extra – Presença em todas as aulas + Participação	1 Ponto

AVALIAÇÃO FINAL

Gravar um vídeo explicando a aplicabilidade do desenho de um problema real com PDF adicional com a documentação do projeto. **Vale 10 pontos**

Obs. Subir em um Repo no Github compartilhado do grupo, pois o prof. se importa com o portfólio de vocês.

Temas Sugeridos:

- Open Banking;
- Drex (Real Digital);
- Agregador de Contas;
- SVA's (Serviços de Valor Agregado);
- Antecipação de Recebíveis;
- E-commerce;
- Entrega/transporte por aplicativo.



LEGENDAS IMPORTANTES



Exercícios

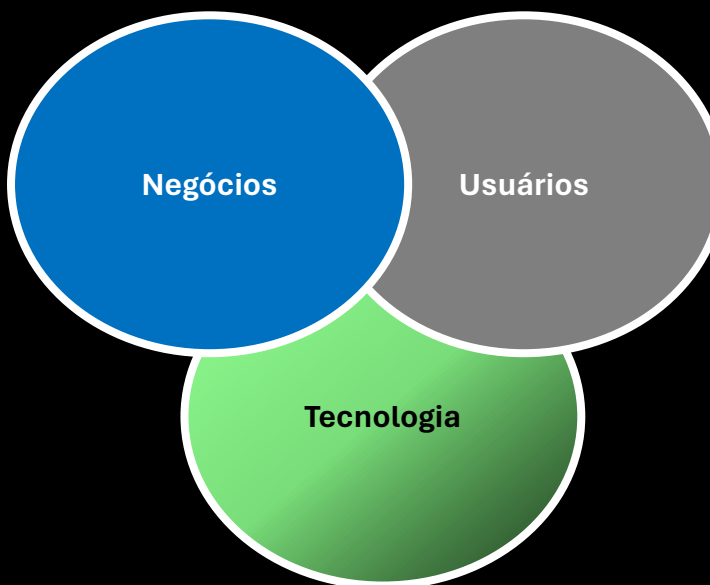


Artigos

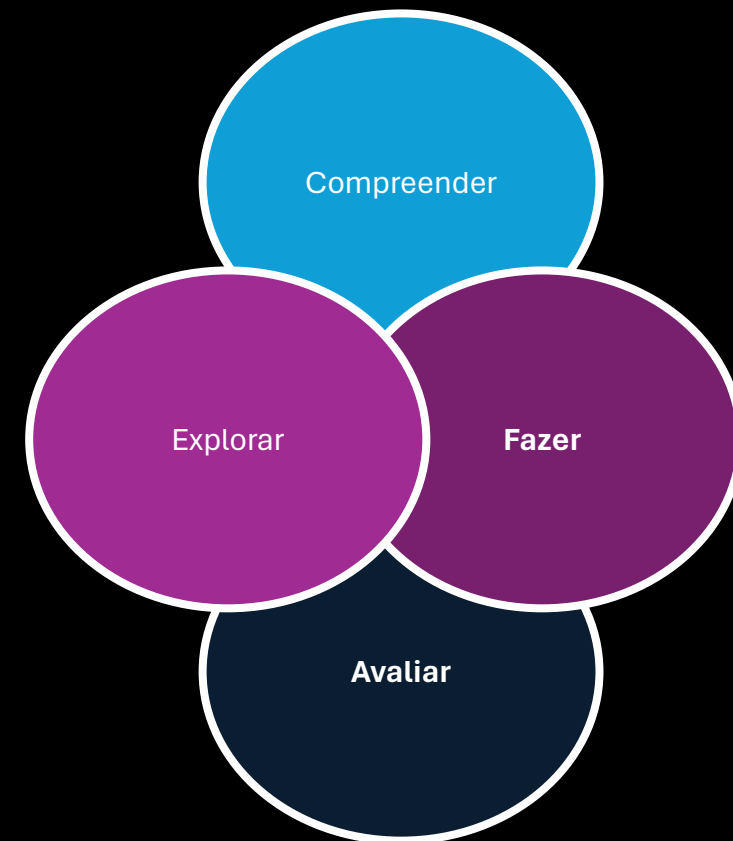
O que é Arquitetura de Software?

A arquitetura de software de um sistema envolve decisões de design importantes que organizam o software para garantir a qualidade e atender plenamente aos requisitos de negócios.

Composição



Mentalidade de design



O que é um arquiteto de Software?

Gerente de projeto?



Ajudar a Codar?



Arquiteto de Software



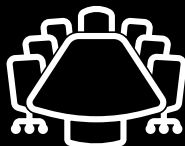
Gerente de produto?



Levantamento de Requisitos



Alinhamento com os Stakeholders / Shareholders?

