

CÓDIGO DE ALTA PERFORMANCE - MOBILE

UNIDADE I: ARQUITETURA DE DISPOSITIVOS MÓVEIS

Apresentação

Nome: IGOR WENNER SILVA FALCÃO

Lattes: <http://lattes.cnpq.br/6677376621642966>

Graduação:

- Sistemas de Informação (UFPA);

Mestrado:

- Engenharia Elétrica - Computação Aplicada (PPGEE)

Doutorado:

- Engenharia Elétrica - Computação Aplicada (PPGEE)

Pesquisador científico: UFPA

Professor Graduação: Computação



- **CODIFICAÇÃO NATIVA, ARQUITETURA DE DISPOSITIVOS MÓVEIS.**
- **NOTIFICAÇÕES PUSH.**
- **LISTENERS, CONTENT PROVIDERS NATIVOS E SERVICES PROVIDERS PERSONALIZADOS.**

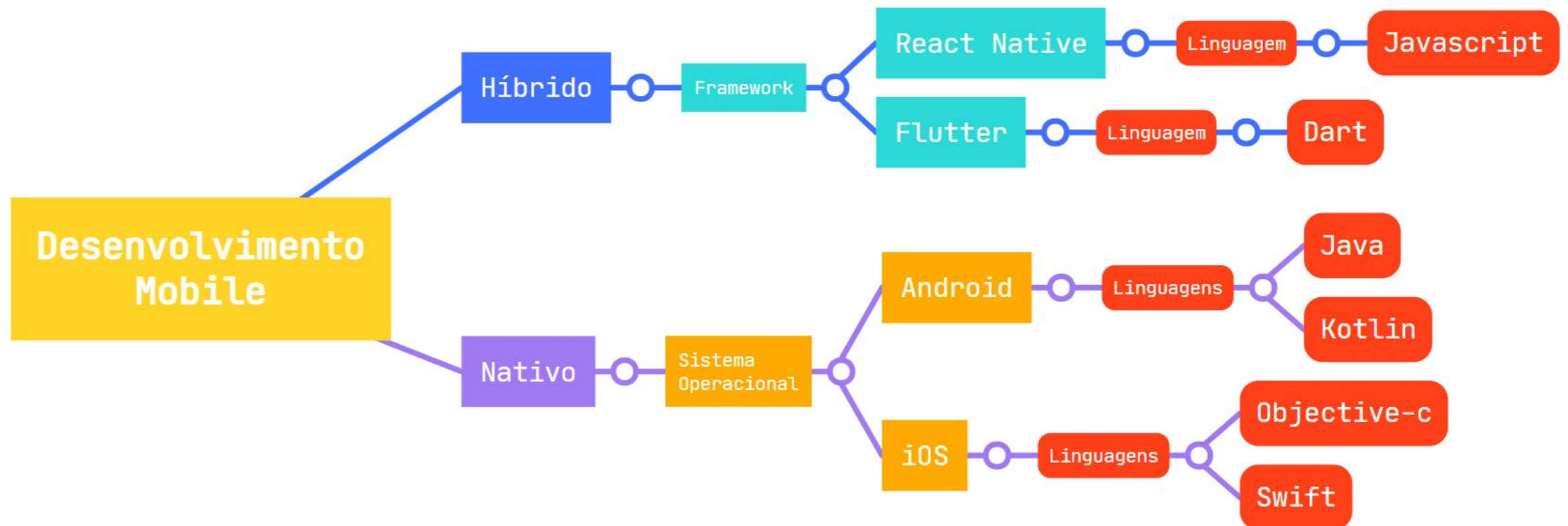
- **CODIFICAÇÃO NATIVA, ARQUITETURA DE DISPOSITIVOS MÓVEIS.**
- NOTIFICAÇÕES PUSH.
- LISTENERS, CONTENT PROVIDERS NATIVOS E SERVICES PROVIDERS PERSONALIZADOS.

Codificação Nativa

- Refere-se ao desenvolvimento de aplicativos móveis usando as linguagens de programação e ferramentas específicas do sistema operacional nativo de um dispositivo.
- Significa que você está escrevendo o código do aplicativo usando a linguagem de programação.
- Utilização de linguagens e as APIs (Interfaces de Programação de Aplicativos).



Codificação Nativa



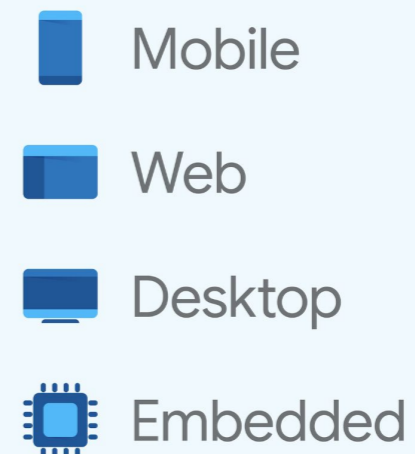
Multiplataforma:



