UNIVERSIDADE LUSÓFONA

Arquiteturas de Software

Prof. Pedro Perdigão
p5617@ulusofona.pt

Agenda

- Objetivos;
- Arquitetura de *Software*;
- O conceito de “Arquiteto de *Software*”;
- O desenvolvimento da Arquitetura de um *Software*;
- Conclusões.

Prof. Pedro Perdigão
p5617@ulusofona.pt

Motivação

- Projetos simples podem ser realizados por uma única pessoa…

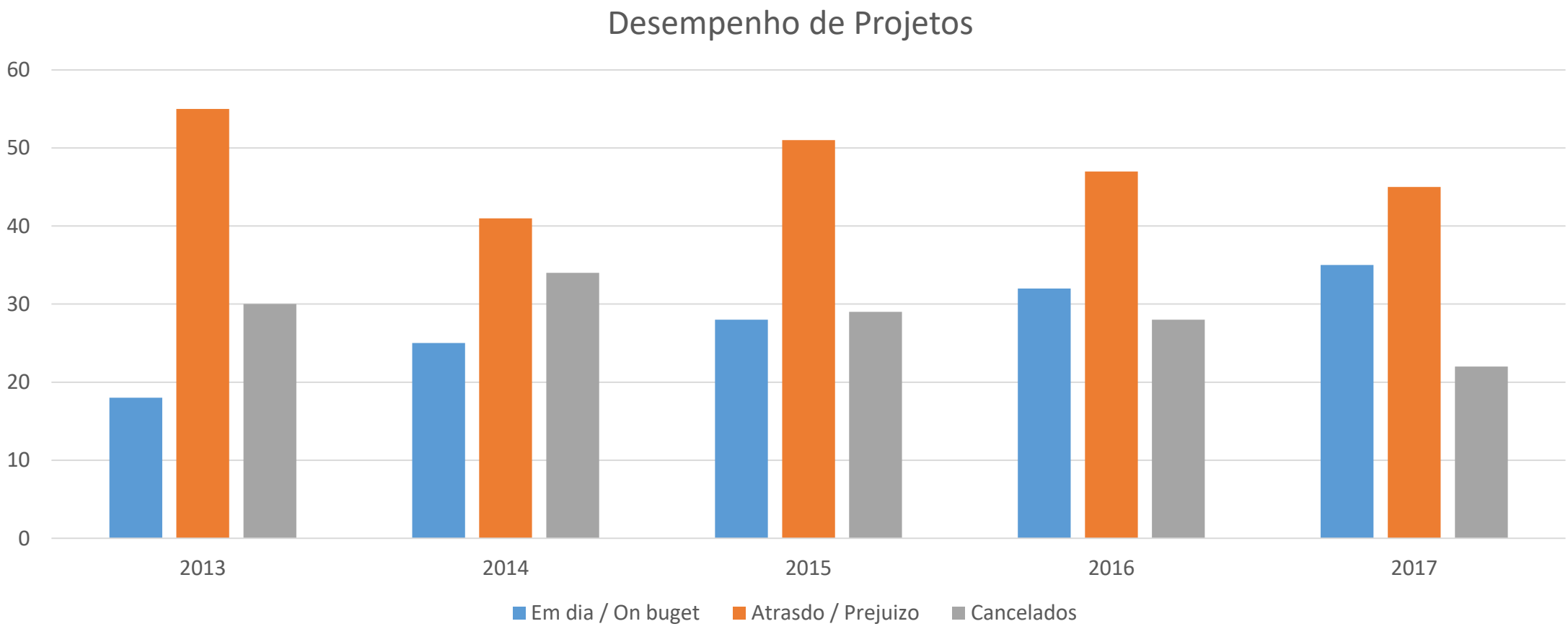
  - Pouca modelagem;
  - Ferramentas simples;
  - Processo simples;
  - Pouca especialização para construir.

Motivação

- Projetos complexos/maiores exigem arquitetura…

  - Mais modelagem;
  - Ferramentas mais poderosas;
  - Processos bem definidos;
  - Alta especialização para construir.

Motivação



Fonte: Standish Group – The Chaos Chronicles

Definição

- A Arquitetura de um *Software* é a estrutura do sistema que compreende:
  - Os elementos do *Software;*
  - O relacionamento entre estes elementos;
  - As propriedades externamente visíveis destes elementos.

Bass, Clements, and Kazman. Software Architecture in Practice 2nd ed, Addison-Wesley 2003.

Definição

- A Arquitetura é a organização fundamental de um sistema compreendida pelos:
  - Os seus componentes;
  - Os relacionamentos entre si;
  - Os relacionamentos com o ambiente/ecossistema;
  - Os princípios que guiam o seu desenho e evolução.

IEEE Recommended Practice for Architectural Description of Software-intensive Systems (IEEE Std 1471 / 2000).

Importância

- Obter a “visão geral” do sistema;
  - Compreender os elementos importantes do *Software.*
- Construir sistemas complexos e desafiadores;
- Aumentar a consistência e a escalabilidade do sistema;
- Documentar decisões de alto impacto para os *stakeholders;*
- Aumentar o reuso;
- Diminuir o *rework* e a redundância.

Importância

- Mitigar riscos cedo e continuamente;
  - Tecnológicos, Humanos, Negócio, Gestão, etc..

- Reduzir custos de desenvolvimento, manutenção e evolução do *Software;*
- Definir estratégias de gestão do desenvolvimento;
- Manter o foco em criar *Software* de qualidade.

