



Embarcadero Conference

MICROSSERVIÇO, ESCALABILIDADE E RESILIÊNCIA - #COMOFAS?

Mario Guedes

Embarcadero

Conference



Mario?

41 anos, filho da Valdete e marido da Tamires
Artesão de software desde 1999
Delphi, Python, Lua, JavaScript, noSQL ...
Vivência em grandes soluções para Contact Center
CTO na Sofie Tecnologia
Embarcadero MVP desde 2016
Em todas as redes: @jmarioguedes
mario.guedes@arrayof.com.br





Indo direto ao ponto

- Já estamos trabalhando multicamada, certo?
- Temos planos mirabolantes e disruptivos que vai mudar o mundo, certo?
- Estamos migrando o software para a tal da nuvem, certo?
- *Mobile first*, certo? Data Science é palavra de ordem, certo?
- Mas e o seu backend? É **escalável** e **resiliente**?
 - **Escalável:** *Capacidade em atender de 10 a 10.000 usuários sem reescrita de código*
 - **Resiliente:** *Capacidade de tolerar e se recuperar rapidamente de rupturas*

#comofas?

RESTful

Microserviço

API Gateway

Mensageria

noSQL

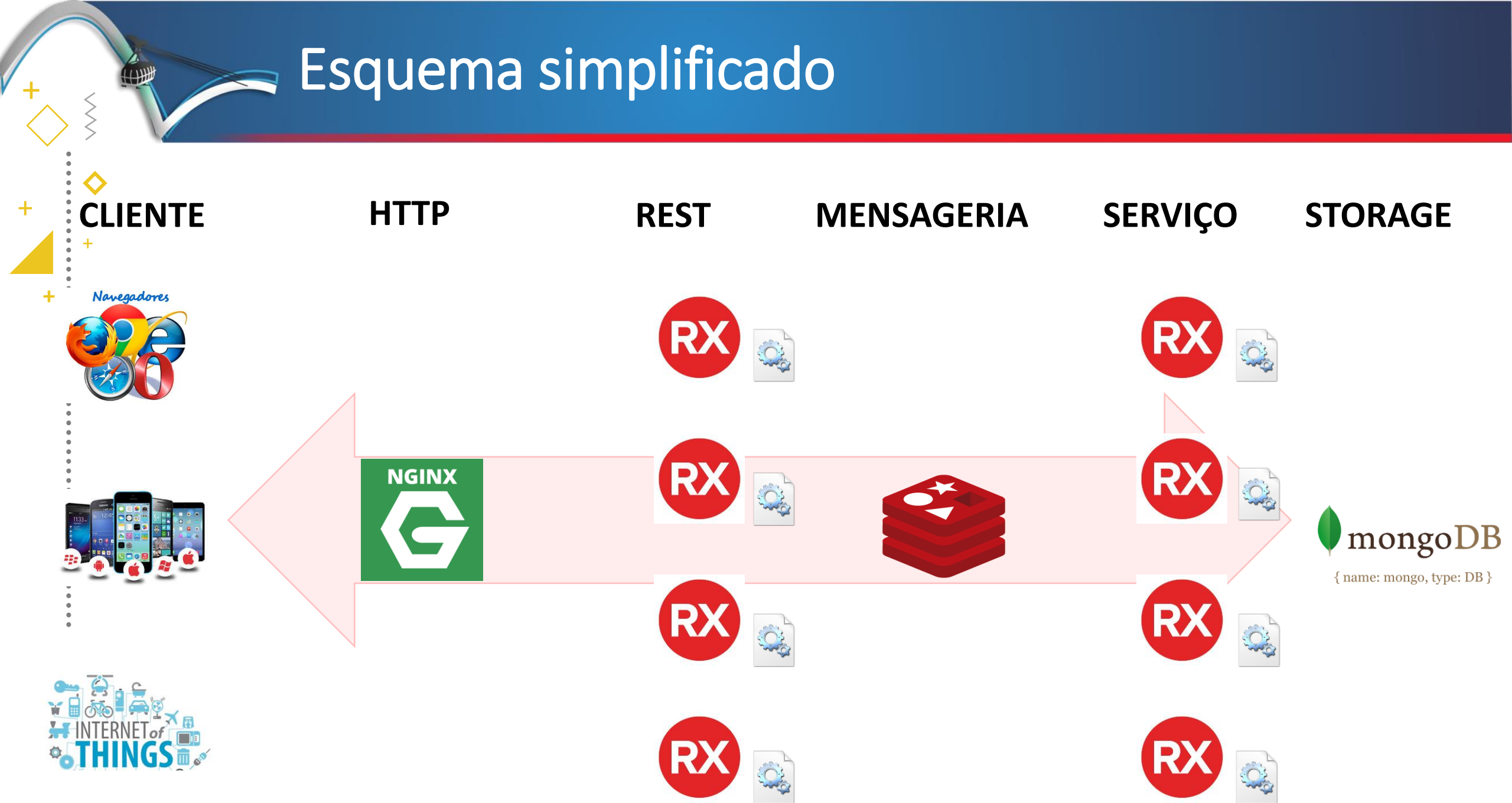
Cacheamento
lado servidor

Linux

Contêiner

Servless

Esquema simplificado

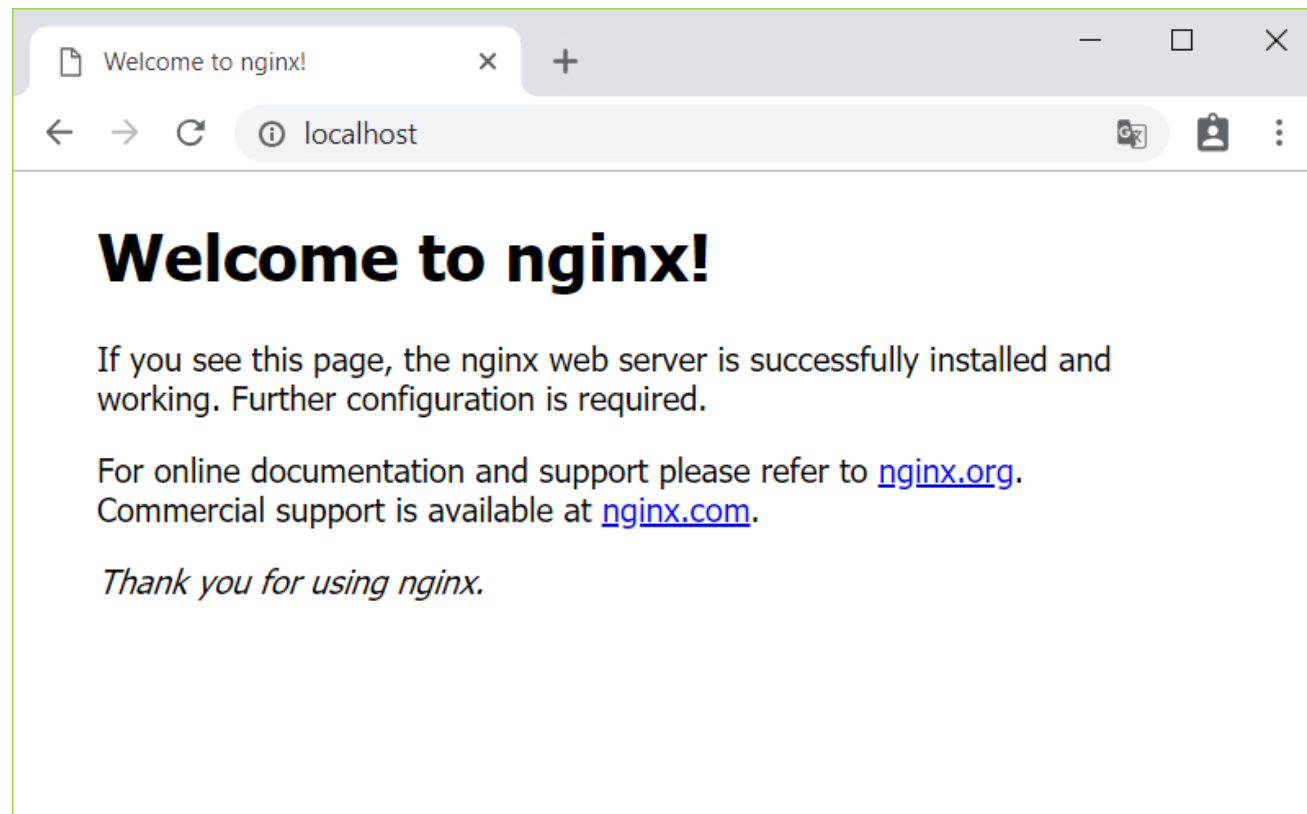




Vamos para cima?