Codificação Híbrida

Flutter

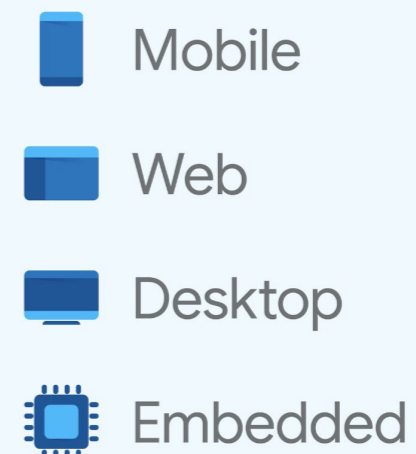
- Flutter é um framework criado para o desenvolvimento de aplicativos mobile, tanto para sistema Android como para iOS. O seu objetivo é facilitar o desenvolvimento por meio da transformação do código-fonte em código-nativo



Codificação Híbrida

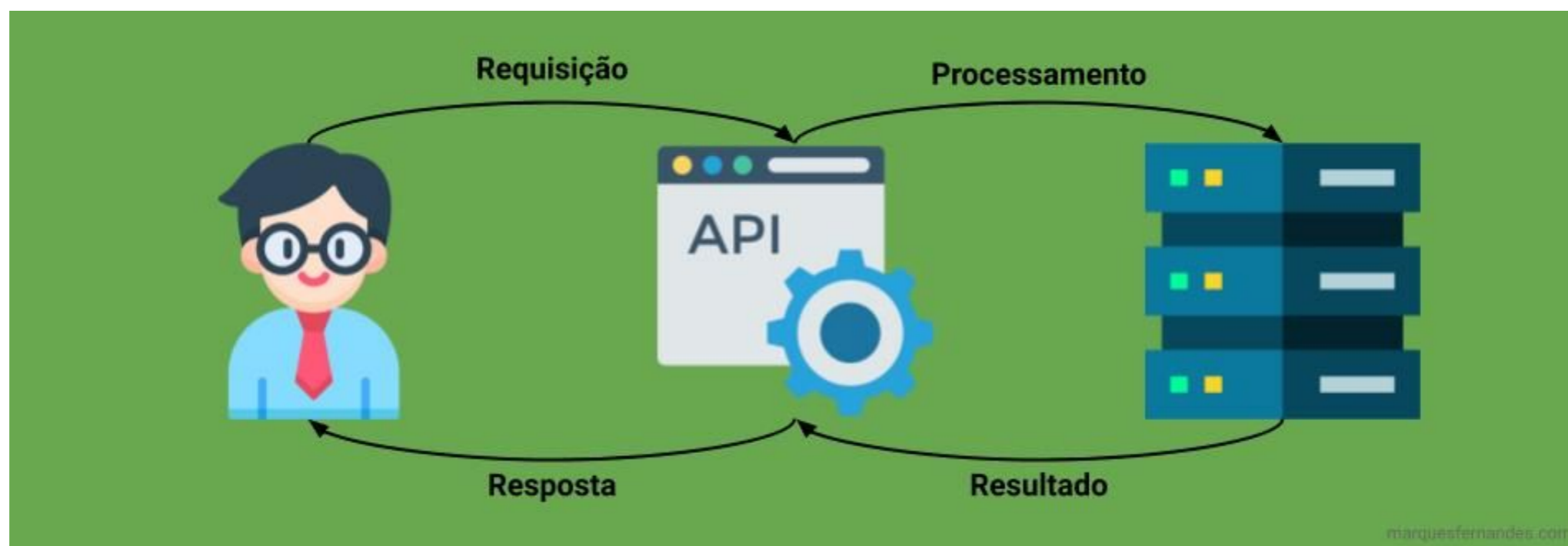
Flutter

- O Flutter usa a linguagem de programação Dart e inclui um rico conjunto de widgets personalizáveis, que são os componentes visuais e de interação dos aplicativos.
- Ele compila diretamente em código nativo, o que permite alta performance e uma experiência fluida no uso dos aplicativos.



Codificação Nativa

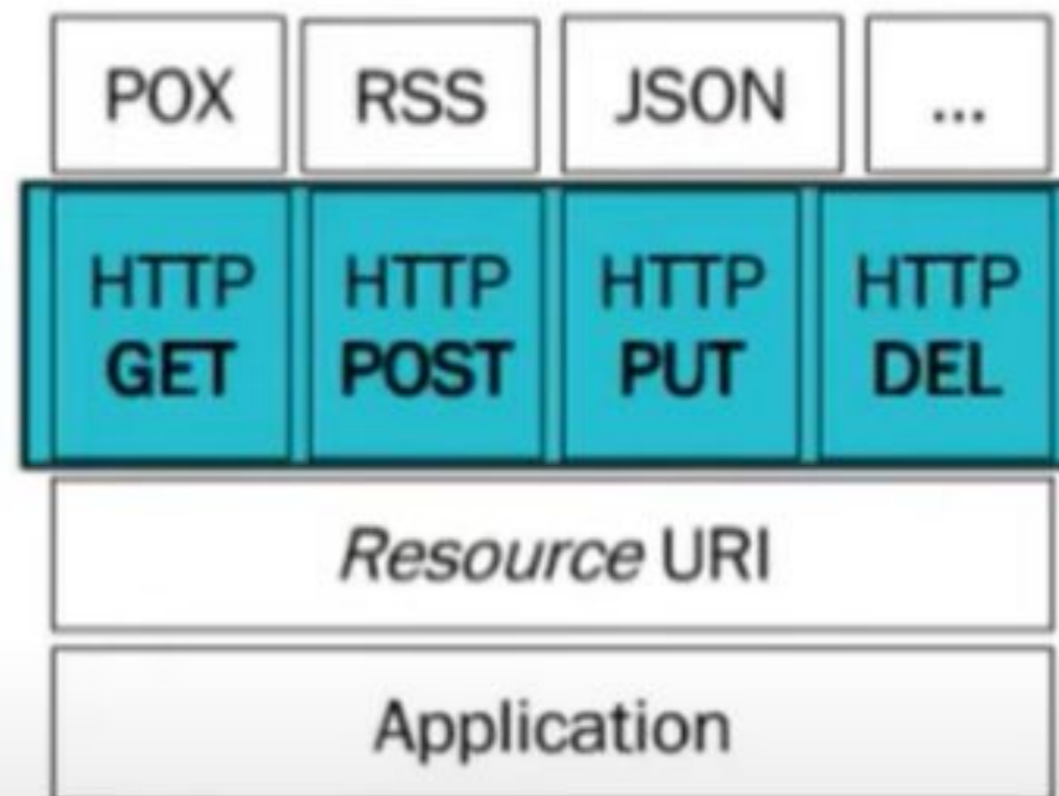
- **APIs** (Interfaces de Programação de Aplicativos).
 - É um conjunto de funções que fornecem alguma capacidade de negócio ou técnica e pode ser chamada por aplicativos usando um protocolo definido (IBM, 2021)
 - Exemplos: API REST e API SOAP



