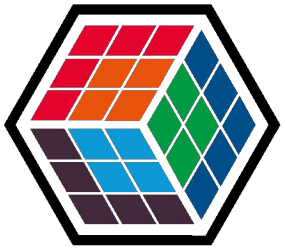


THE DEVELOPER'S CONFERENCE

Arquitetura Java – Pastoreando gatos

Rodrigo Stefani Domingues
Principal Architect na CI&T

$~: whoami

THE DEVELOPER'S CONFERENCE

Desenvolvedor a 12 anos
Java a 9 anos
Na CI&T a 6 anos
Microserviços a 3 anos
Nerd

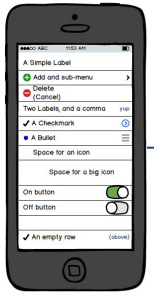
rodrigosd@ciandt.com

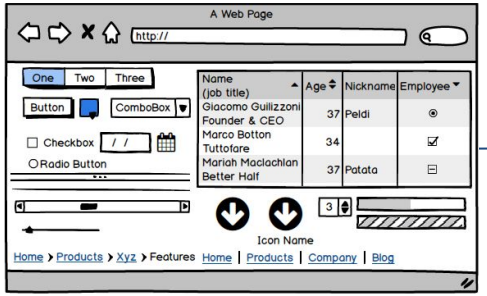
https://github.com/rsdomingues

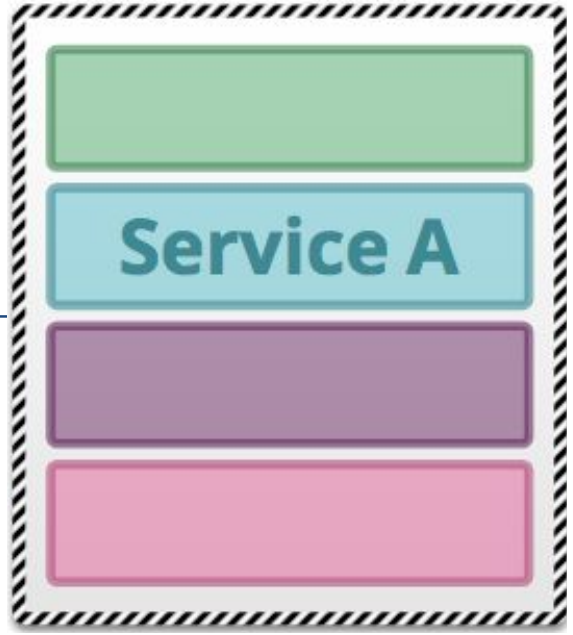
"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