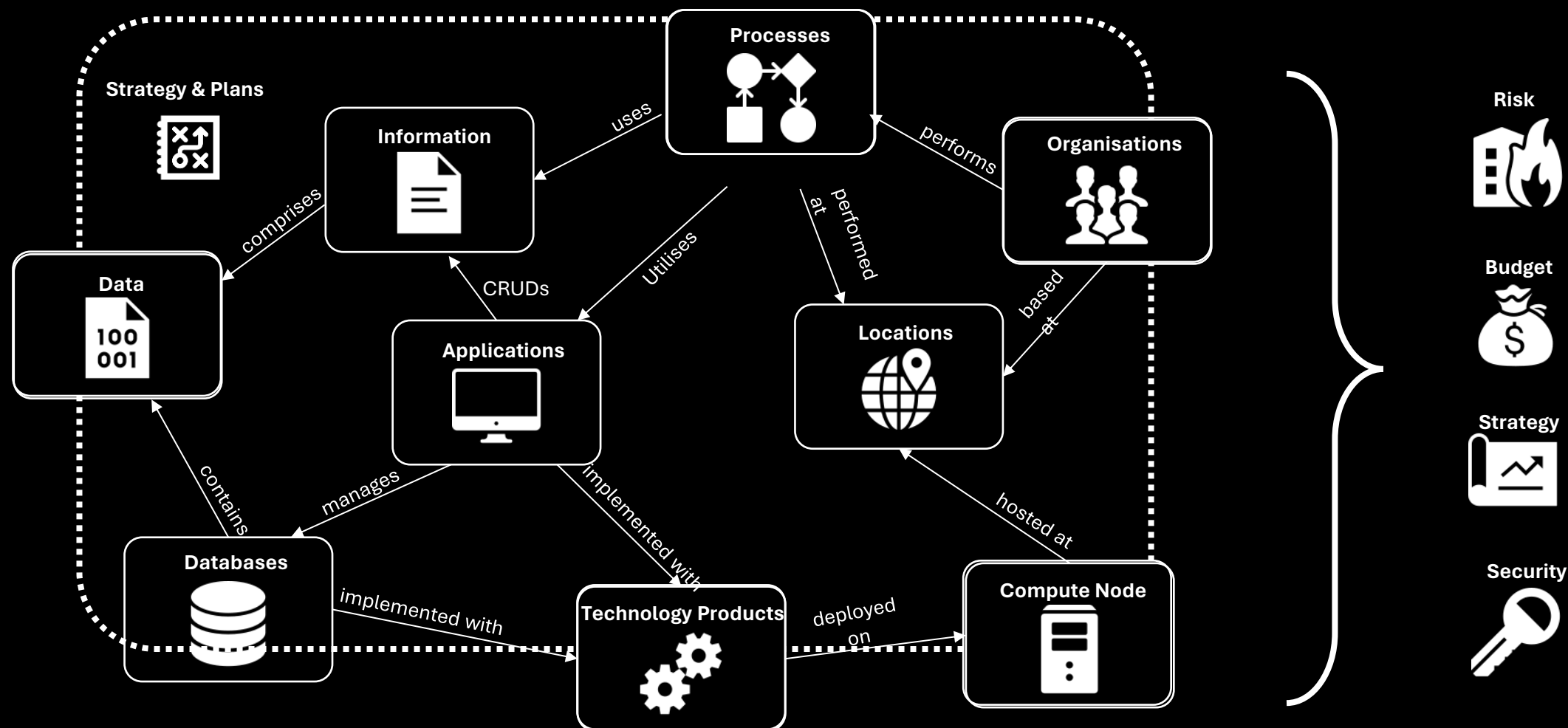


Definir Priorizações PO?



Software em um contexto de sistema

Funcionamento do ecossistema em uma organização!