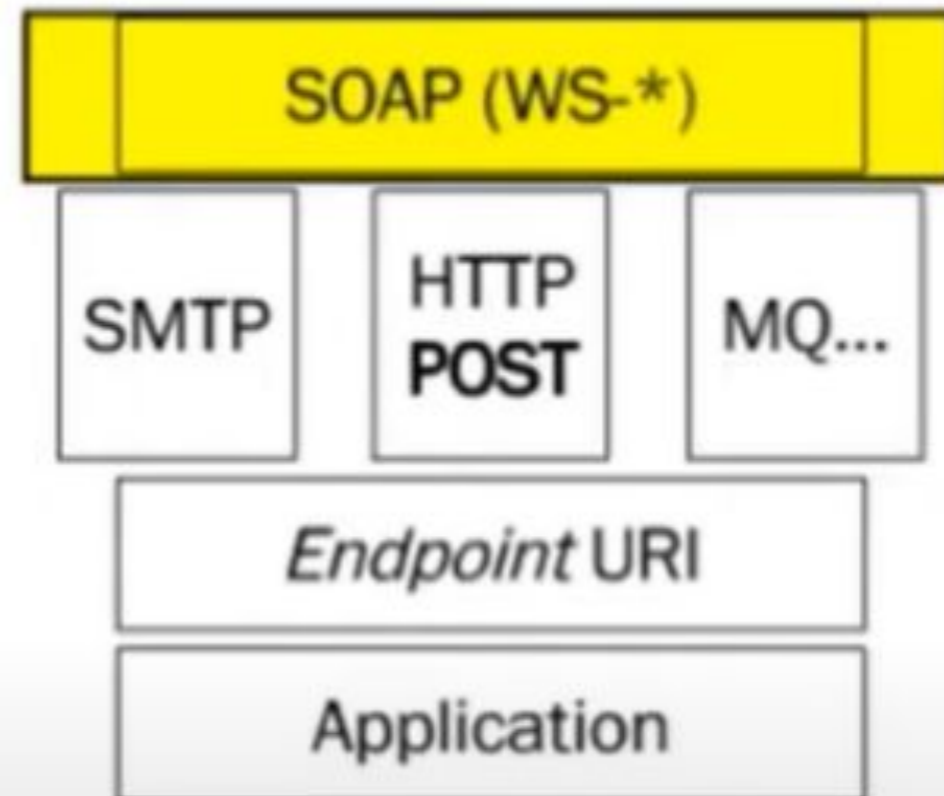
Codificação Nativa

- **APIs** (Interfaces de Programação de Aplicativos).
 - Exemplos: API REST e API SOAP

REST



SOAP



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Acesso a Recursos Avançados
- Adoção de Plataformas Múltiplas



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Adoção de Plataformas Múltiplas

APIs permitem que os desenvolvedores utilizem funcionalidades e recursos já existentes em vez de criar tudo do zero



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Adoção de Plataformas Múltiplas

Ao utilizar APIs, os desenvolvedores podem se concentrar nas partes específicas e únicas do aplicativo, em vez de gastar tempo construindo recursos comuns



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Adoção de Plataformas Múltiplas

APIs permitem a integração de serviços de terceiros, como serviços de pagamento, redes sociais, mapas e muito mais



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Adoção de Plataformas Múltiplas

APIs bem projetadas facilitam a escalabilidade do aplicativo



Codificação Nativa

Benefícios de se utilizar APIs:

- Reutilização de Código
- Eficiência e Rapidez
- Integração de Serviços
- Escalabilidade
- Adoção de Plataformas Múltiplas

Se você deseja lançar seu aplicativo em várias plataformas (iOS, Android, web)



Codificação Nativa

A codificação nativa tem vantagens:

- Desempenho otimizado
- Acesso total às funcionalidades
- Melhor integração com a plataforma

Codificação nativa tem desvantagens:

- Desenvolvimento mais demorado
- Maior complexidade
- Custo mais elevado



Dispositivos Móveis

O que são dispositivos móveis?