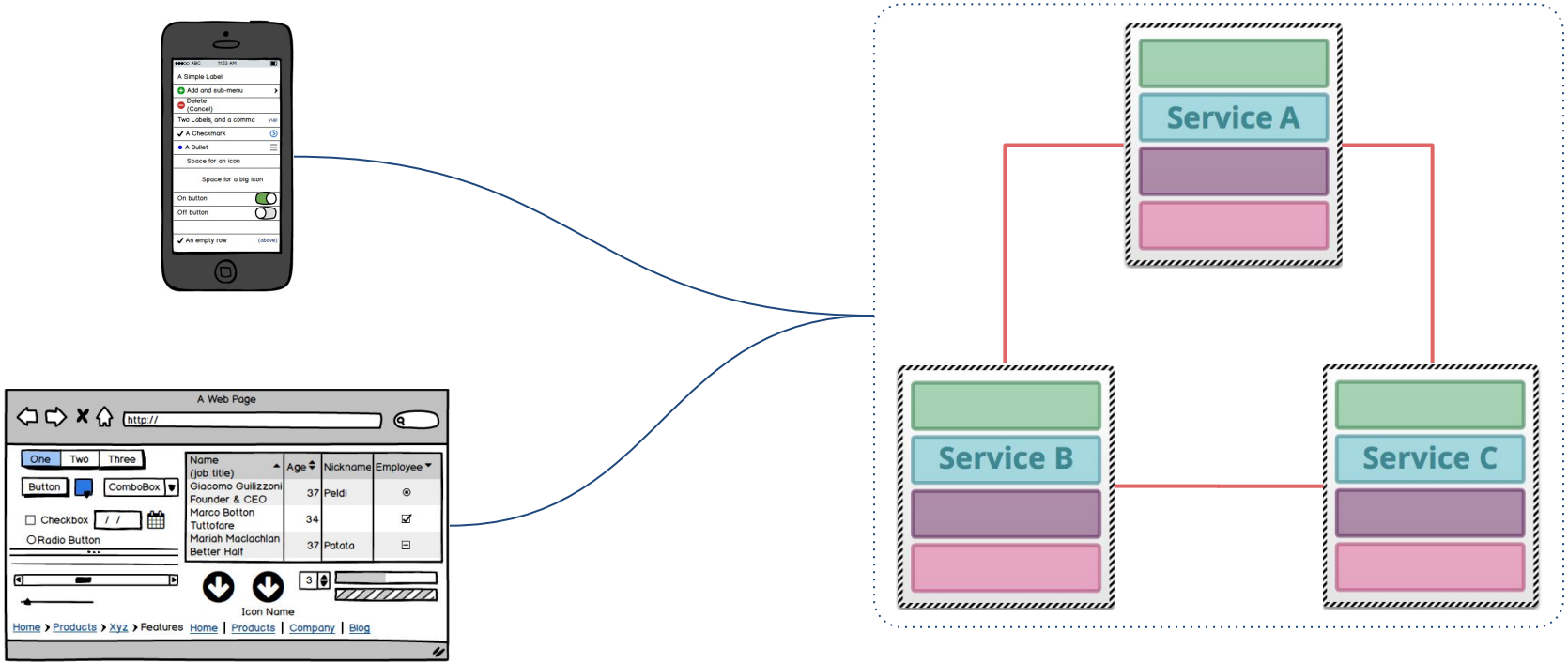




"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

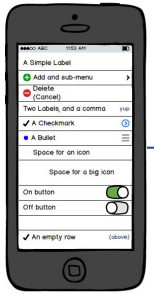


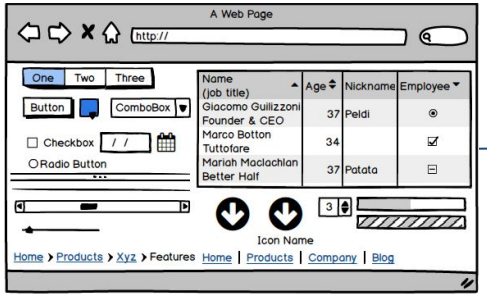

MICROSERVICES
MICROSERVICES EVERYWHERE

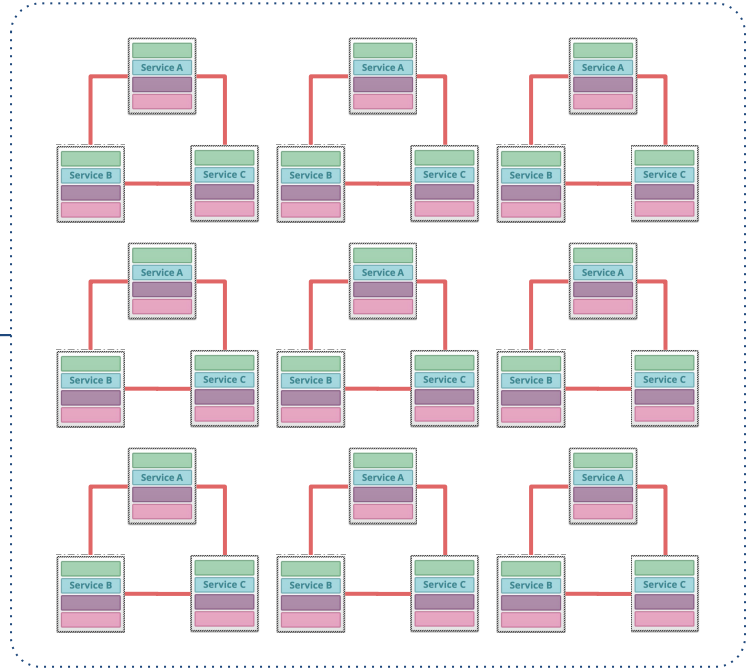
"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



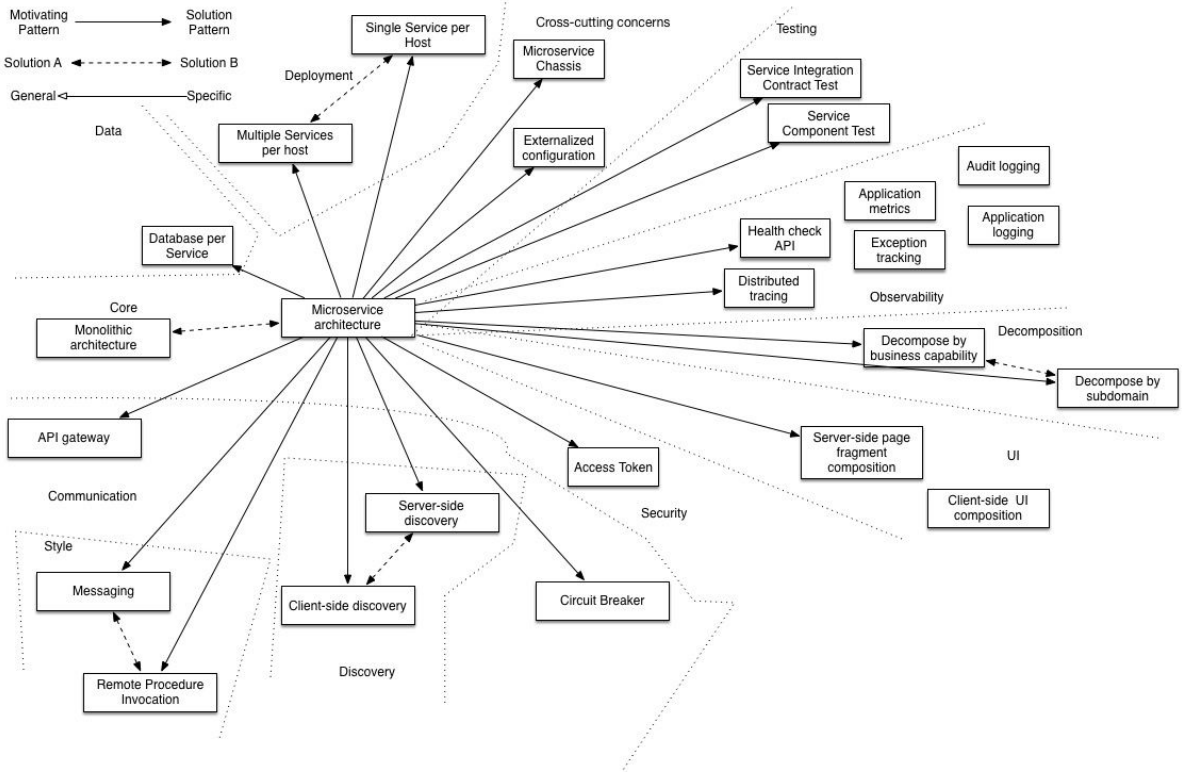




"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

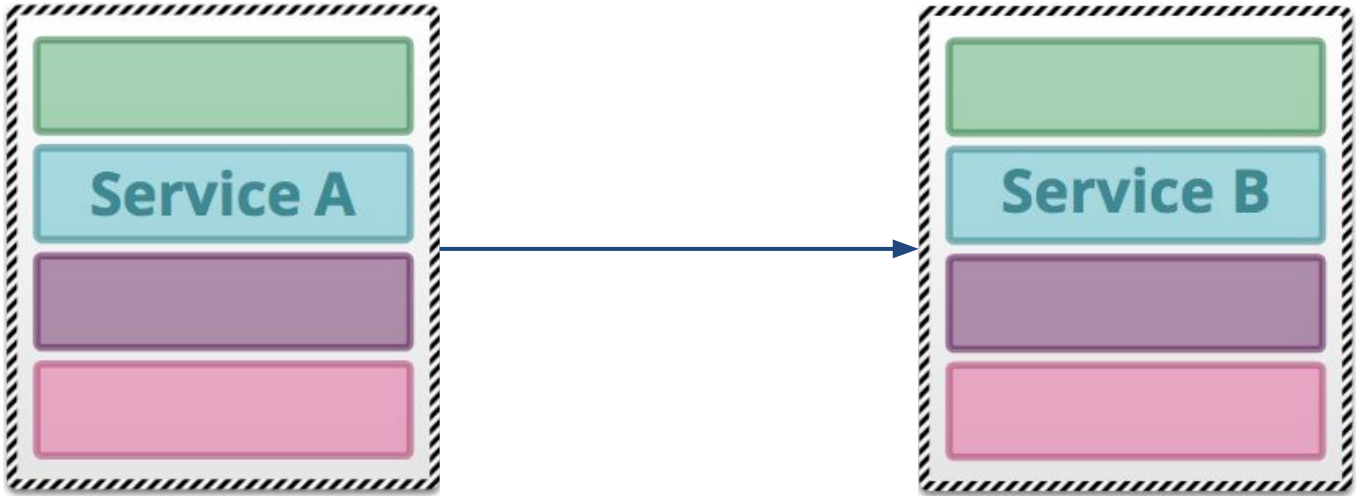
THE DEVELOPER'S CONFERENCE



Comunicação

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.