Pilares importantes em escolhas arquiteturais

Time to market



Segurança



Performance



Disponibilidade



Escalabilidade



Manutenibilidade



Acessibilidade



User experience



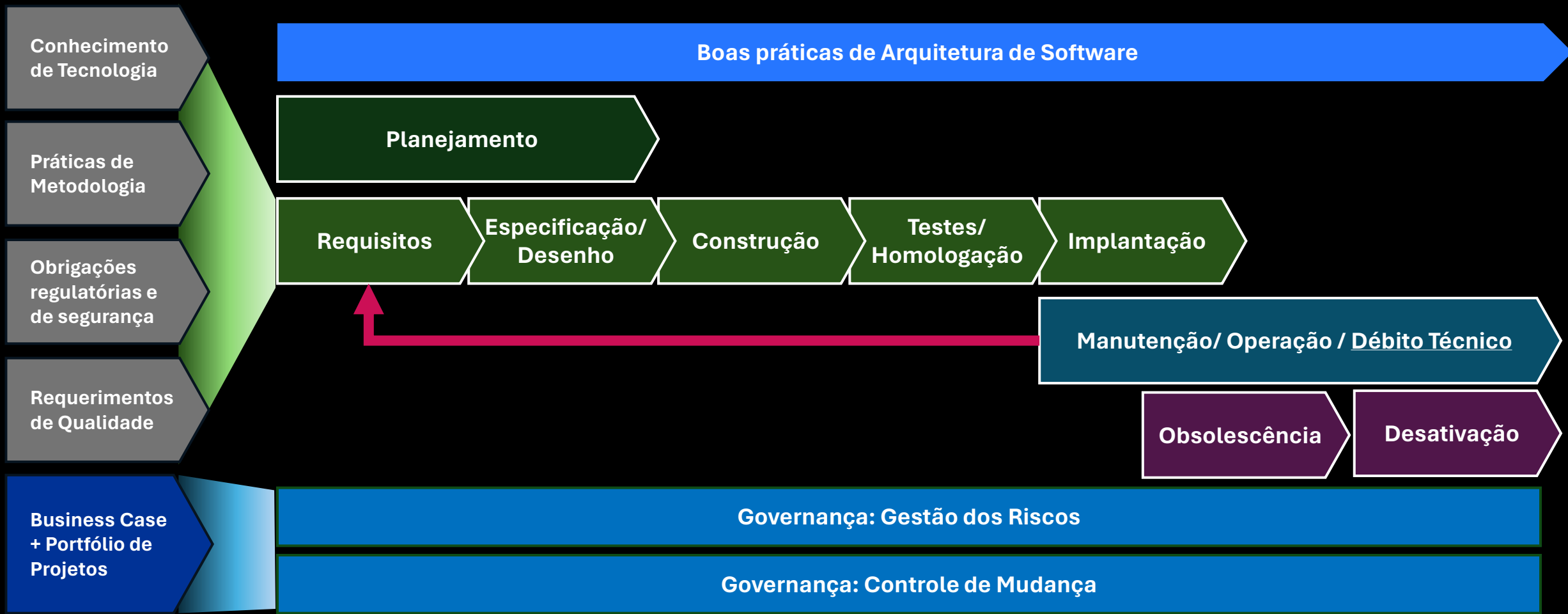
Domínio tecnológico



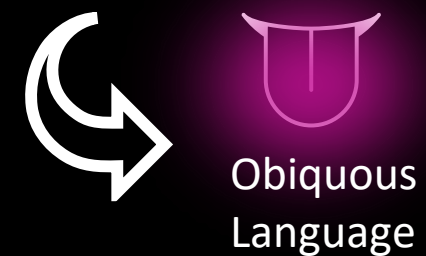
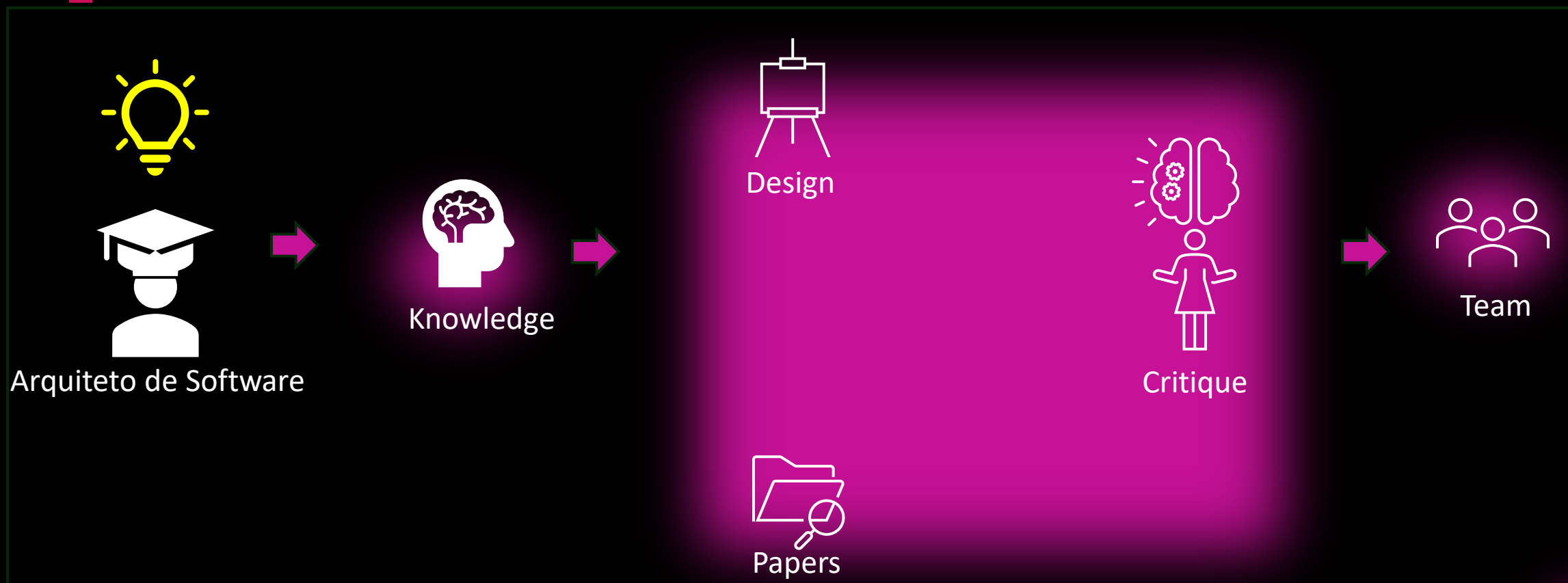
Elasticidade