De onde vem o termo MOBILE?

Dispositivos Móveis

História:

- Primeiros testes foram feitos entre o ano de 1947 e 1973.
- O nome do primeiro aparelho lançado pela Motorola era DynaTAC e era somente um protótipo.
- Primeiro modelo liberado ao público foi o DynaTAC 8000X em 1983.

Ficha Técnica

Fabricante: Motorola
Modelo: DynaTAC 8000x
Ano: 1983
Peso: 1Kg
Dimensão: 30 cm
Memória: 30 números
Autonomia: 1 hora
Preço: US\$ 3,995

US\$ 12,000 ou R\$ 20.000,00



Dispositivos Móveis

História:

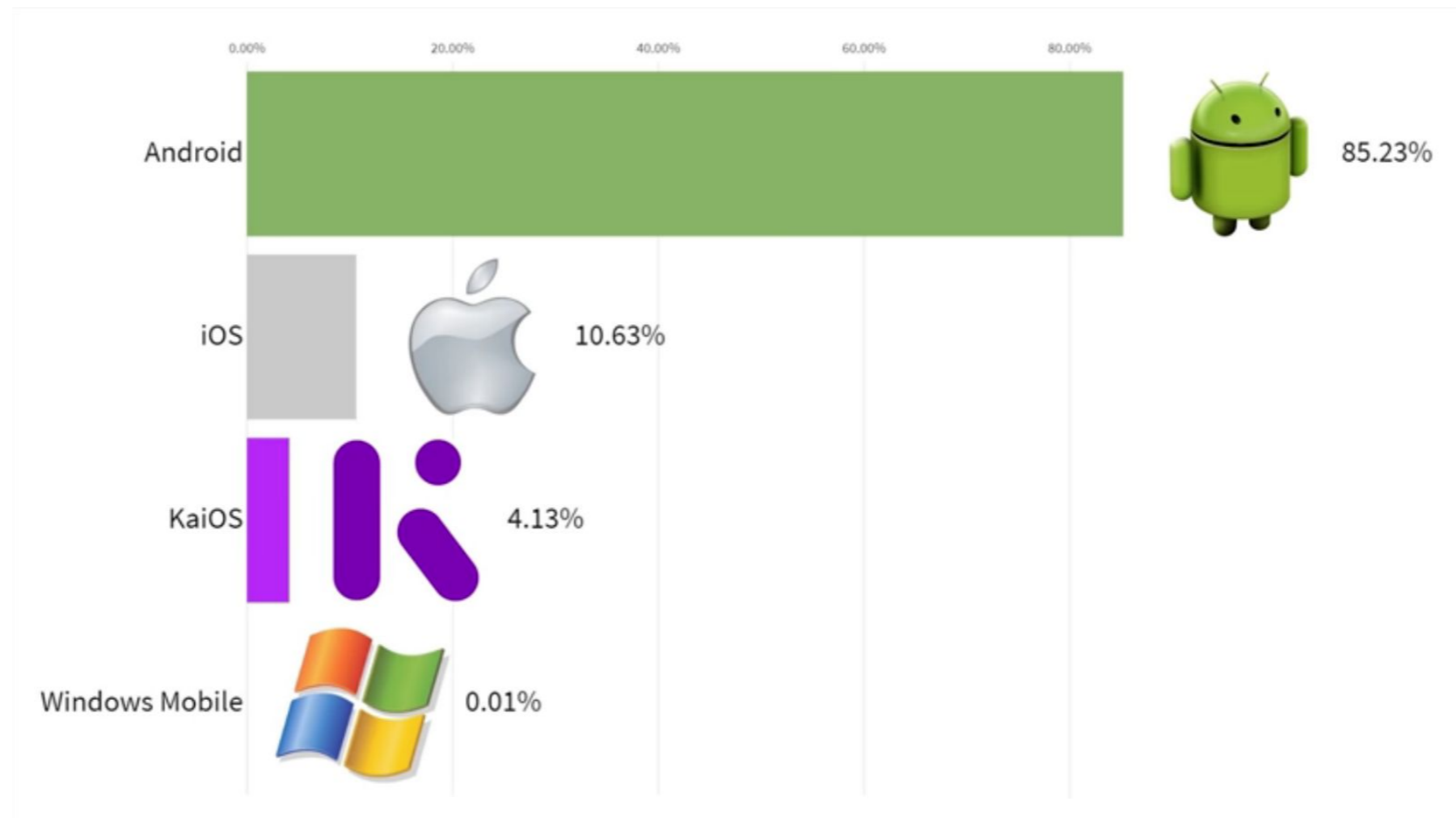
- Smartphone
- PDA (Personal Digital Assistant)
- Celular
- Console Portátil
- Televisão Portátil
- Aparelhos GPS (Sistema de posicionamento Global)



Dispositivos Móveis

Sistemas Operacionais (SO) dos Dispositivos Móveis:

- Um sistema operativo é um programa ou um conjunto de programas cuja função é gerir recursos de um sistema.