"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

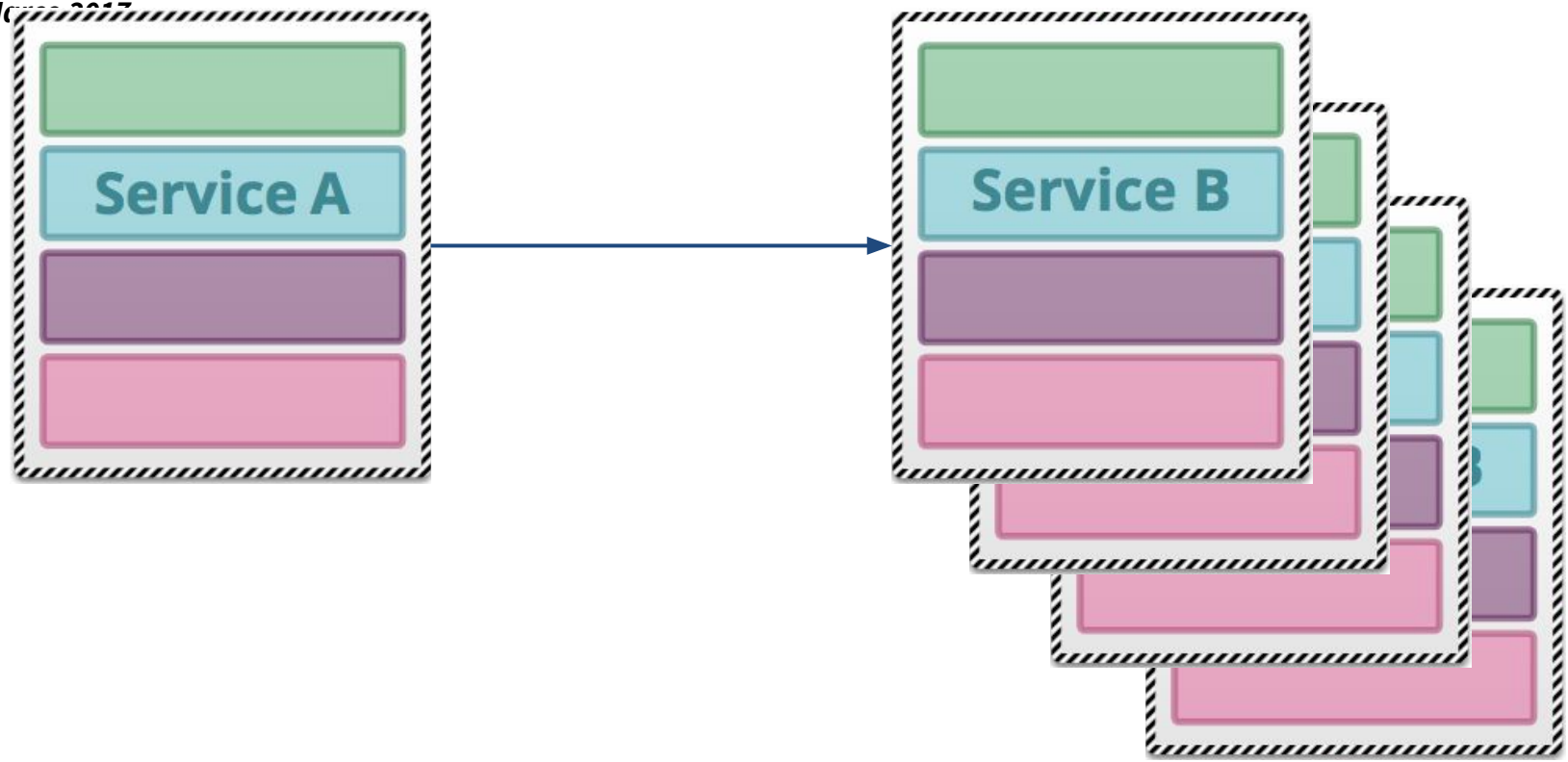
PaymentClient.java Raw

```java
package com.tdcsample.checkout.gateway.feign;

import com.tdcsample.checkout.gateway.feign.jsons.PaymentRequest;
import com.tdcsample.checkout.gateway.feign.jsons.PaymentResponse;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

@FeignClient(value = "payment", fallback = PaymentFallback.class)
public interface PaymentClient {

    @RequestMapping(method = RequestMethod.POST, value = "/api/v1/payment")
    PaymentResponse process(@RequestBody PaymentRequest request);
}
```

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, M[illegible]

THE DEVELOPER'S CONFERENCE



"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

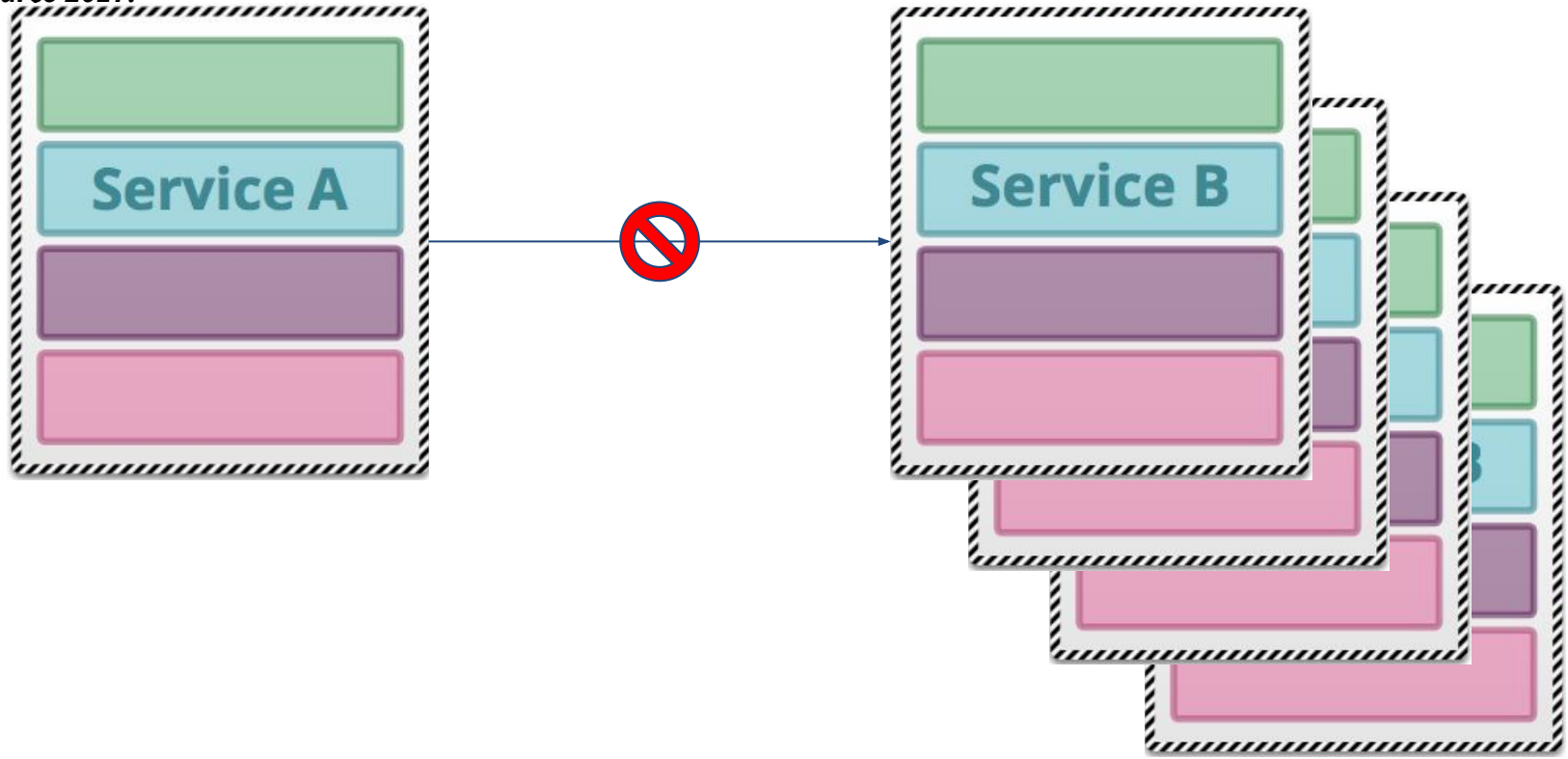
application.yml Raw

```
[...]

payment:
  ribbon:
    ReadTimeout: 30000
    ConnectTimeout: 30000
    listOfServers: http://localhost:8081,http://localhost:8082,http://localhost:8083

[...]
```

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Marco 2017.