Ciclo de vida de uma Aplicação

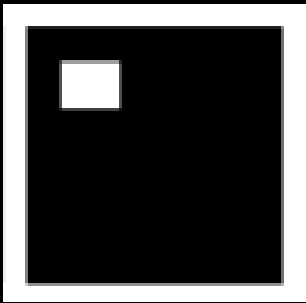


Evangelização do time de Arquitetura





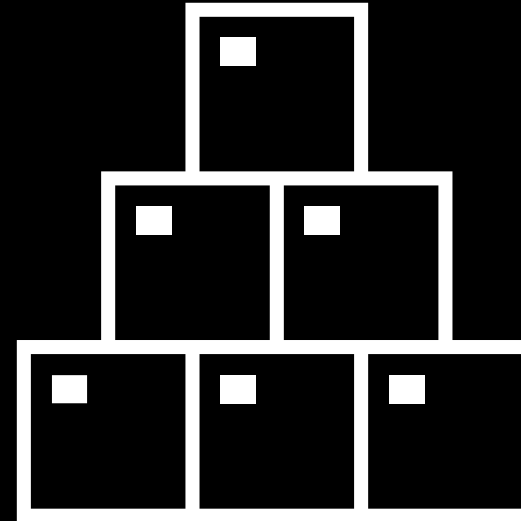
Componentes x Módulos



Componentes

- Idéia em tempo de execução;
 - **Exemplos tipo:** Objeto; Conexão; Thread; Camada; Processo; Servidor de...
 - **Exemplos de uso:** Ligar;; Assinar; Canalizar; Publicar; Retornar

- Dependendo do momento de projeto caso não se encaixe em nenhum dos dois, podemos chamar de elemento.



Módulos

- Elementos em tempo de Design;
 - **Exemplos tipo:** Classe; Pacote; Camada; Procedimento Armazenado; Módulo; Arquivo de Configuração; Tabela de Banco de dados.
 - **Exemplos de uso:** Relacionado ao..., permitido para ..., dependem de...

Perguntas Poderosas do Arquiteto no Início do Projeto

Quem são os principais Stakeholders?

Após a Implementação, quem será responsável pela sustentação?

Quais os principais riscos do projeto?

Quais são os principais objetivos da organização com esse projeto?

Estamos transformando grandes problemas em problemas menores?

Como é o As IS?

O que podemos aproveitar desse projeto para os próximos?

Quais as tecnologias estão envolvidas?

Quais as restrições?





O que é Design Thinking

É menos um processo;
Mais uma forma de pensar sobre problema de soluções a partir da perspectiva das pessoas afetadas por eles.



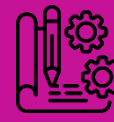
Empatizar



Definir



Ideação



Protótipo



Teste



Solução

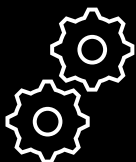
Princípios do Design Thinking

H



Regra Humana

A



Ambiguidade

R



Regra

T

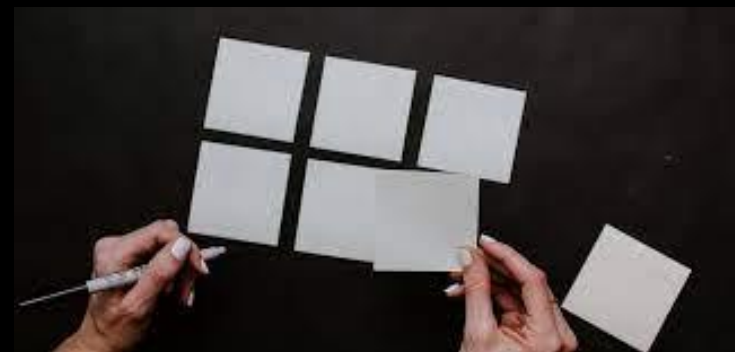


Tangibilidade

C

S

D



Certezas

Suposição

Dúvidas

Os quatro princípios do Design Thinking

Adote uma mentalidade de design

Temos quatro mentalidades de design: **entender, explorar, fazer e avaliar.**

Devemos aprender sobre as pessoas que serão afetadas pelo nosso sistema e o que elas precisam?
Se colocar nos sapatos do Stakeholder.

Entender

Avalie

Como saber se uma decisão de design resolverá o problema? Quando adotamos a *mentalidade de avaliação*, determinamos a adequação de nossas decisões de design em relação ao nosso entendimento atual.

Explorar

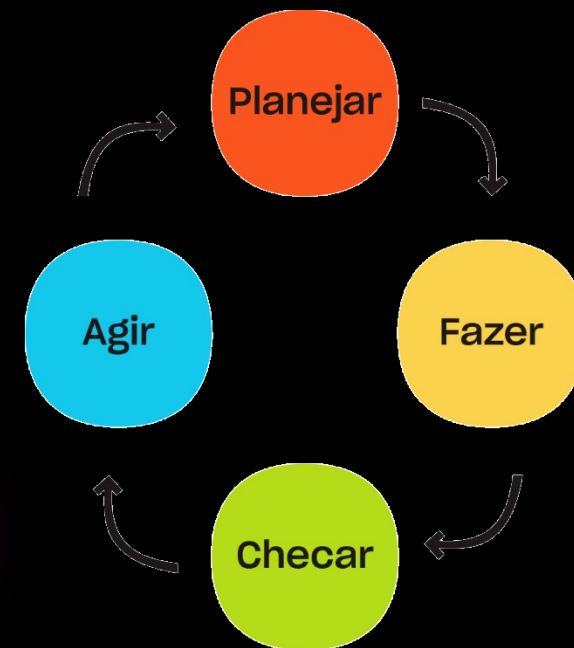
Fazer

Tente combinações de estruturas até encontrar uma combinação que melhor promova os **atributos de qualidade** desejados