Dispositivos Móveis

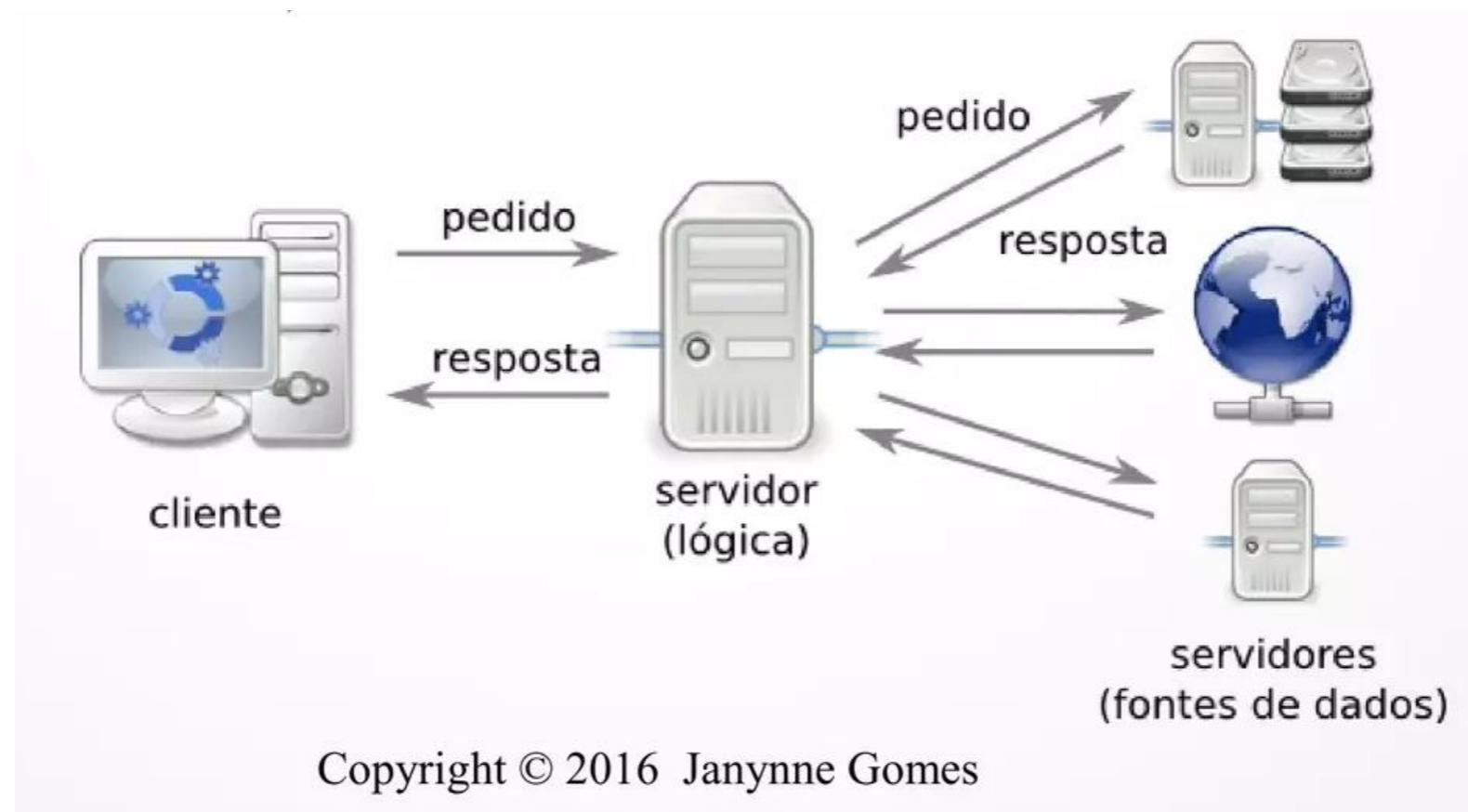
Sistemas Operacionais (SO) dos Dispositivos Móveis:

- Atualmente os principais sistemas operacionais existentes são: Java ME(em alguns celulares), Blackberry OS, Windows Mobile, Windows Phone, iOS, Symbian, WebOS, Android e Maemo, MeeGo, sendo os cinco últimos baseados em Linux.