THE DEVELOPER'S CONFERENCE



"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

THE DEVELOPER'S CONFERENCE

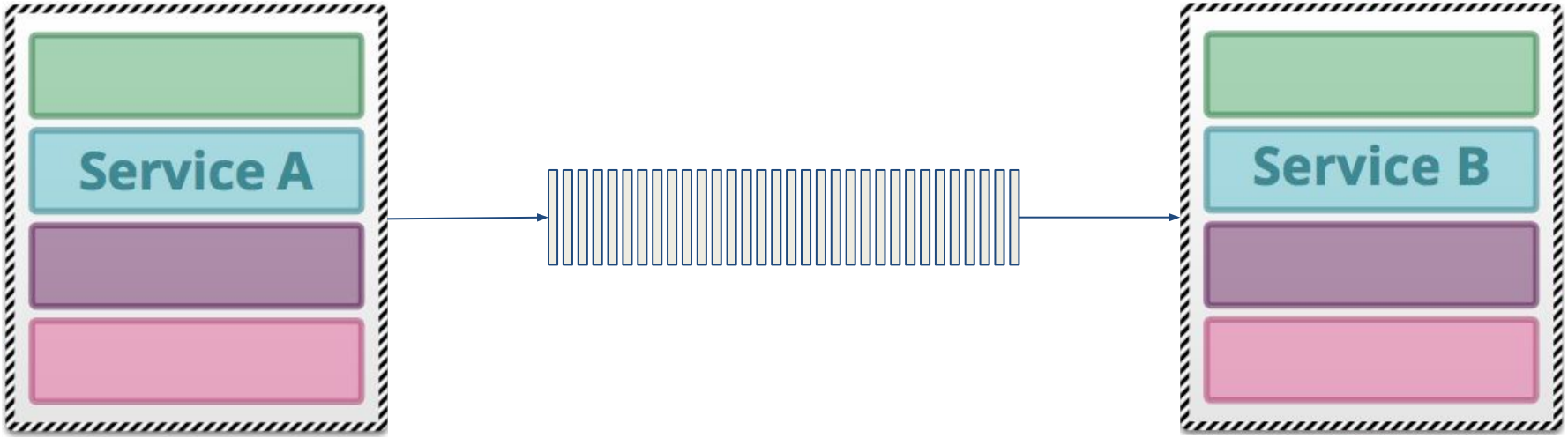
PaymentFallback.java Raw

```java
package com.tdcsample.checkout.gateway.feign;

import com.tdcsample.checkout.gateway.feign.jsons.PaymentRequest;
import com.tdcsample.checkout.gateway.feign.jsons.PaymentResponse;
import com.tdcsample.checkout.gateway.feign.jsons.PaymentStatus;
import org.springframework.stereotype.Component;

@Component
public class PaymentFallback implements PaymentClient {

    @Override
    public PaymentResponse process(PaymentRequest request){
        PaymentResponse response = new PaymentResponse();
        response.setStatus(PaymentStatus.NOT_SENT);
        response.setTransactionId("internal-12345");
        response.setDetails("");
        return response;
    }
}
```

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

MyApplication.java Raw

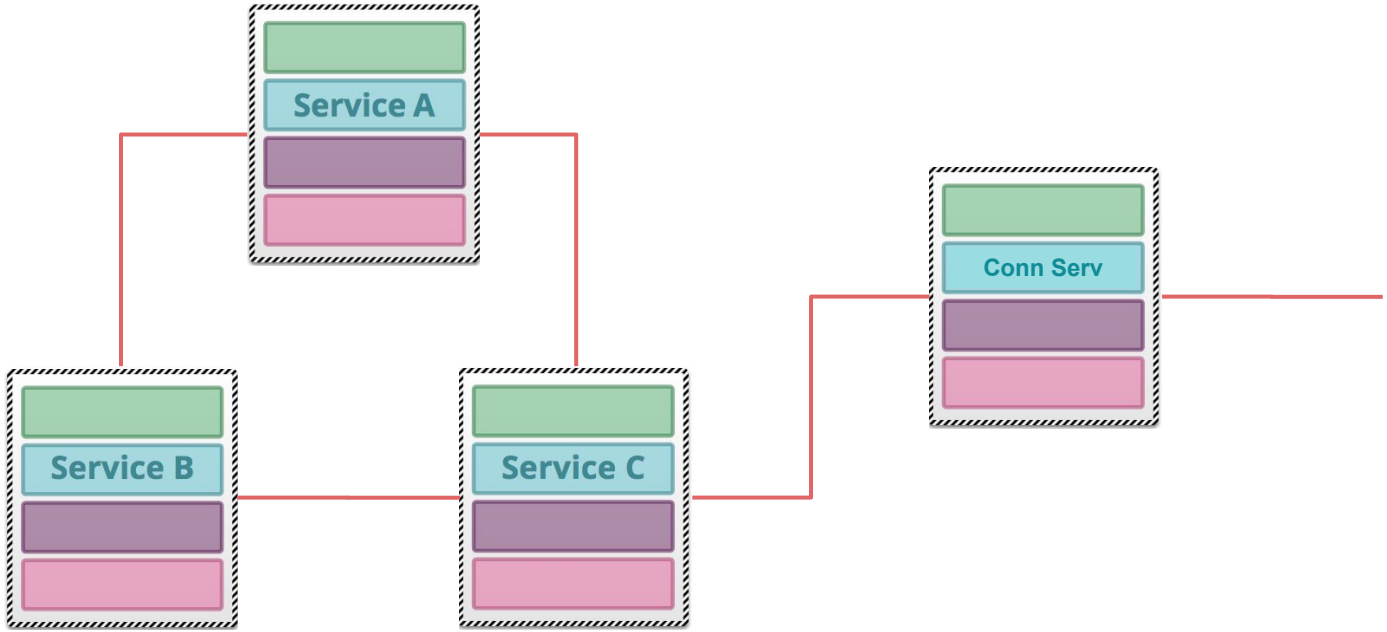
```java
@SpringBootApplication
@EnableBinding(Processor.class)
public class MyApplication {

	public static void main(String[] args) {
		SpringApplication.run(MyApplication.class, args);
	}

	@StreamListener(Processor.INPUT)
	@SendTo(Processor.OUTPUT)
	public String handle(String value) {
		System.out.println("Received: " + value);
		return value.toUpperCase();
	}
}
```