Pesquise uma ampla gama de **padrões**;
Pesquise **tecnologias** e **práticas de desenvolvimento**;

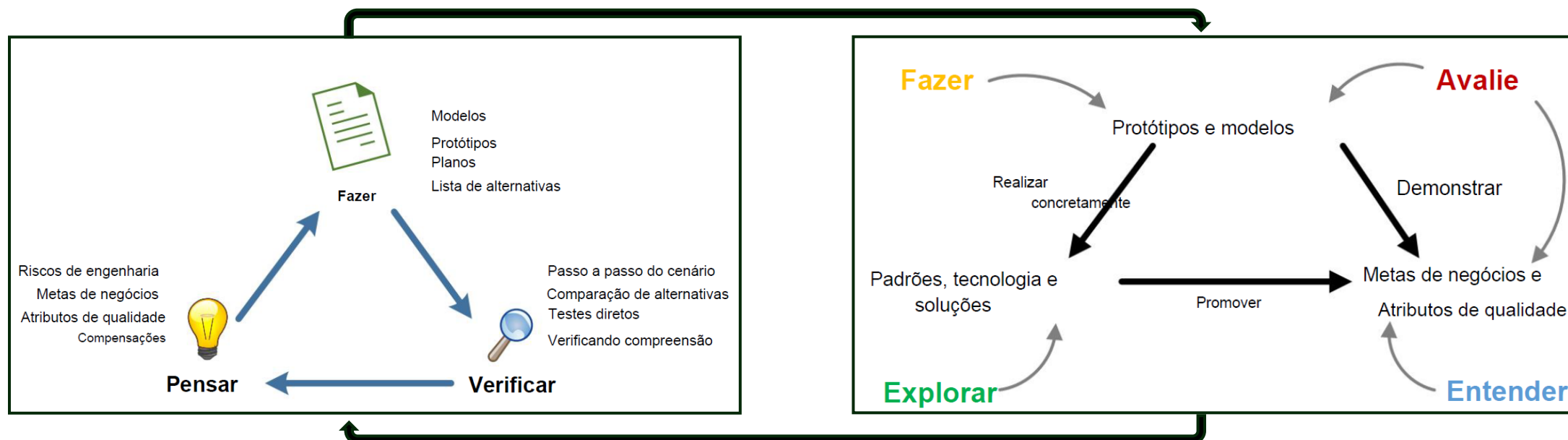



Torne isso real: A maneira mais comum de tornar a arquitetura real é criando modelos, não é sobre você e sim sobre o projeto.



Ciclo PDCA (Analogia)

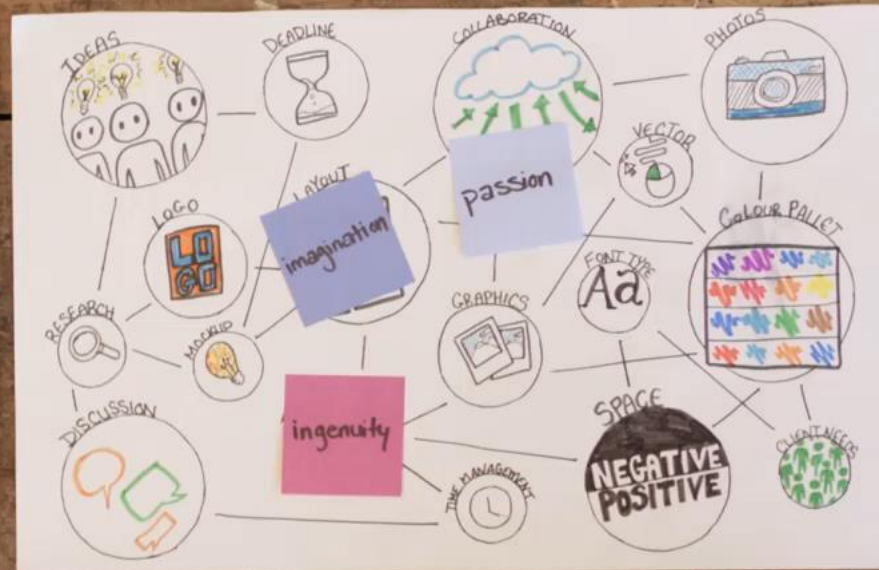
Pense, faça, verifique...



 **Pense:** O que esperamos aprender? Que perguntas precisamos que sejam respondidas? Quais são os nossos principais riscos? Pensar envolve criar um plano para aprender o que precisamos para responder a perguntas específicas ou reduzir riscos.