A Arquitetura pode levar ao sucesso ou ao fracasso de um projeto de Software…

Arq. de Software – Contexto de Atuação

Arq. de Software – Contexto de Atuação

- Analista de Negócios
  - Integração especial quando está a lidar com visões e ligados a utilizadores finais.
- Gestor de projeto
  - Auxiliar no desenvolvimento de planos ou avaliá-los;
  - Organizar informação técnica, *feedback*, conselhos, avaliação de risco, etc…
- Especialistas em Tecnologia
  - Obter informações detalhadas sobre tecnologia – As suas aplicações e restrições.

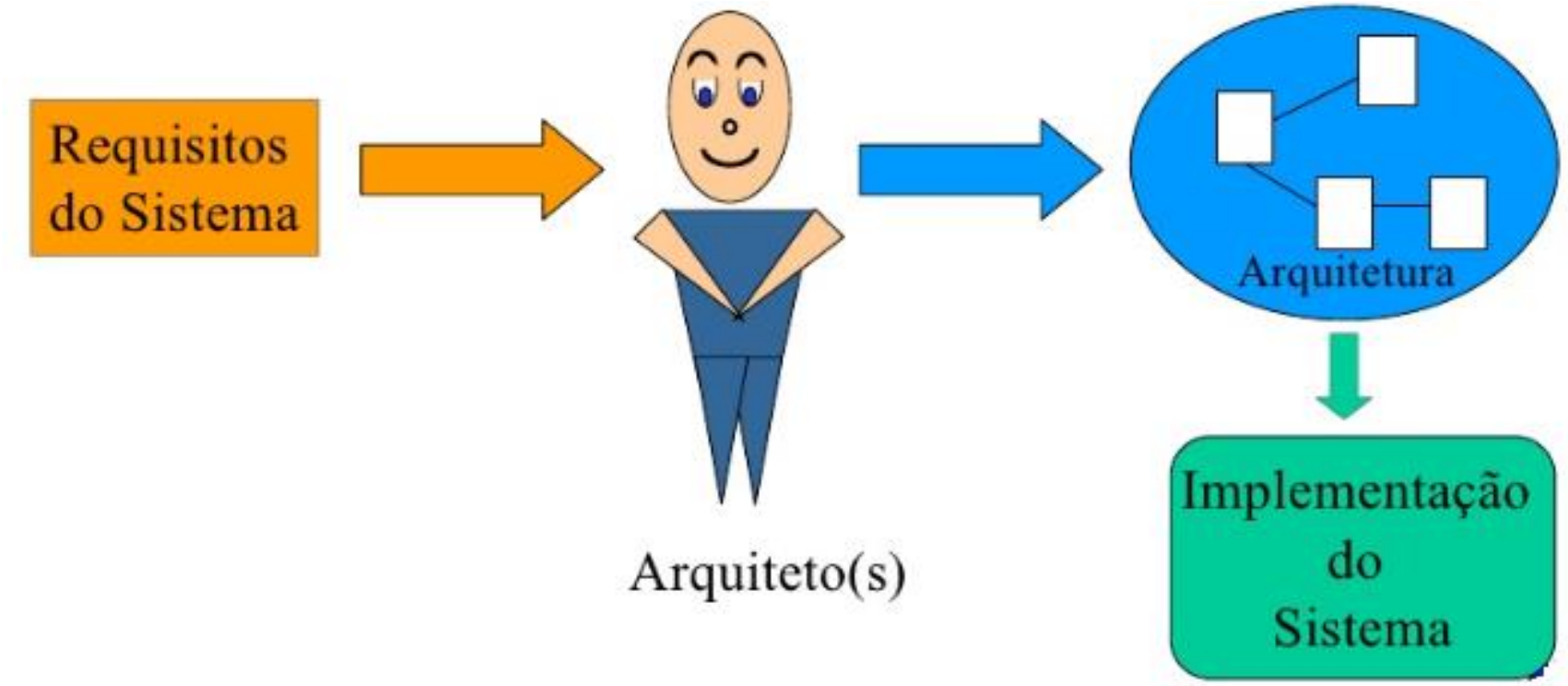
Arq. de Software – Contexto de Atuação

- <u>Na equipa de desenvolvimento:</u>
  - Liderança técnica para garantir que a arquitetura definida é devidamente seguida;
  - Auxilio, acompanhamento e revisão de desenhos produzidos pela equipa;
  - *Mentoring* e organizador de formações e *workshops*;
  - Envolvimento nos testes de Sistema e de Integração;
  - Desenvolver código, se necessário.