“Engine X”

- NGINX
- Criado em 2004 pelo russo Igor Sysoe
- Alta performance com relativo baixo consumo de recursos computacionais.
- Servidor web
- Cache
- Streaming
- Balanceador de carga
- Proxy reverso por design






Instalação e configuração

- Em produção, prefira servidores Linux
- As rotas e demais configurações são feitas nos arquivos *.conf
- A configuração se dá por **contextos** e **diretivas**.
- A versão *free* pode ser baixada em <http://nginx.org>
- Instruções para Windows em:
<http://nginx.org/en/docs/windows.html>

Configuração inicial – arquivos estáticos



```
1  events {}
2
3  http {
4
5      include mime.types; # Referência a uma arquivo externo que, no caso, possui uma extensa lista de mime-types
6
7      server {
8
9          listen 80; # Porta HTTP
10         server_name 127.0.0.1; # Domínio associado à hospedagem virtual e.g. *.arrayof.com.br
11
12         root c:/site_arrayOF/; # Diretório raiz dos arquivos referentes à hospedagem virtual
13
14     } # Configurando uma hospedagem virtual
15
16 }
17
```

Um passo à frente – tratando URIs

```
root c:/site_array07/; # Diretório raiz dos arquivos referentes a hospedagem virtual

location /qualquer {
    return 200 'Qualquer URI que comece com "/qualquer"';
} # Bloco de localização específica à uma URI - COMBINAÇÃO PELO PREFIXO

location = /exatamente {
    return 200 'Somente a URI "/exatamente"';
} # Bloco de localização específica à uma URI - COMBINAÇÃO EXATA

location ~ /fruta_(uva|maca|laranja) {
    return 200 'URI que combine com a RegEx /fruta_(uva|maca|laranja)';
} # Bloco de localização específica à uma URI - COMBINAÇÃO COM REGEX

location ^~ /fruta_uva {
    return 200 'URI que combine com prefixo de forma preferencial (em detrimento a RegEx)';
} # Bloco de localização específica à uma URI - COMBINAÇÃO PREFERENCIAL PELO PREFIXO

location = /inspect {
```



Lidando com variáveis

<http://nginx.org/en/docs/varindex.html>

```
location = /inspect {  
    if ($arg_apikey != 12345678) {  
        return 401 'Chave de API incorreta';  
    }  
  
    set $EC2018 'HOJE *NAO* EH O DIA';  
    if ($date_local ~ '23-out-2018') {  
        set $EC2018 'HOJE *EH* O GRANDE DIA';  
    }  
  
    return 200 "Quando: $date_local - $EC2018\n\nHost: $host\nRecurso: $uri\nQuery string: $args\n\nNome: $arg_nome";  
} # Bloco de localização exemplificando uso de variáveis  
  
location = /google {
```

Chegando ao DataSnap!

```
} # Bloco de localização exemplificando uso de variáveis  
  
location = /google {  
    return 307 'http://www.google.com.br';  
} # Bloco de localização exemplificando um REDIRECIONAMENTO  
  
rewrite ^/api/v\d/(.*) /datasnap/rest/$1; # Exemplo de uma reescrita de URI  
  
location /datasnap/rest/ {  
    add_header Proxy 'NGINX - EC2018 - RESPONSE'; # Adicionando cabeçalho na resposta ao cliente  
    proxy_set_header Proxy 'NGINX - EC2018 - REQUEST'; # Adicionando cabeçalho na requisição ao DataSnap  
    proxy_pass http://127.0.0.1:8080; # URL do servidor DataSnap que será concatenado com a URI  
}  
# Bloco de localização ENCAMINHANDO para um processo DataSnap  
# Configurando uma hospedagem virtual
```