Atividades

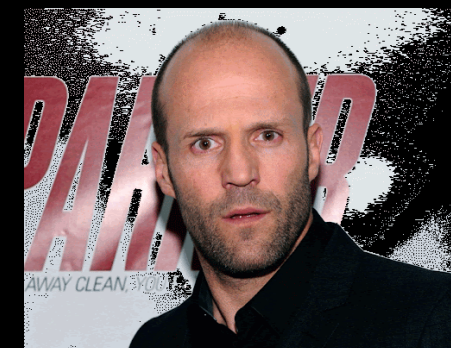
Exercício 1



Projeto do Programa

Uma empresa pública enfrenta desafios financeiros e precisa cortar custos.

O patrocinador **Jason Stathan** contratou nossa equipe para agilizar as operações da Secretaria de Gestão e Orçamento (OMB), pensando que isso poderia ser utilizado para a iniciativa privada também.



Atualmente, quando um funcionário municipal precisa fazer uma compra significativa, o OMB publica uma Solicitação de Propostas (RFP) no jornal local.

Empresas competem nas RFPs e o OMB concede contratos com base na competitividade das propostas e outros fatores.

O OMB gerencia mais de 500 contratos e RFPs para diversos itens, como papel higiênico, suprimentos médicos e bolas de basquete, usando planilhas.

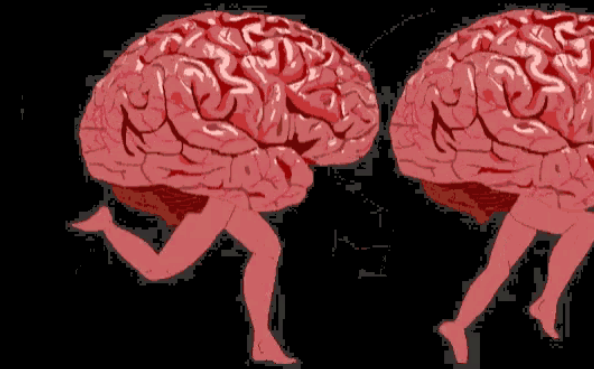
Jason Stathan espera que a modernização do OMB melhore aspectos estratégicos, como:

- **Concorrência:** Mais da metade das RFPs recebem apenas uma proposta, resultando em custos elevados e serviços de qualidade inferior.
- **Tempo de Finalização:** Finalizar um contrato leva meses, fazendo com que muitas empresas desistam do processo.
- **Publicação de RFPs:** A publicação de uma nova RFP pode levar até 6 semanas, um processo que precisa ser mais rápido.

Atividade

Pontos principais para se considerar e pensar nesse projeto:

- ☐ O que esperamos aprender com esse projeto?
- ☐ Que perguntas precisamos que sejam respondidas?
- ☐ Quais são os nossos principais riscos?
- ☐ Crie um plano para aprender o necessário para responder a perguntas específicas;
- ☐ Crie um plano para reduzir riscos.