Dispositivos Móveis

Arquitetura de Aplicação

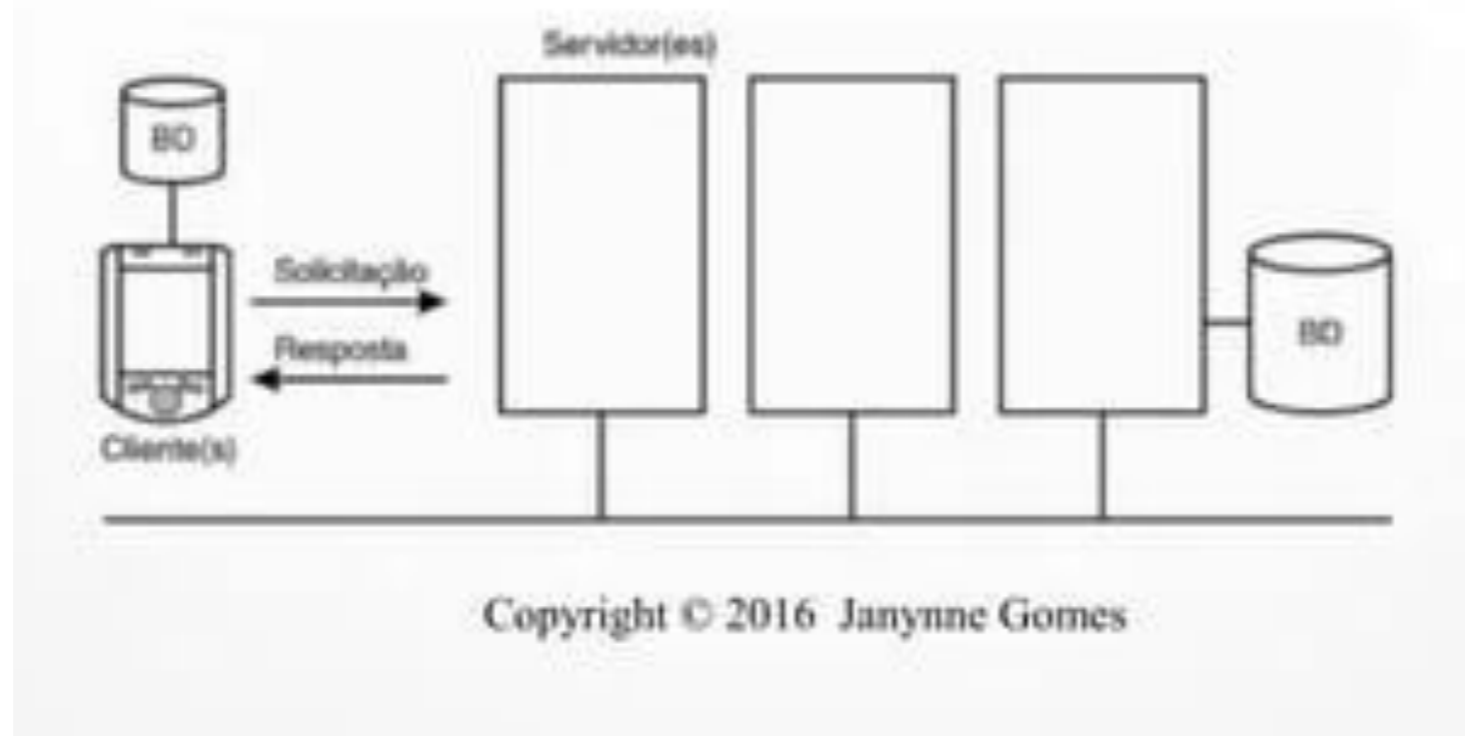
- São modelagens que ilustram layout total do software (Ex: código e Plataforma) e hardware (Ex: Cliente, servidor, dispositivos de rede).



Dispositivos Móveis

Arquitetura: Cliente-Servidor

- Um ou mais dispositivos clientes solicitam informações a um servidor
- Comunicação em camadas e filas



Arquitetura: Cliente-Servidor

Em camadas:

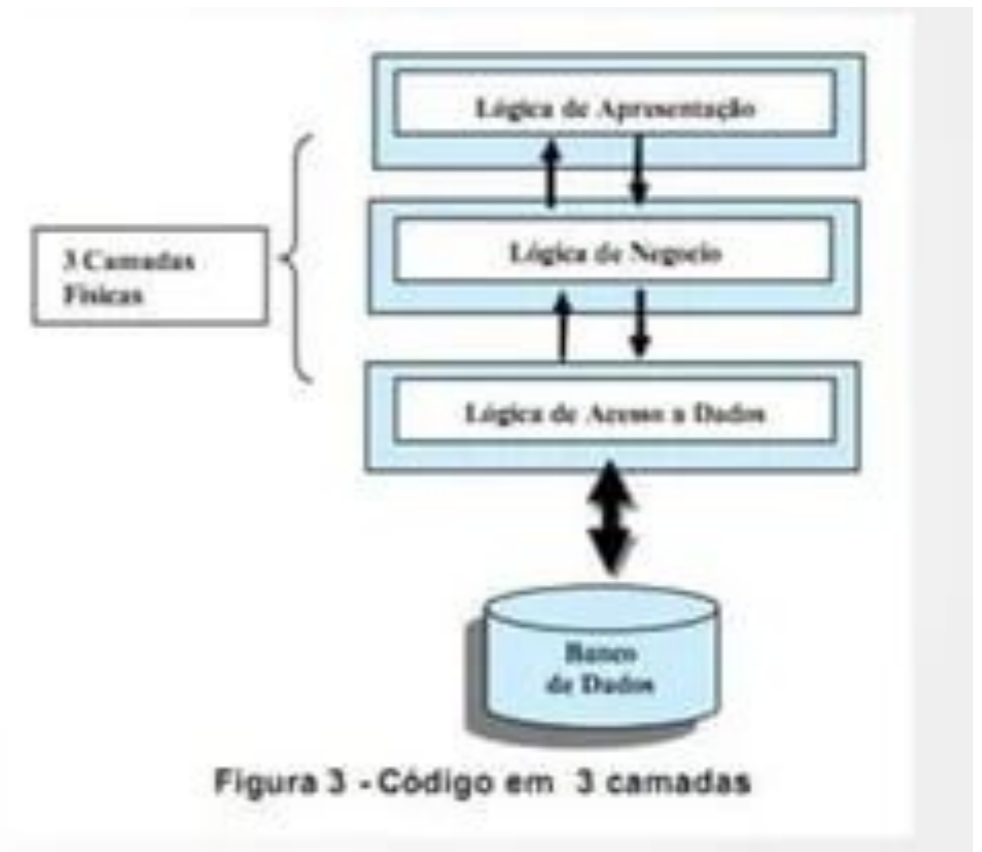
- Divide o trabalho dentro do código, separando as tarefas pertinentes;
- .Separam a lógica da aplicação, comunicação com banco de dados e a interface com o usuário;
- São elas: Apresentação, Negócios e Acesso a dados.
- Os clientes podem ser “magros” ou “gordos”:
- Clientes Magros: Não possuem camada com código personalizado da aplicação, todo o código fica no servidor.
- Clientes “Gordos”: Possuem um ou mais camadas localmente.