Arq. de Software – Contexto de Atuação

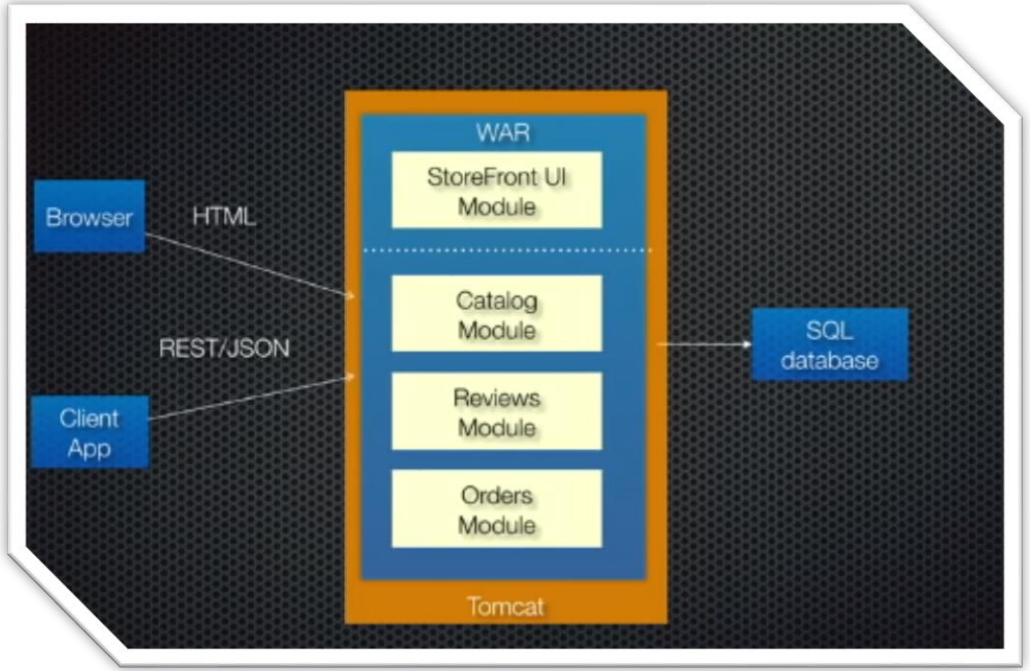
- *Stakeholders*
  - Identificar, envolver e balancear as necessidades;
  - Alinhar as expectativas do cliente com o objetivo do projeto;
    - Utilizador final -> Funções corretas, usabilidade;
    - Administrador -> Ferramentas de monotorização;
    - Marketing -> Conjunto de *features – Time to Market;*
    - Cliente -> Preço, *features*;
    - *Developers* -> Requisitos claros, bom desenho técnico;
    - Gestor -> Uso produtivo de recursos, prazo, custo;
    - Suporte -> Documentação clara.

Arq. de Software – Contexto de Atuação



Tipos de Arq. de *Software*

- ***Monolítica***
  - Simples para…
    - ✓ Desenvolver;
    - ✓ Testar;
    - ✓ *Deploy*;
    - ✓ Escalar.
  - Não são ideais para *Continuous Delivery*;
  - Não são ágeis;
  - Não é autónoma.