Elabore uma estratégia de design

Tempo de desenvolvimento

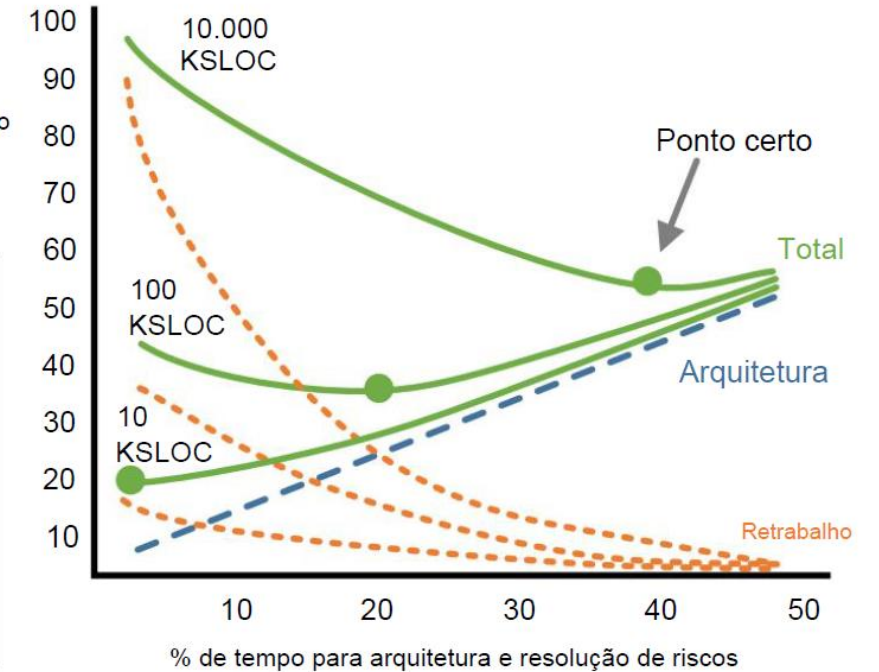
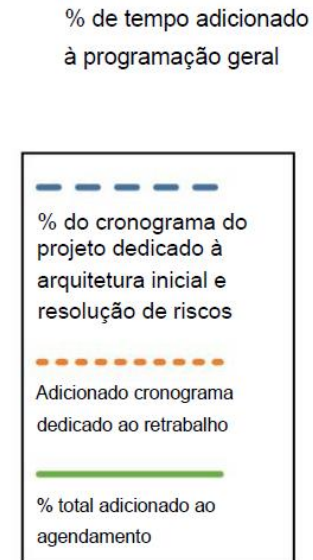
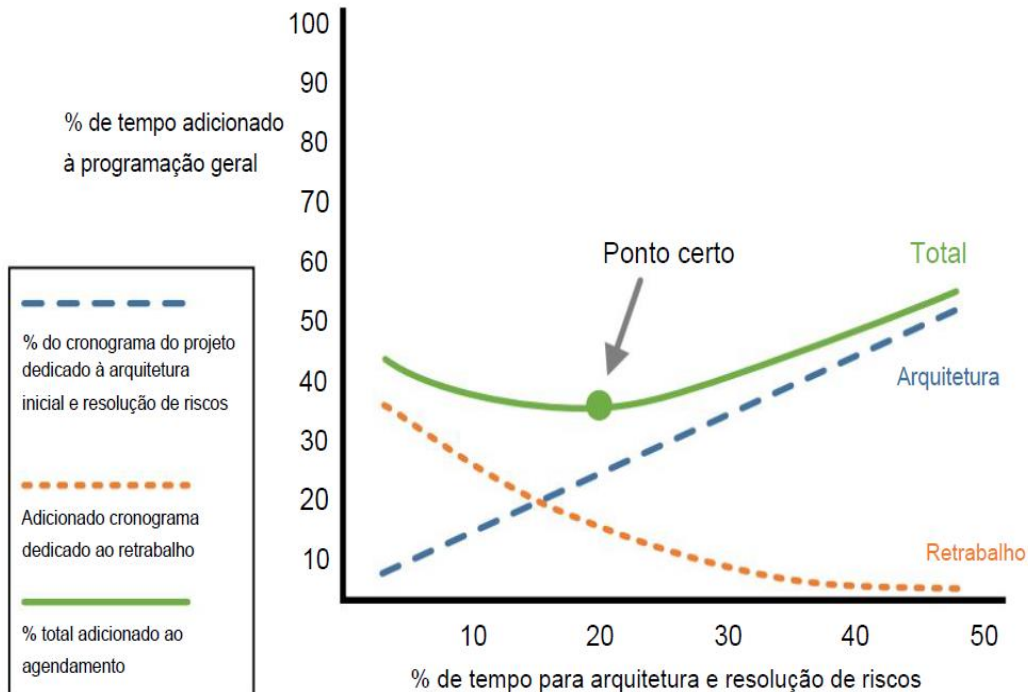
Arquitetura e Tempo de Redução de Risco

+ Tempo de retrabalho (corrigir defeitos, reescrever, erros)

Tempo total do projeto

Nota: O tempo gasto na arquitetura pode acelerar o desenvolvimento e reduzir o retrabalho!

Encontre o ponto ideal do design



Dias gastos em arquitetura	Dias gastos em retrabalho	Dias em Cronograma Total
5	38	143
17	21	138
33	7	140

*"KSLOC" significa "mil linhas de código fonte" e é uma medida comum usada para avaliar o tamanho e a complexidade de um software. Esta medida é útil para estimar a carga de trabalho e os recursos necessários para desenvolver ou dar suporte a um determinado software.

Elabore uma estratégia de design

Identifique condições e consequências

Aqui está uma declaração de risco no formato condição-consequência. Conhecer a condição e as consequências cria gatilhos para decidir o que fazer em relação ao risco.



Analogia com Burritos



Um novo restaurante de burritos abriu do outro lado da rua do meu escritório;
colegas de equipe podem ficar doentes comendo muitos burritos.

Condição, algo verdadeiro hoje

Consequência, algo ruim que pode acontecer

Declaração de Risco Ruim

- ➡ Inauguração de um novo restaurante de burritos.
- ➡ A equipe pode exagerar nos burritos.
- ➡ Comer muitos burritos pode dar dor de barriga.
- ➡ Se um colega do time comer muitos burritos, esse colega vai ficar doente.