Resolvendo alguns problemas

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

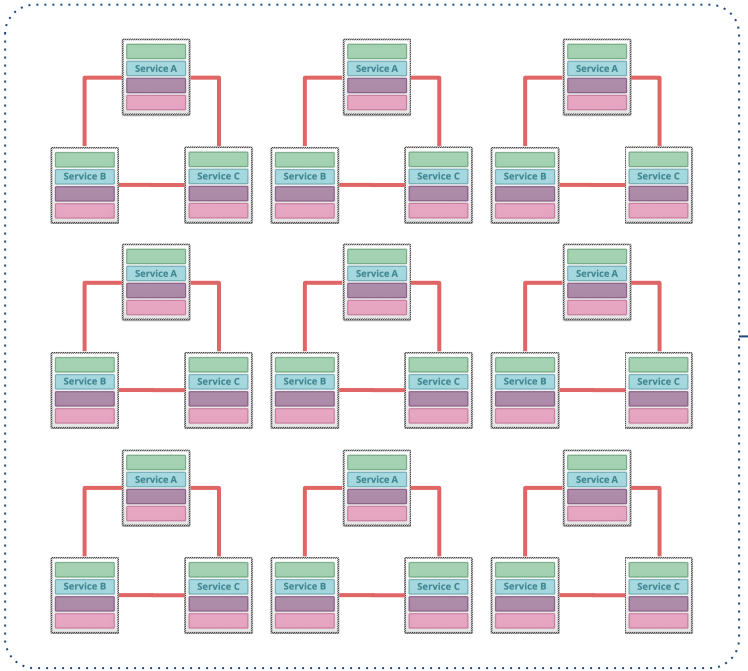
THE DEVELOPER'S CONFERENCE

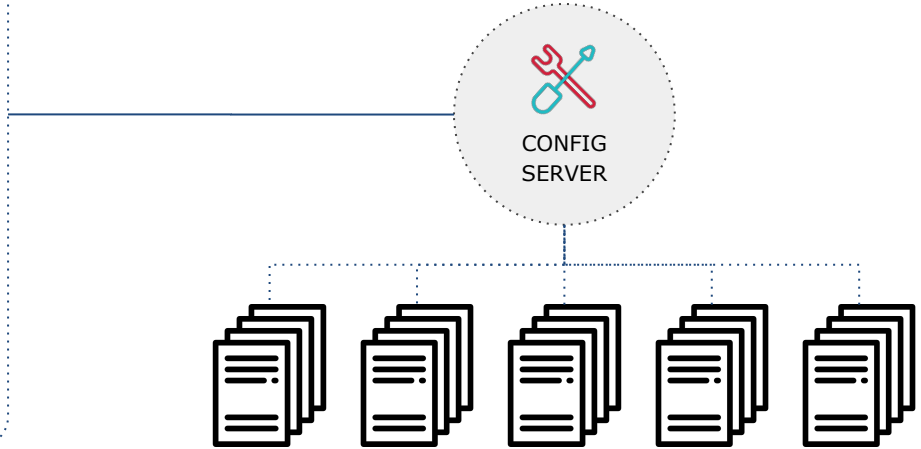


"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

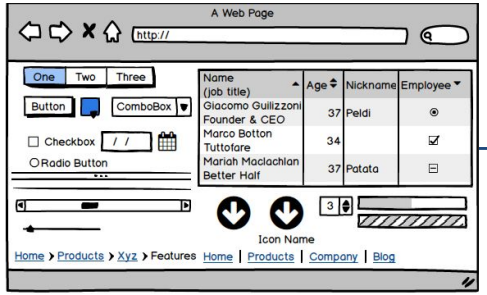


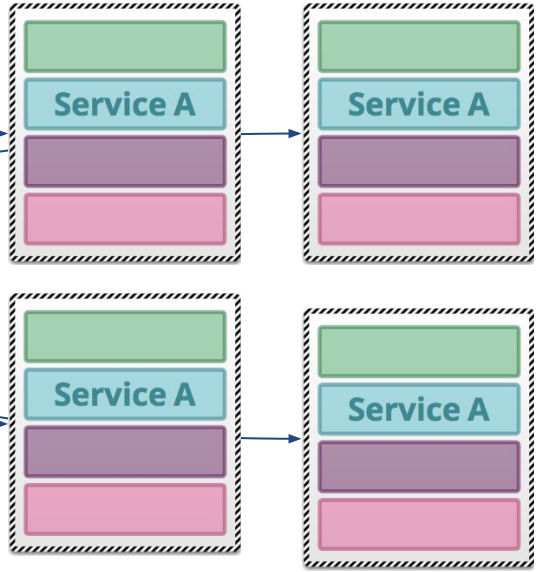


"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE





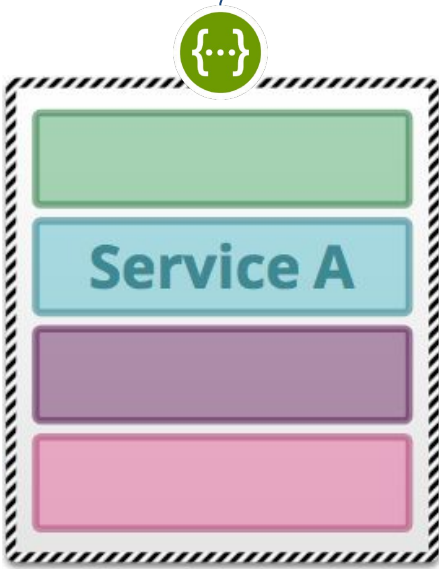
❌ `POST: {"items": [{...}],{...},{...},{...},{...}}`

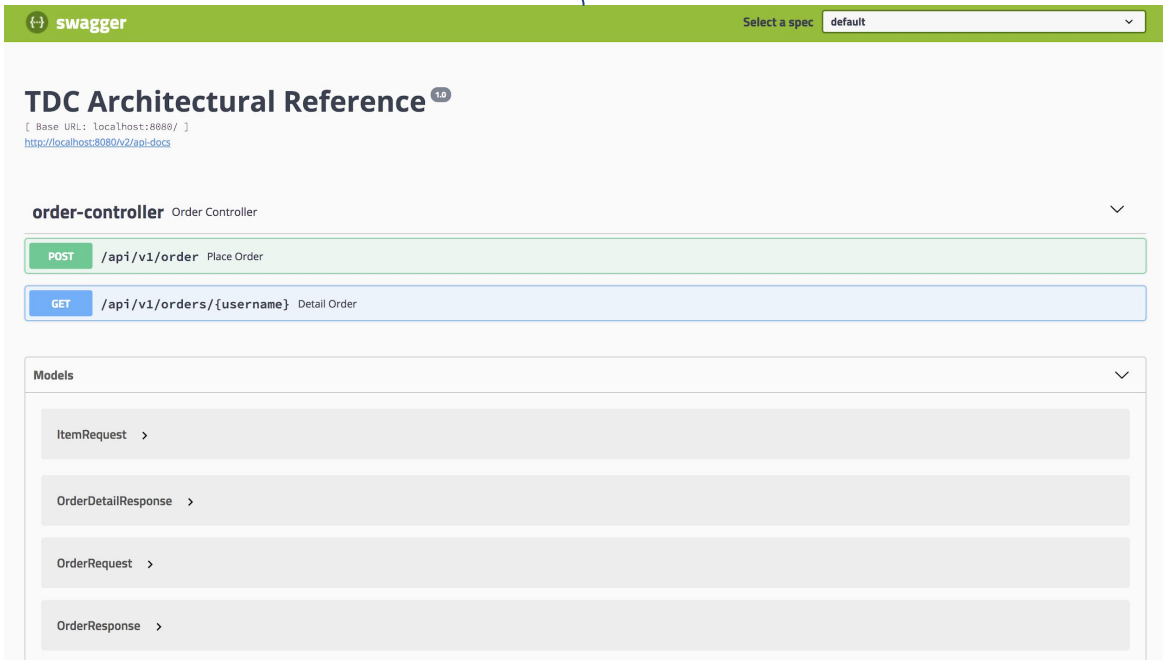
👌
```
POST: {"item": {...}}
POST: {"item": {...}}
POST: {"item": {...}}
POST: {"item": {...}}
```

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE





Any fool can write code that a computer can understand. Good programmers write code that humans can understand.