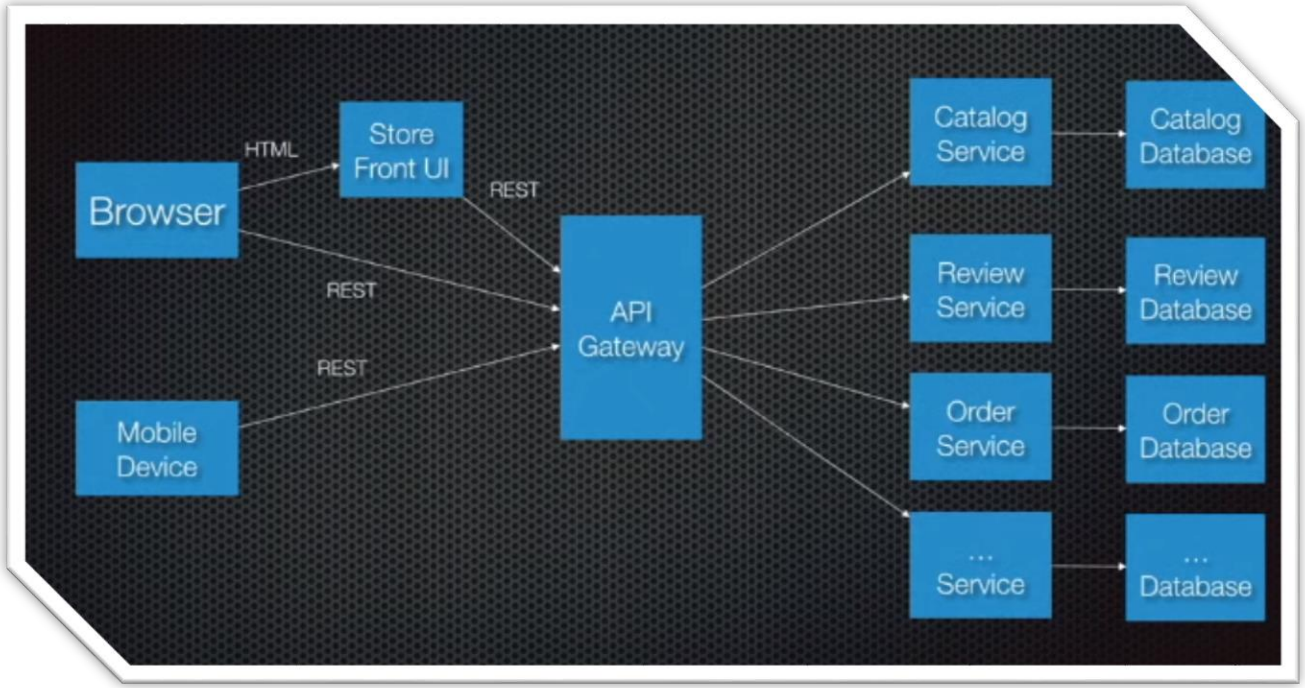
Tipos de Arq. de *Software*

Mas as aplicações bem sucedidas continuam a crescer…

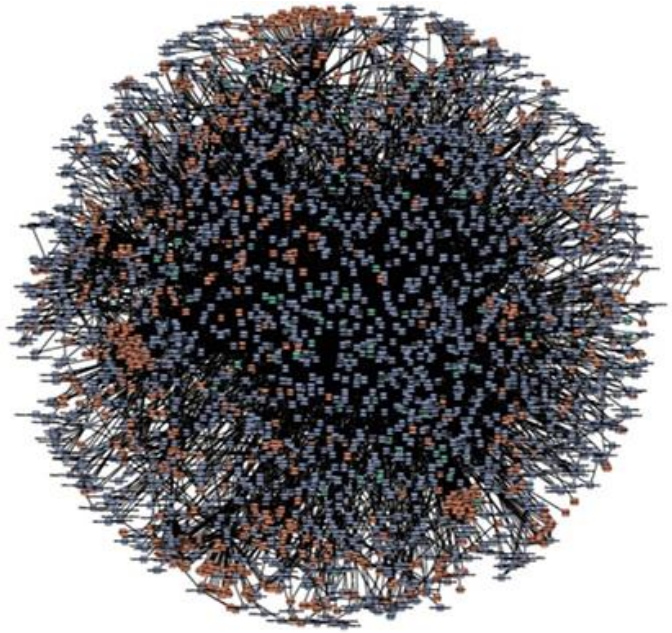
Tipos de Arq. de *Software*

- ***Micro-Services***
  - Escaláveis;
  - Autónomas;
  - Estruturadas;
  - Ideais para *Continuous Delivery*.

  - *<u>Drawbacks</u>*
    - São arquiteturas complexas devido a formarem um Sistema distribuído;
    - É difícil desenvolver um sistema de comunicação entre processos;
    - Gestão de transações;
    - <u>Cuidado com as dependências</u>.