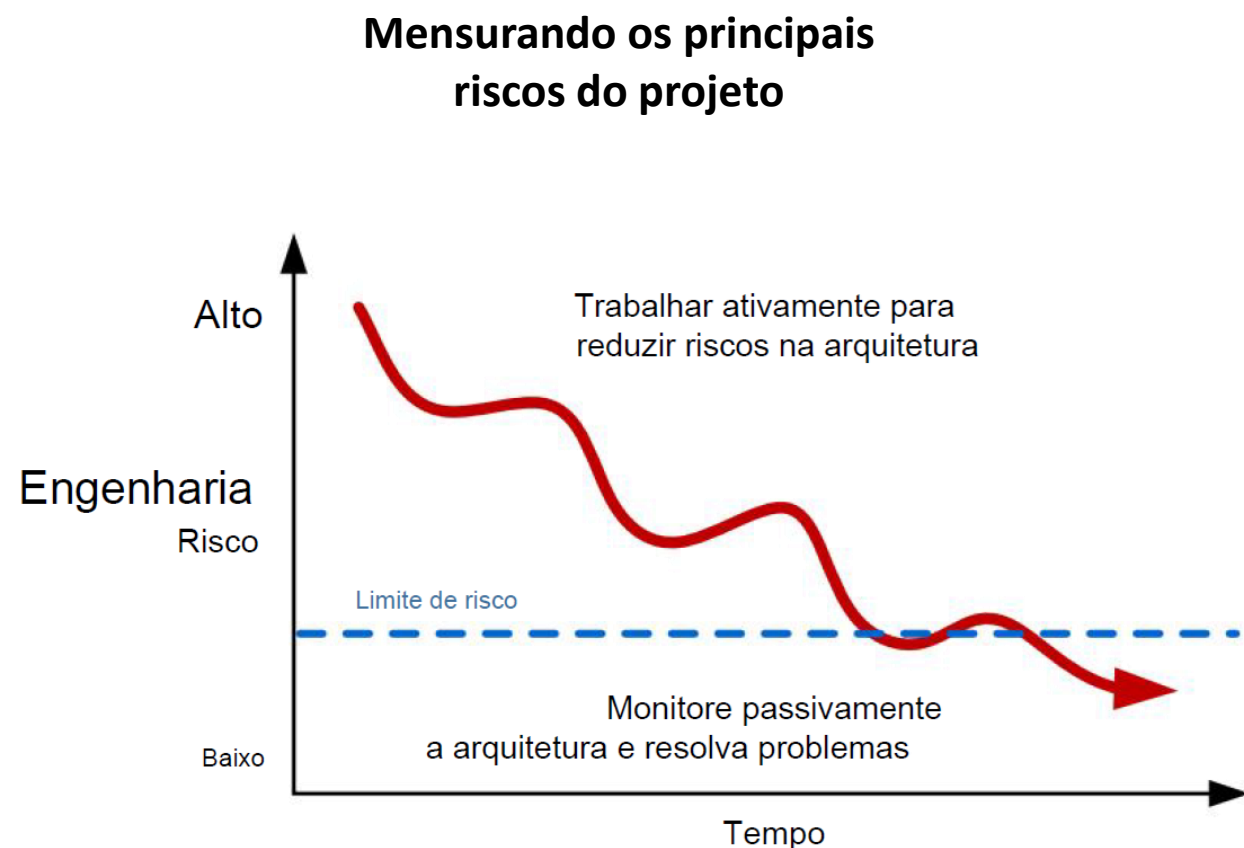
Por que é?

- ➡ Ruim e daí? O Impacto negativo não é claro. Na verdade, isso parece ótimo, pois nós amamos burritos
- ➡ Parece ruim? Será que precisamos nos preocupar com isso agora?
- ➡ É verdade, mas o que isso tem a ver com a minha equipe?
- ➡ Caso um meteoro caia no escritório, ficaremos doentes também? O que nos levou a se preocupar com os burritos?

Crie uma estratégia de design

Perguntas para nos ajudar a decidir qual mentalidade de design pode ser apropriada para um respectivo projeto.

Tentar...	Se...
➡ Entenda a mentalidade	➡ O risco é sobre o problema? Você precisa de uma compreensão mais profunda com os stakeholders ou outros usuários?
➡ Explore a mentalidade	➡ O Risco da Solução? Já avaliamos soluções alternativas suficientes?
➡ Crie a mentalidade	➡ O Risco é sobre comunicação? Os Stakeholders entendem completamente os conceitos de design em jogo e conseguem ver a arquitetura?
➡ Avalie a mentalidade	➡ O risco envolve uma decisão de projeto ou uma adequação? Precisamos tomar uma decisão de design?



Crie uma plano de design

Mas o que é o Básico da Arquitetura? O que é considerado como mínimo? Vamos falar sobre conceitos importantes.



VS?



MVA é a arquitetura menos complexa e básica que pode sustentar seu produto em uma fase de MVP (e às vezes, também, além).

O MVA é importante porque permite que você se concentre no design mínimo do sistema necessário para fazer seu MVP funcionar. Ao mesmo tempo, garante que você esteja atento ao sistema que está projetando.

Um MVP se concentra em provar a viabilidade de um produto e a adequação ao mercado, implementando a menor funcionalidade necessária e com o menor esforço. Um MVA é semelhante, mas concentra-se na arquitetura de software do produto (desde que o produto seja um aplicativo).

MVPs são voltados para fora. Eles são o que o usuário experimenta. **MVAs são voltados para dentro.** Eles estão ocultos do usuário.

Dúvidas?



Nossa Professor, já foi a primeira aula?



<https://www.linkedin.com/in/leonardo.pinho-1/>



profleonardo.pinho@fiap.com.br

FIAP MBA⁺



MBA⁺



Copyright © **2024** Prof. Leonardo Pinho / Prof. Marcos Macedo

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, da Instituição FIAP e do Professor Leonardo Pinho.