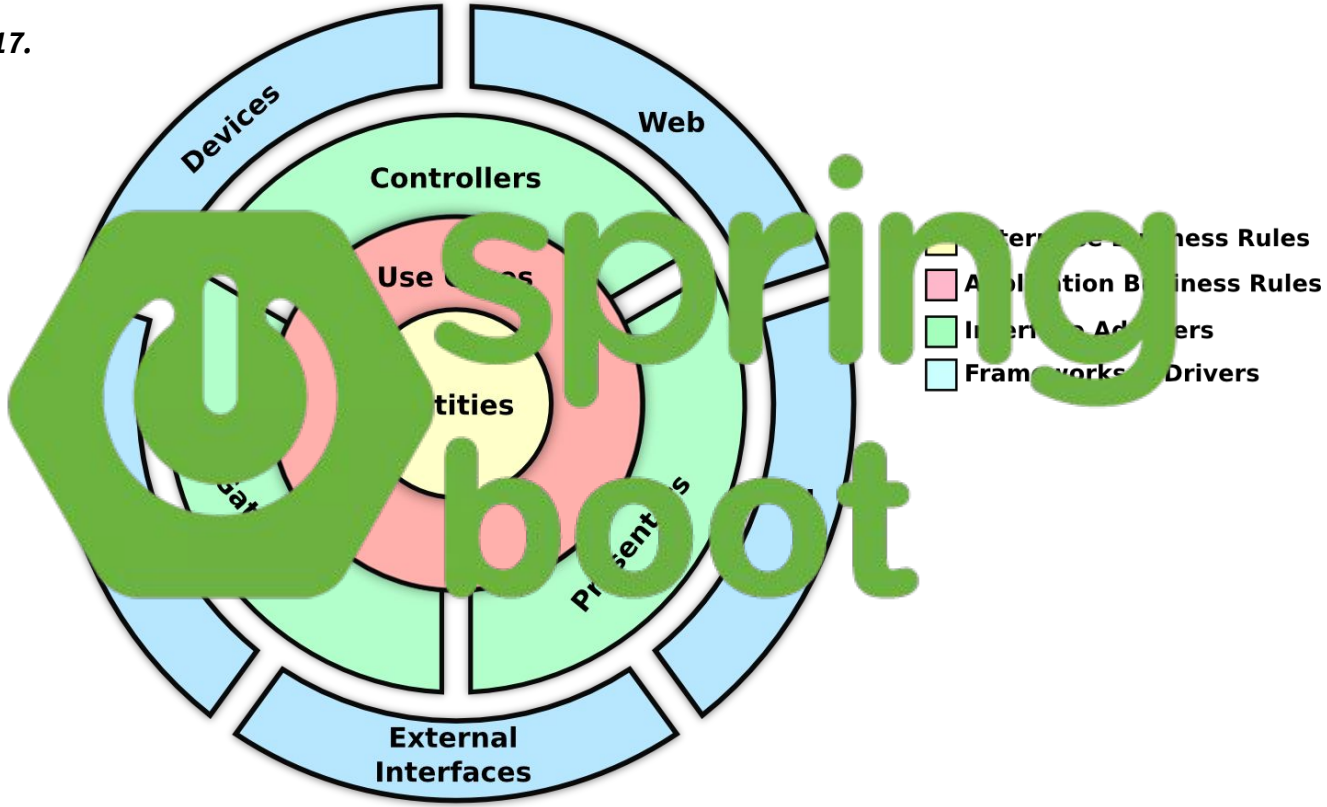
— Martin Fowler —

AZ QUOTES

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE

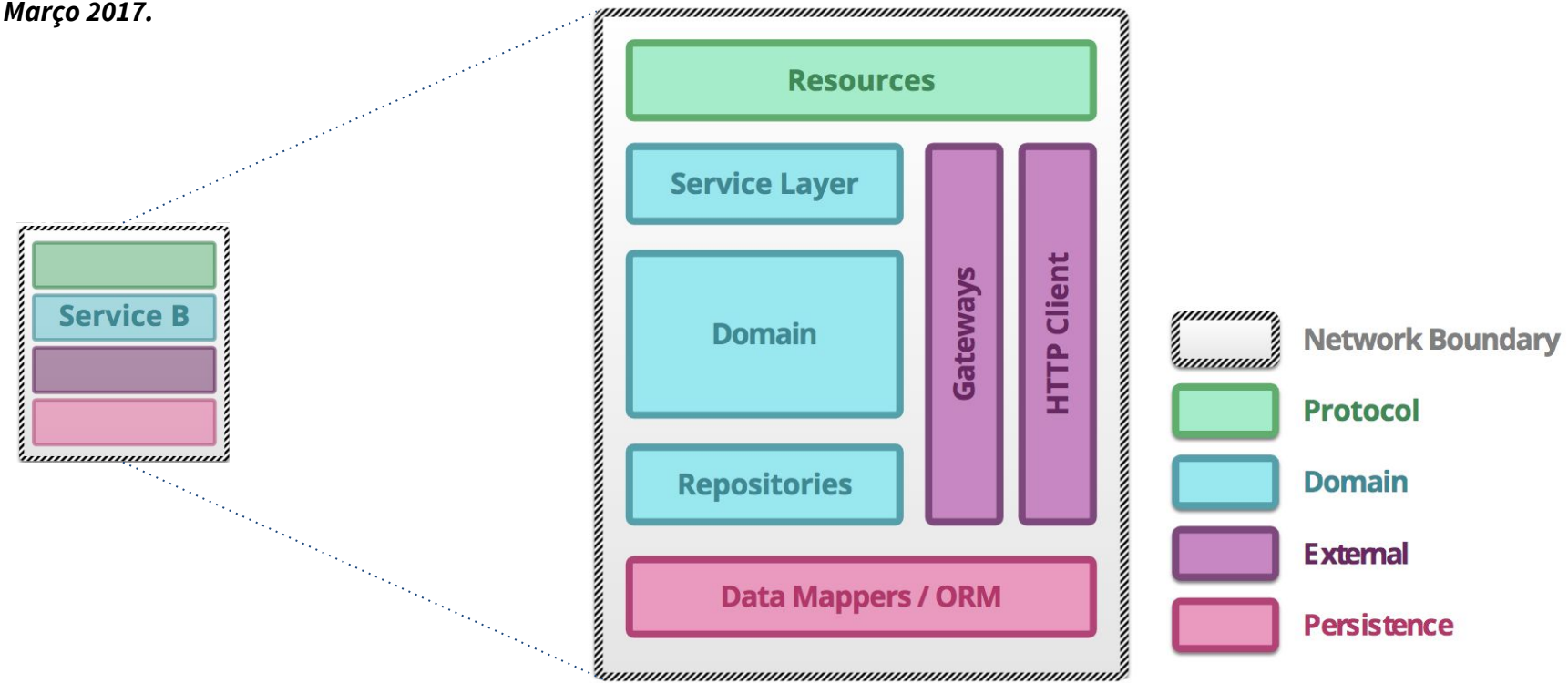


Estratégia de testes

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

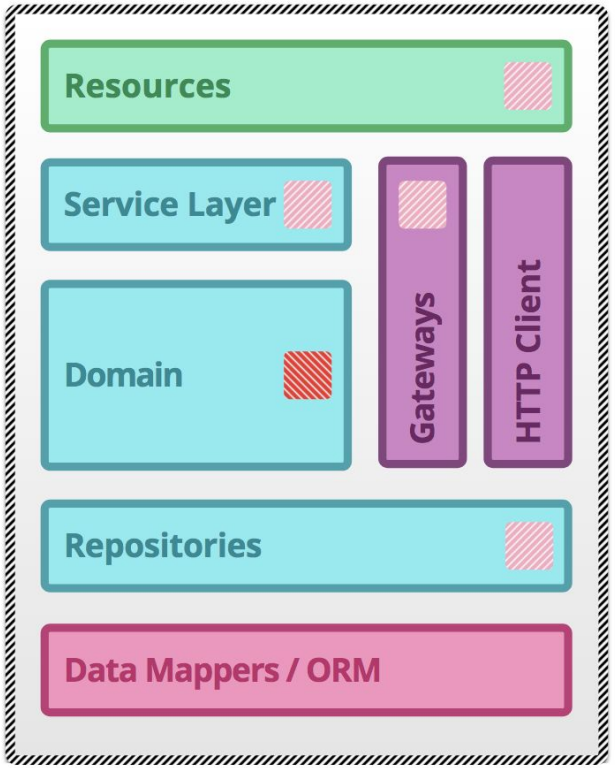
Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



"O **teste de unidade** exercita a menor parte do software testável na aplicação para determinar se ele se comporta conforme o esperado."

THE DEVELOPER'S CONFERENCE



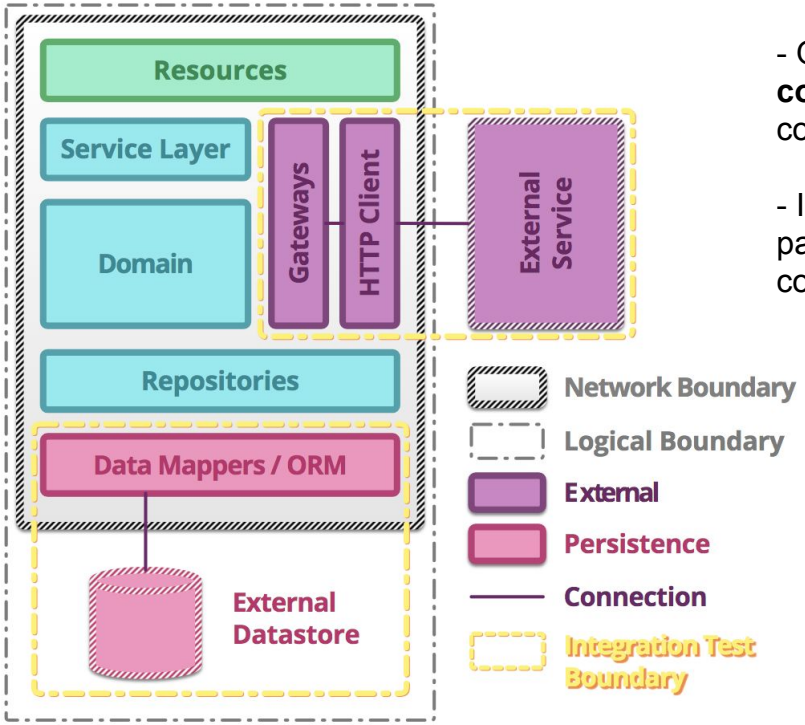
- A responsabilidade do teste unitário é de garantir a unidade (menor elemento), em nosso caso os métodos e funções, logo quaisquer dependências deles devem ser **mockados**.

- Cobertura de 100% do código → The broken window theory