Dispositivos Móveis

Arquitetura: Cliente-Servidor

Em camadas:

- Apresentação: está mais próxima do usuário, é utilizada para exibir a interface com o usuário.
- Negócios: contém a lógica comercial do software.
- Acesso a dados: trata a comunicação com o banco de dados.



Dispositivos Móveis

Arquitetura: Cliente-Servidor

Em camadas:

- Camada de Aplicação: Apresentação das aplicações e serviços oferecidos aos usuários. Exploração de exemplos de aplicativos populares.
- Camada de Sistema Operacional: Importância do sistema operacional móvel (Android, iOS). Gerenciamento de recursos, segurança e multitarefa. Interação entre aplicativos e sistema operacional.

Dispositivos Móveis

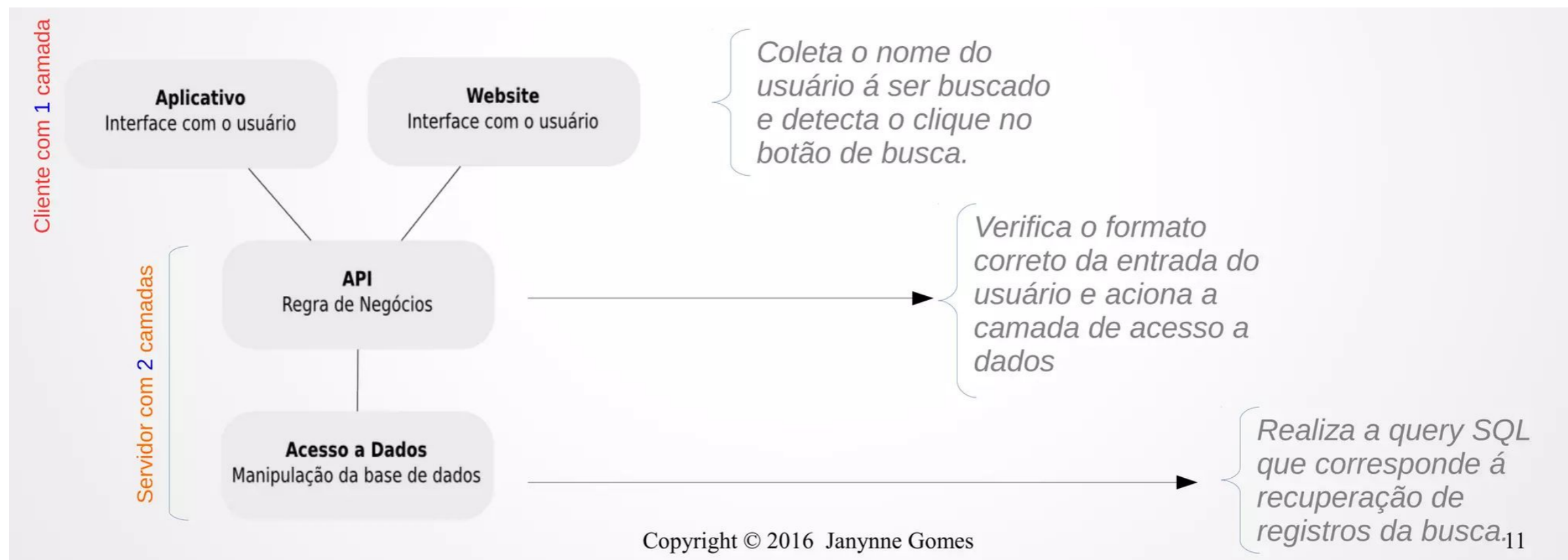
Arquitetura: Cliente-Servidor

Em camadas:

- Camada de Middleware: Bibliotecas e serviços intermediários para comunicação entre aplicativos e hardware. Exemplos de serviços, como notificações e serviços de localização.
- Camada de Hardware: Componentes físicos do dispositivo (CPU, GPU, sensores). Integração com camadas superiores da arquitetura.