Tipos de Arq. de *Software*

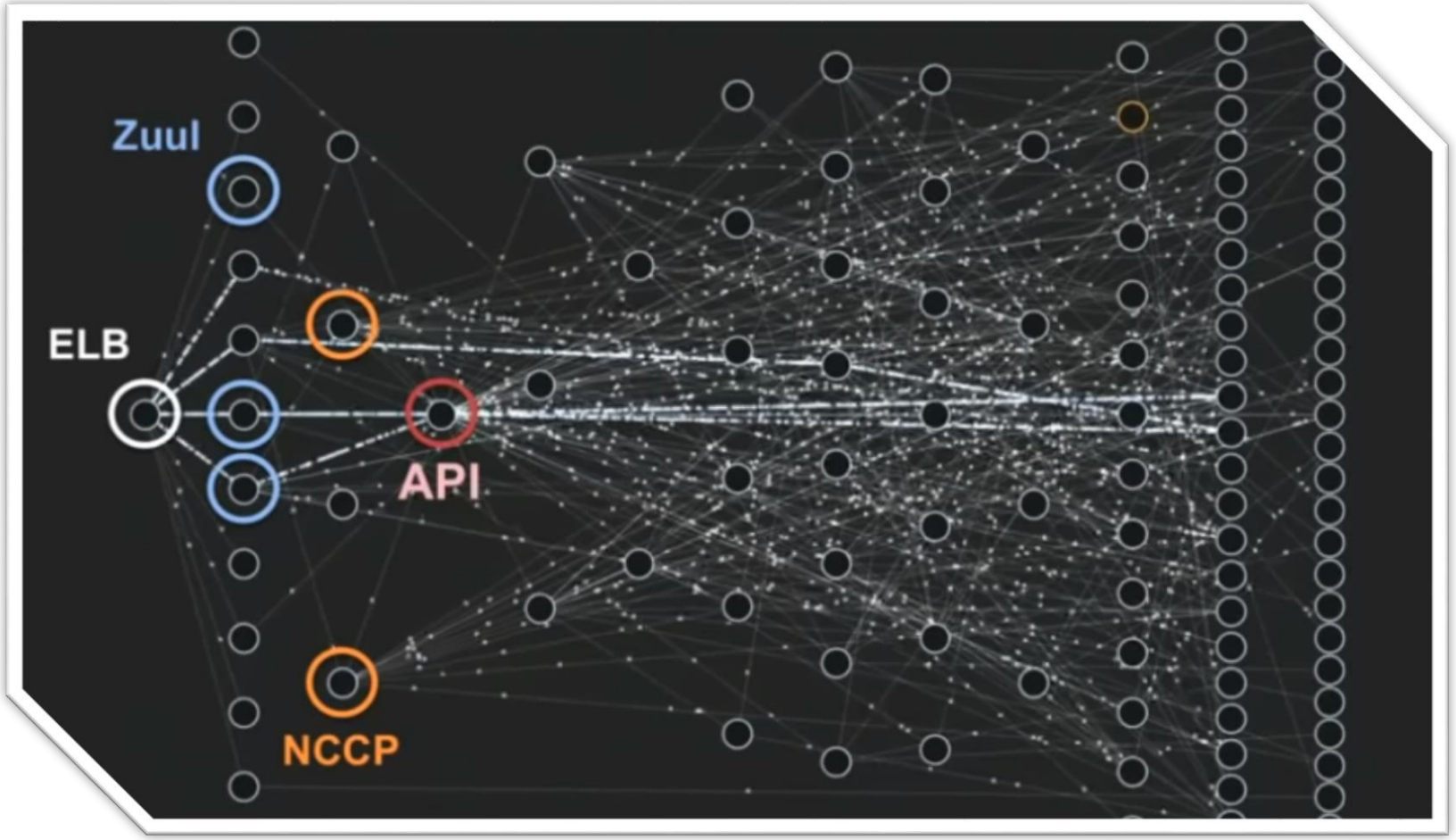


amazon.com

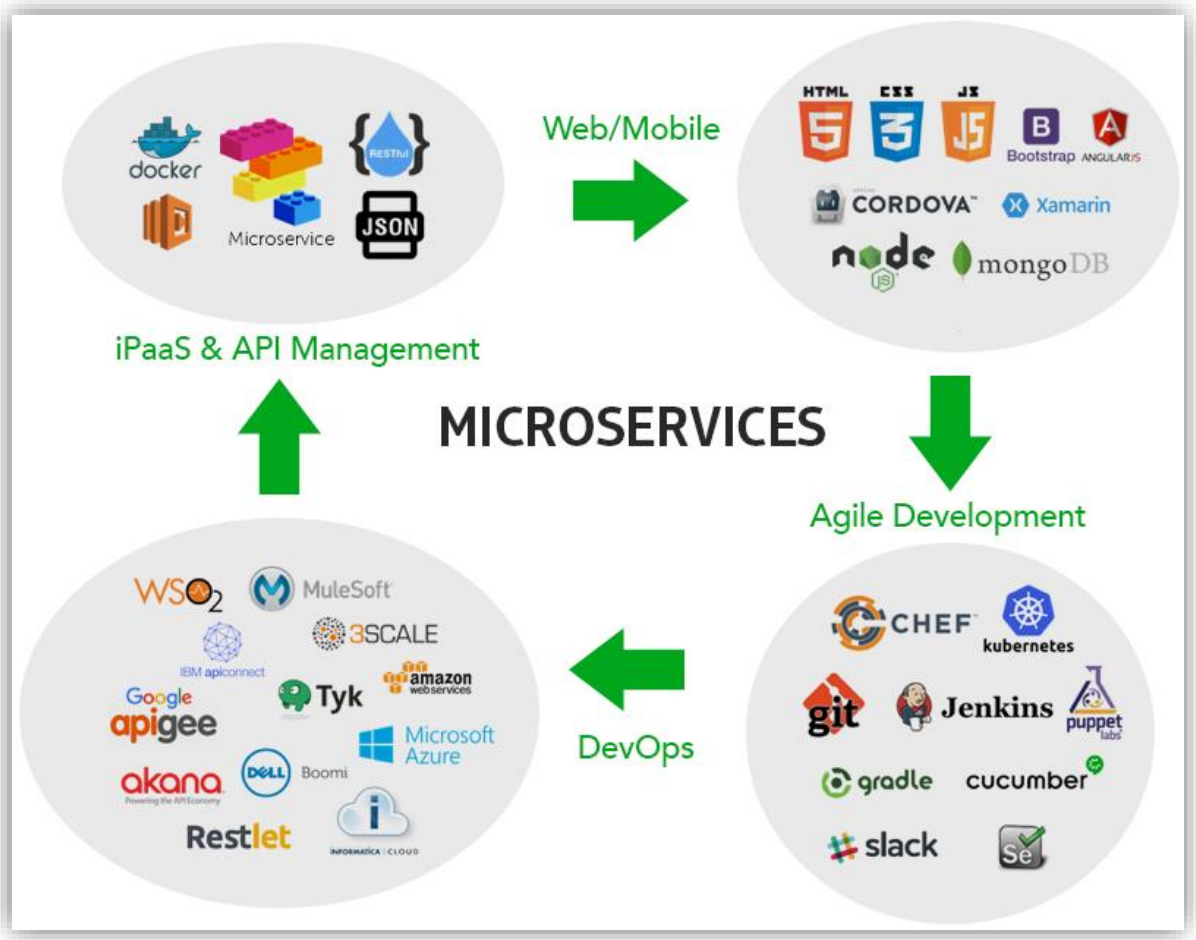
NETFLIX

"Death star diagram"