Agora, balanceando a carga ...

```
http {  
  
    include mime.types; # Referência a uma arquivo ex  
  
    upstream datasnap {  
        server 127.0.0.1:8080;  
        server 127.0.0.1:8081;  
        server 127.0.0.1:8082;  
    } # Serviços DataSnap "espalhados" pela rede  
  
    server {  
  
        listen 80; # Porta HTTP  
        server_name 127.0.0.1; # Domínio associado à
```

```
location /datasnap/rest/ {  
  
    add_header Proxy 'NGINX - EC2018 - RESPONSE'; # Adicionando cabeçalho na resposta ao cliente  
  
    proxy_set_header Proxy 'NGINX - EC2018 - REQUEST'; # Adicionando cabeçalho na requisição ao DataSnap  
  
    # proxy_pass http://127.0.0.1:8080; # URL do servidor DataSnap que será concatenado com a URI  
    proxy_pass http://datasnap; # Referência à um contexto UpStream, provendo balanceamento e redundância  
  
} # Bloco de localização ENCAMINHANDO para um processo DataSnap
```



Like a boss

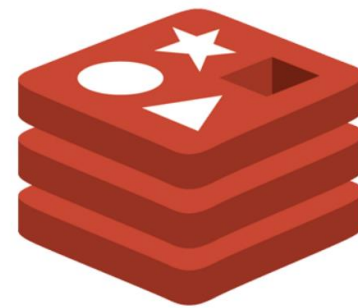
```
upstream datasnap {  
    # ip_hash; # Cria afinade entre cliente e servidor  
    least_conn; # Dá preferência para o serviço com o menor número de conexões  
  
    server 127.0.0.1:8080;  
    server 127.0.0.1:8081;  
    server 127.0.0.1:8082;  
} # Serviços DataSnap "espalhados" pela rede
```

```
upstream datasnap {  
    ip_hash; # Cria afinade entre cliente e servidor  
  
    server 127.0.0.1:8080;  
    server 127.0.0.1:8081;  
    server 127.0.0.1:8082;  
} # Serviços DataSnap "espalhados" pela rede
```

```
upstream datasnap {  
    # ip_hash; # Cria afinade entre cliente e servidor  
    # least_conn; # Dá preferência para o serviço com o menor número de conexões  
  
    server 127.0.0.1:8080;  
    server 127.0.0.1:8081;  
    server 127.0.0.1:8082;  
  
    server 127.0.0.1:9090 backup; # Só será acionado se os primários estiverem "down"  
} # Serviços DataSnap "espalhados" pela rede
```

REDIS

- REmote DIctionary Server
– Servidor de Dicionário Remoto
- Criado por Salvatore Sanfiippo
- Código livre
- Feito em C “puro”
- Aceita scripts Lua



redis



Utilizando como mensageria

- O REDIS não é a melhor opção para mensageria mas também não é a pior: É suficientemente bom em relação à simplicidade.
- Dependendo dos seus requisitos considere a adoção do **RabbitMQ**
- Nesta apresentação darei foco no **ENFILEIRAMENTO** e **PUB/SUB**
- *Porém o REDIS é ótimo para controle de sessão de usuários bem como cacheamento lado servidor*



Show me the code

- Demonstração:
 - Publicação e assinatura
 - Enfileiramento
- Redis Client: <https://github.com/danieleteti/delphiredisclient>



<http://arrayof.com.br>

Se cadastrando hoje, **23/10**, você terá acesso gratuito à primeira turma de um dos seguintes cursos:

- REDIS com Delphi
- MongoDB com Delphi
- RabbitMQ com Delphi – Mensageria
- Telecom com FreeSWITCH
- Sistema de auto atendimento com FreeSWITCH
- Construção de Aplicativo Mobile



OBRIGADO



arrayOF

Treinamento e Assessoria

<http://arrayof.com.br>

<http://fb.me/arrayOFAssessoria>

mario.guedes@arrayof.com.br



(11) 9-9217-0907

Embarcadero

Conference



Embarcadero Conference