Dispositivos Móveis

Arquitetura: Cliente-Servidor



Dispositivos Móveis

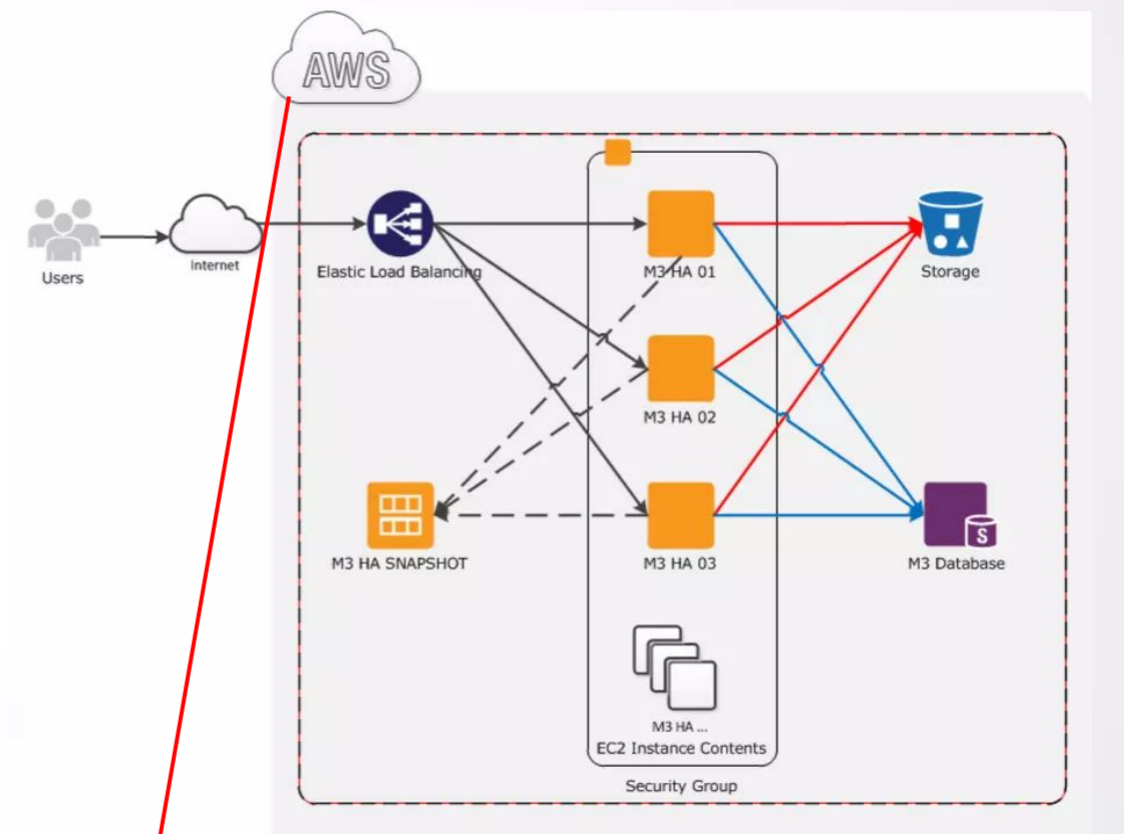
Arquitetura: Cliente-Servidor (EXEMPLO)

Cenário

Estamos na Black Friday e livraria Saraiva está recebendo muitos acessos simultâneos, tem muitos clientes querendo comprar na promoção.

O servidor de aplicação não aguenta responder e processar todas as solicitações.

Utilizando uma arquitetura escalável, basta subir mais servidores de aplicação para responder as demais solicitações.



A computação em nuvem é um modelo de tecnologia em que recursos de computação, como armazenamento, processamento e software, são fornecidos através da Internet.

Dispositivos Móveis

Componentes Essenciais dos Dispositivos Móveis:

- Processador (CPU) e sua influência no desempenho.
- Memória (RAM) e armazenamento interno.
- Bateria e estratégias para otimização do consumo.
- Tipos de tela e resolução.

Dispositivos Móveis

Desenvolvimento de Aplicativo

- Linguagens de Programação: Java, Swift, Objective C, Python, PHP, Ruby e JavaScript
- Framework: Ionic, React, Flutter e PhoneGap.
- Ferramentas: AppMachine, GoodBarber, BuildFire entre outros.
- Plataformas de Distribuição
- Monetização de Aplicativos:

Dispositivos Móveis

Futuro da Arquitetura Móvel

- **Necessidade de Adaptação:** arquiteturas precisam se ajustar às mudanças rápidas no cenário tecnológico.
- **Conectividade Ubíqua:** Suportará uma conectividade contínua em vários cenários.
- **Novos Modelos de Negócios:** arquiteturas móveis impactarão as indústrias.
- **Colaboração e Ecossistemas:** Discussão sobre a interconexão de dispositivos e serviços.

Referências Recomendadas

- MAZZA, Lucas. HTML5 e CSS3: domine a web do futuro. Editora Casa do Código, 2014.
- MARTIN, Robert C. Clean Architecture: A Craftsman's Guide to.
- TURINI, Rodrigo. PHP e Laravel: crie aplicações web como um verdadeiro artesão. Editora Casa do Código, 2015.
- MILANI, André. MySQL-guia do programador. Novatec Editora, 2007.
- SILVA, Maurício Samy. JavaScript-Guia do Programador: Guia completo das funcionalidades de linguagem JavaScript. Novatec Editora, 2020.
- BAHIT, Eugenia. Poo y mvc en php. El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC, 2011.

Obrigado

Dúvidas?

Prof. Msc. Igor Falcão

igorufpa2013.4@gmail.com