Tipos de Arq. de *Software*

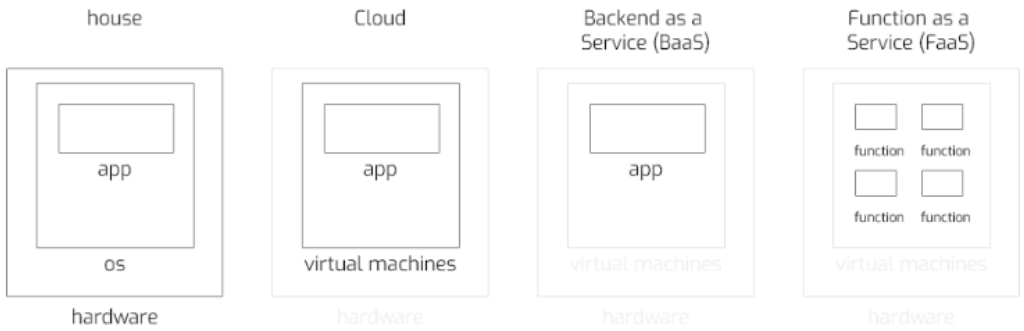


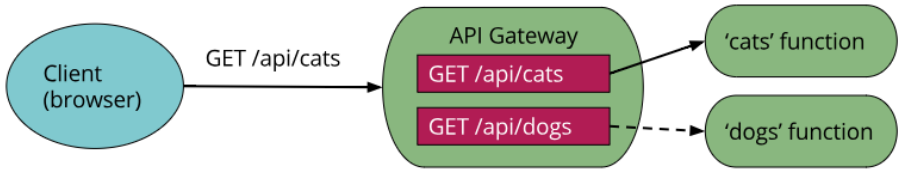
Tipos de Arq. de *Software*

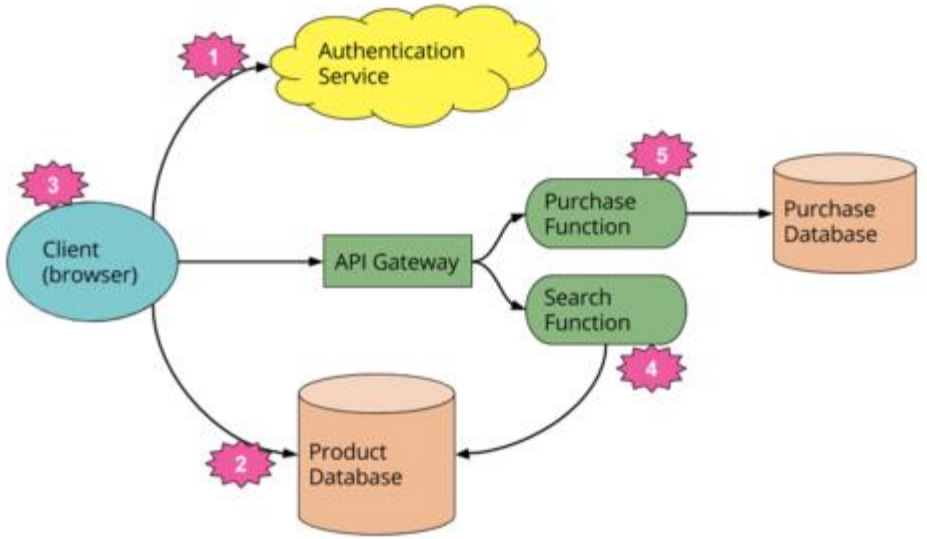


Tipos de Arq. de *Software*

- ***Serverless***
  - As arquiteturas *Serverless* referem-se a aplicações que dependem significativamente de serviços de terceiros (conhecido como *Back-end* como Serviço ou "BaaS") ou na execução de código em *containers* temporários (Função como Serviço ou "FaaS")