- Cobrir 100% não significa cobrir todo código
    - Testes sociavel
    - Foco no código "util"

"O **teste de integração** verifica os caminhos de comunicação e as interações entre os componentes para detectar defeitos na interface."

THE DEVELOPER'S CONFERENCE



- Os testes de integração são responsáveis por garantir o **comportamento das integrações**, tanto no que se refere ao comportado esperado quanto a tolerância a falhas nas integrações.

- Importante notar que devemos garantir o funcionamento das partes do código desenvolvido e não do framework usado para as conexões. Não devemos testar estes frameworks

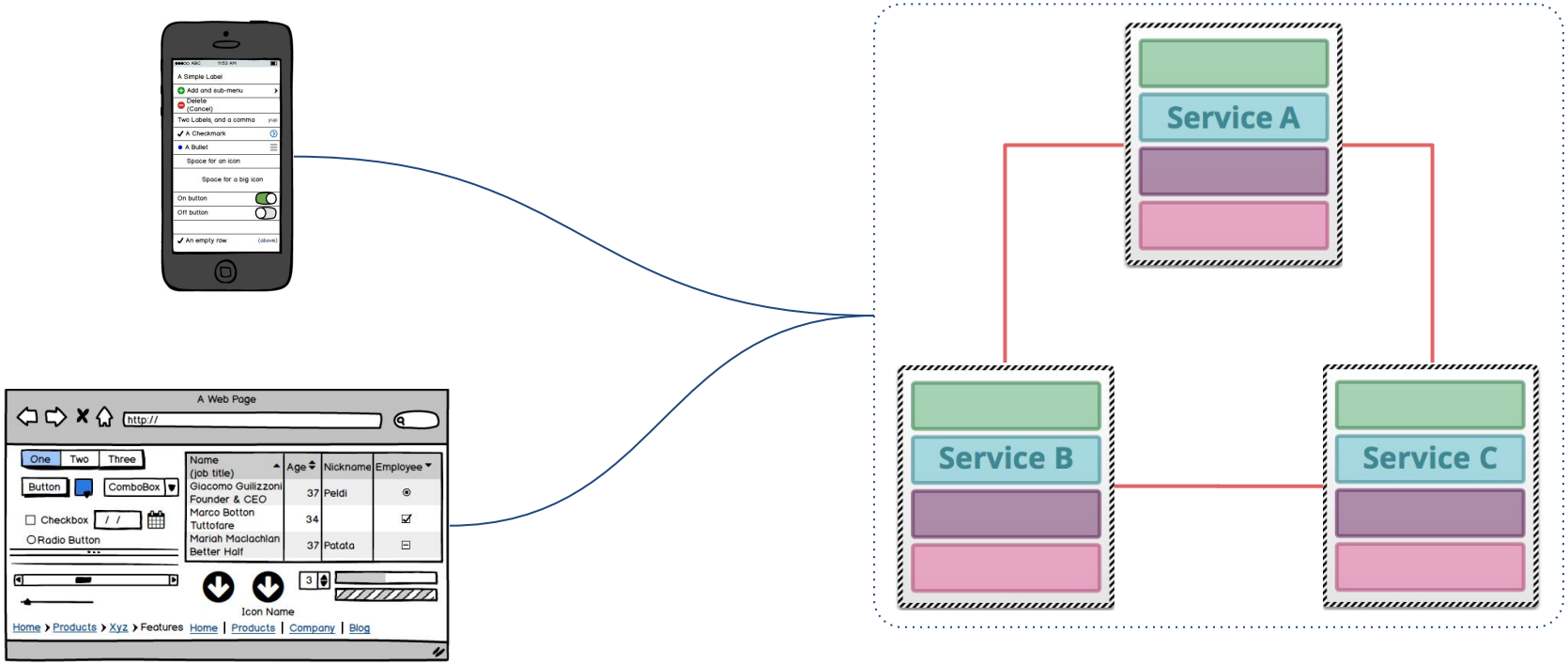
- Este teste deve ser feito o mais próximo possível do código, em muitos casos é usado o mesmo conjunto de ferramentas de teste que o teste unitário.