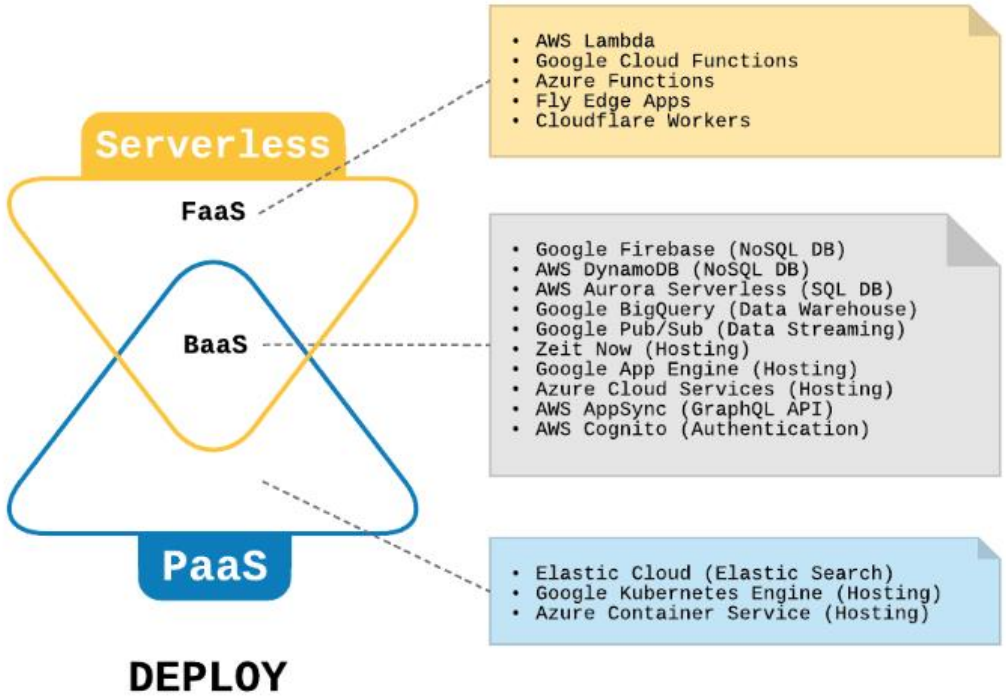




https://martinfowler.com/articles/serverless.html

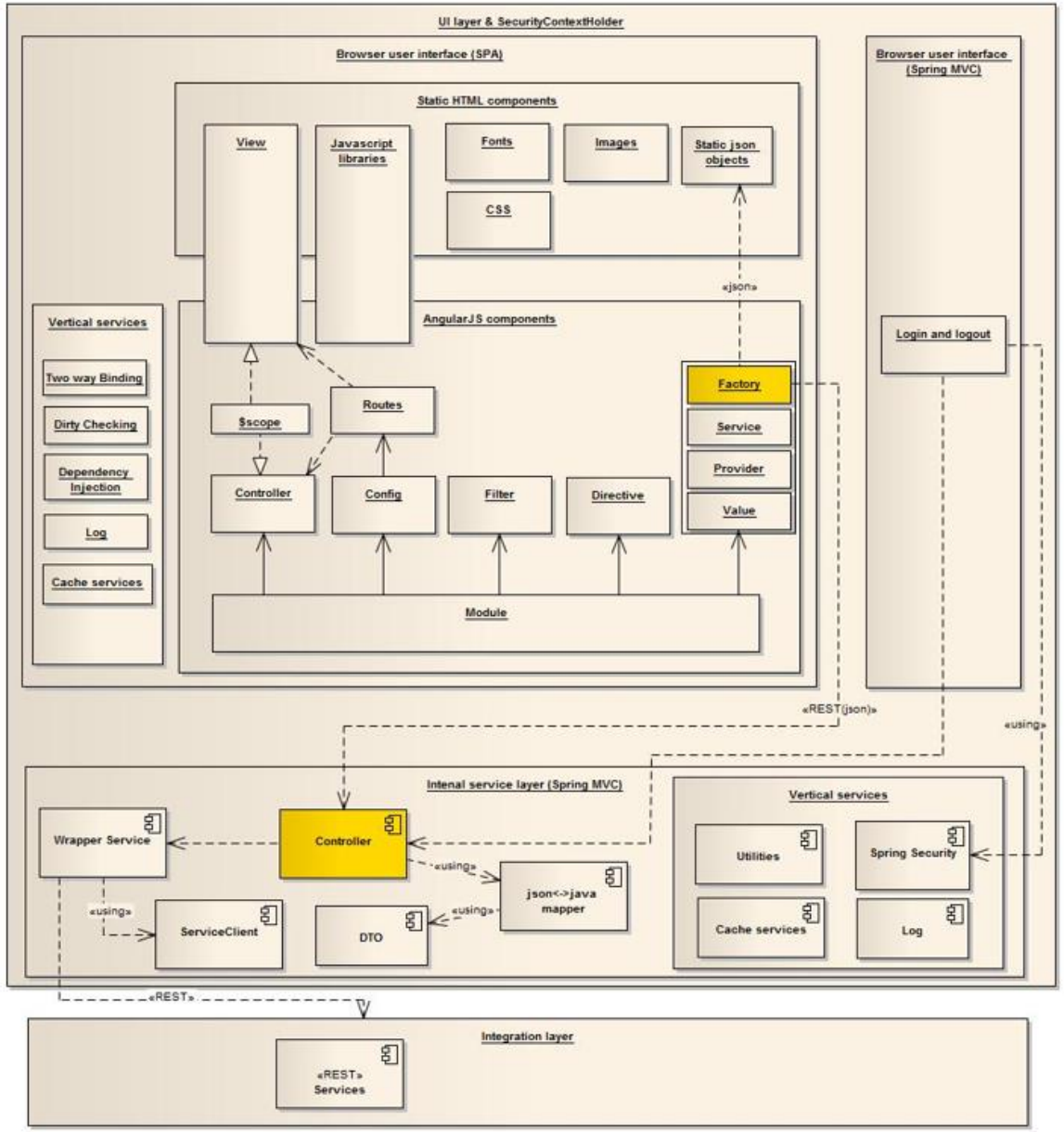
Tipos de Arq. de *Software*

- ***Backend as a Service ("BaaS")***
  - Com a evolução dos *cloud providers*, surgiu a oferta de serviços de uma infraestrutura de *back-end* já construída e distribuída, conhecido como "BaaS". Desta forma, os *developers* podem-se concentrar em implementar as regras de negócio definidas nos requisitos, abstraindo a necessidade de pensar e implementar uma infraestrutura.
- ***Function as a Service ("FaaS")***
  - Com este trabalho periférico mitigado, os *developers* passam a ter mais tempo para se dedicarem há suas funções primárias, entregando muito mais *Software* no mesmo período de tempo.



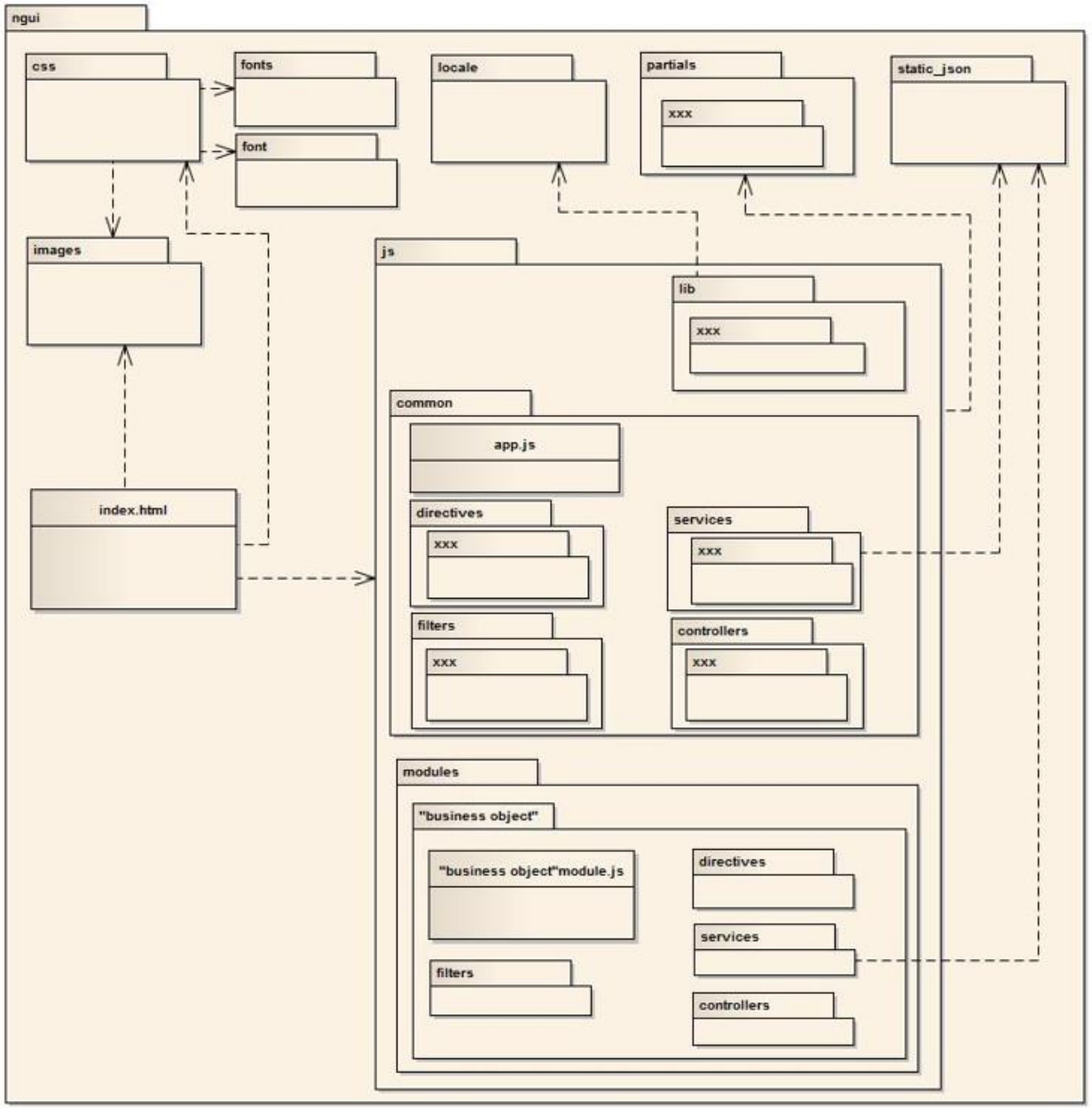
Tipos de Arq. de *Software*

- ***Serverless***
  - *Auto scaling*
    - A escalabilidade é realizada de forma automática: quanto mais consumo houver nas funções despoletadas, mais disponibilidade o *cloud provider* dará. Deste modo, podemos dizer que a escalabilidade é quase infinita.
  - Redução de custo
    - Cada vez que uma função é executada, é cobrado um valor apenas pelo processamento consumido. Sendo assim, não é cobrado tempo de inatividade, que é um problema que ocorre hoje em dia nas máquinas virtuais.



Exemplo

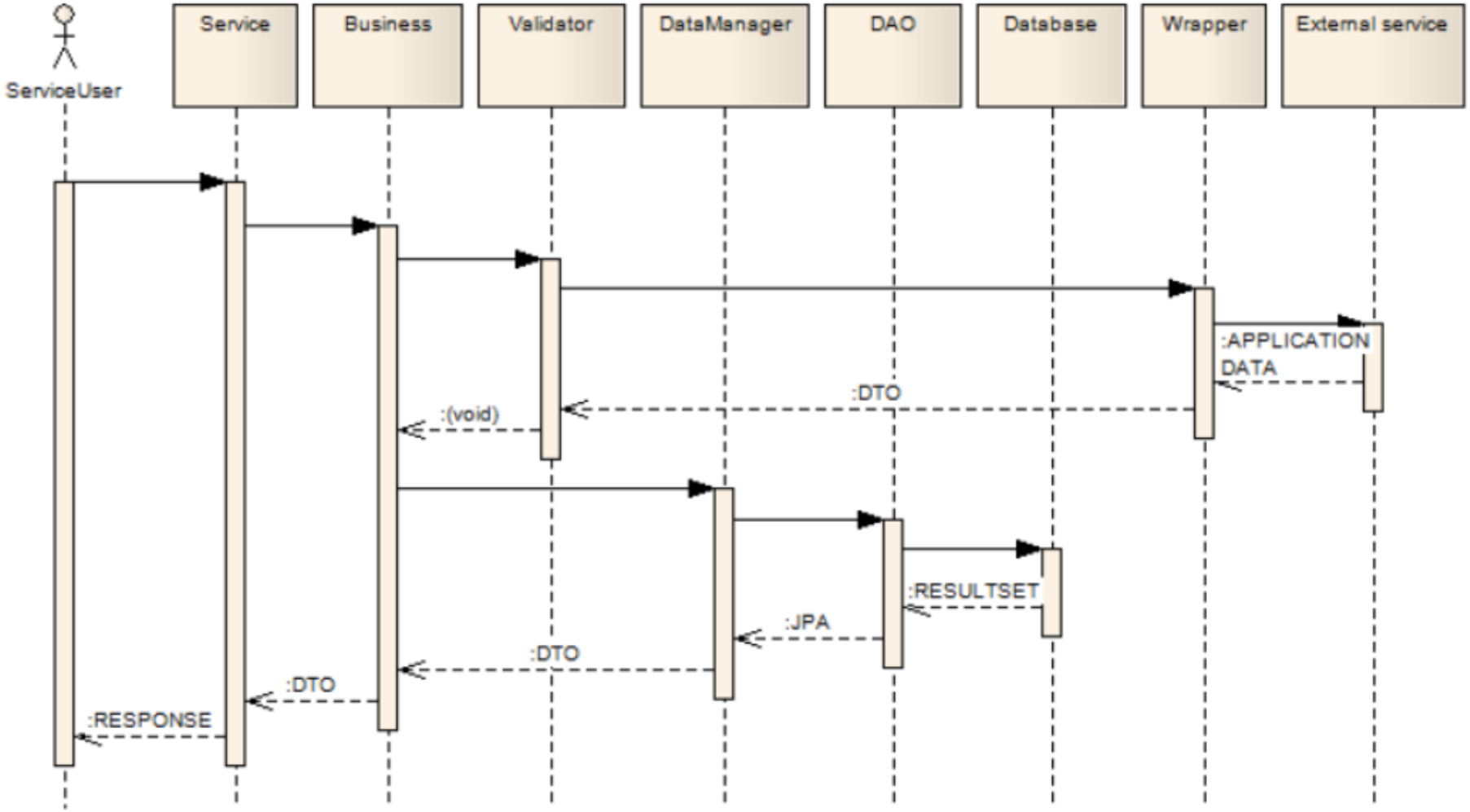
- Arquitetura de uma aplicação Web



Exemplo

- Arquitetura / Estruturação do *Front-end* de um *Software*

Prof. Pedro Perdigão
p5617@ulusofona.pt



Exemplo

- Arquitetura / Estruturação do *Back-end* de um *Software*

Dúvidas?

Prof. Pedro Perdigão
p5617@ulusofona.pt