- Os elementos de integração reais devem ser **mockados** via ferramentas ou no caso de database por estratégia de in-memory database.

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

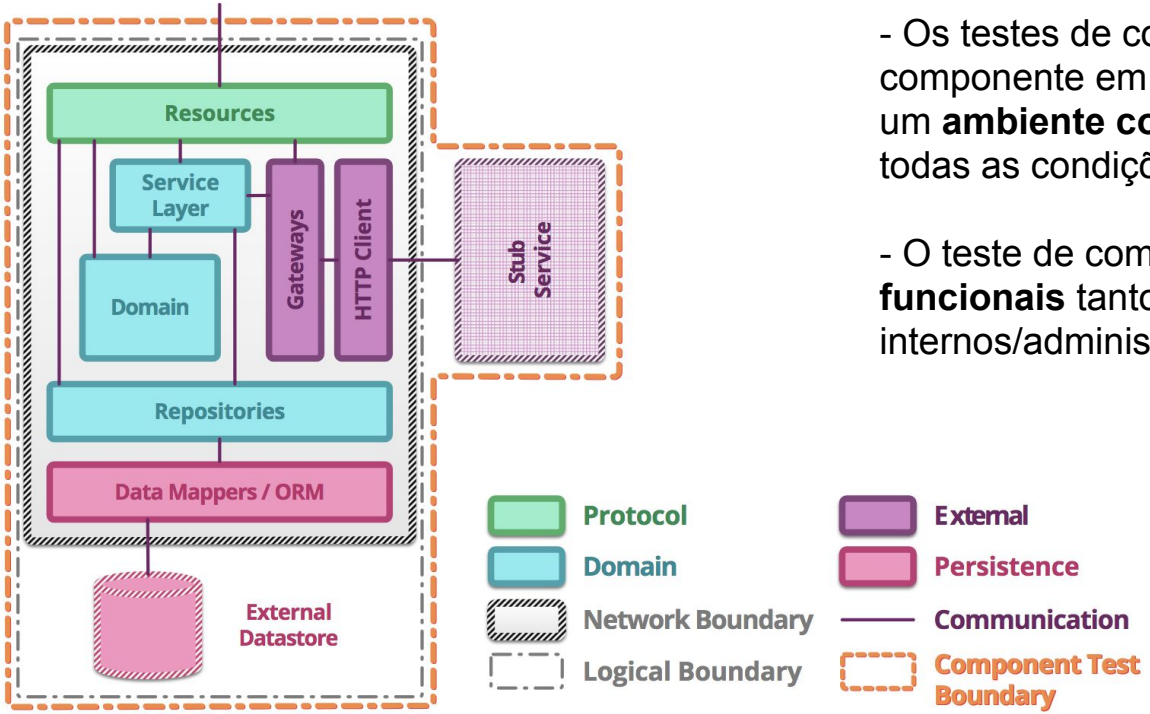
Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



"O **teste de componente** limita o escopo do software a uma parte do sistema testado."

THE DEVELOPER'S CONFERENCE



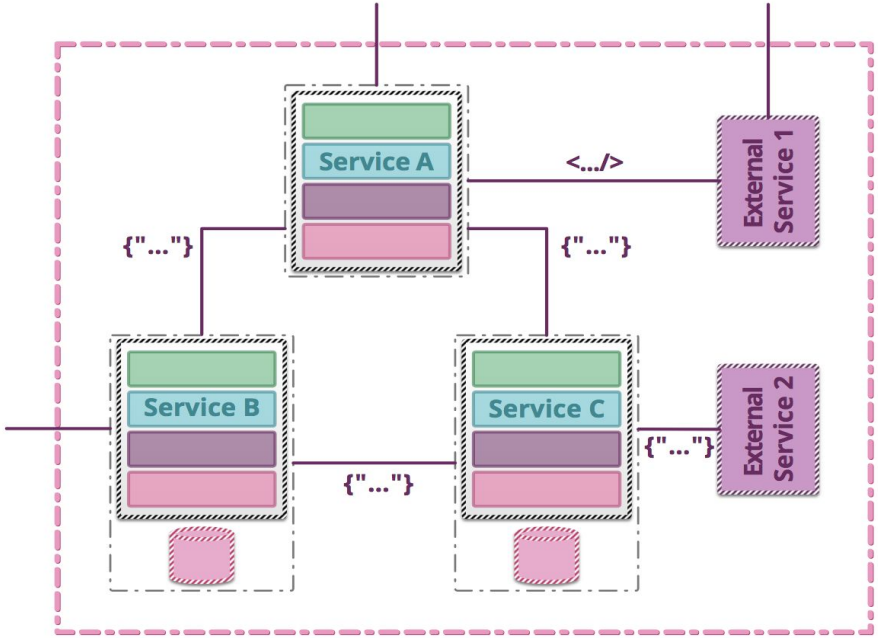
- Os testes de componente deve ser realizado com o componente em **seu estado completo e "rodando"** em um **ambiente controlad**o. Onde seja possível determinar todas as condições de uso.

- O teste de componente deve conter as **variações funcionais** tanto de recursos públicos e recursos internos/administrativos.

- Muitos cenários envolvem chamadas externas e persistência em elementos de dados, neste caso deve ser feito uso de **mock** e elementos de dados vazios

"O **teste de end-to-end** verifica se um sistema atende aos requisitos externos e atinge seus objetivos, testando todo o sistema, de ponta a ponta."

THE DEVELOPER'S CONFERENCE



- O teste de end-to-end (e2e) visa garantir que o comportamento do sistema de ponta a ponta está como esperado/especificado do ponto de vista da j**ornada do usuário**.

- Devido a complexidade de se criar um ambiente conectado com massa de dados para testes, os casos de teste incluídos no e2e precisam ser cuidadosamente avaliados para garantir que a **gestão da massa de dados** seja possível.

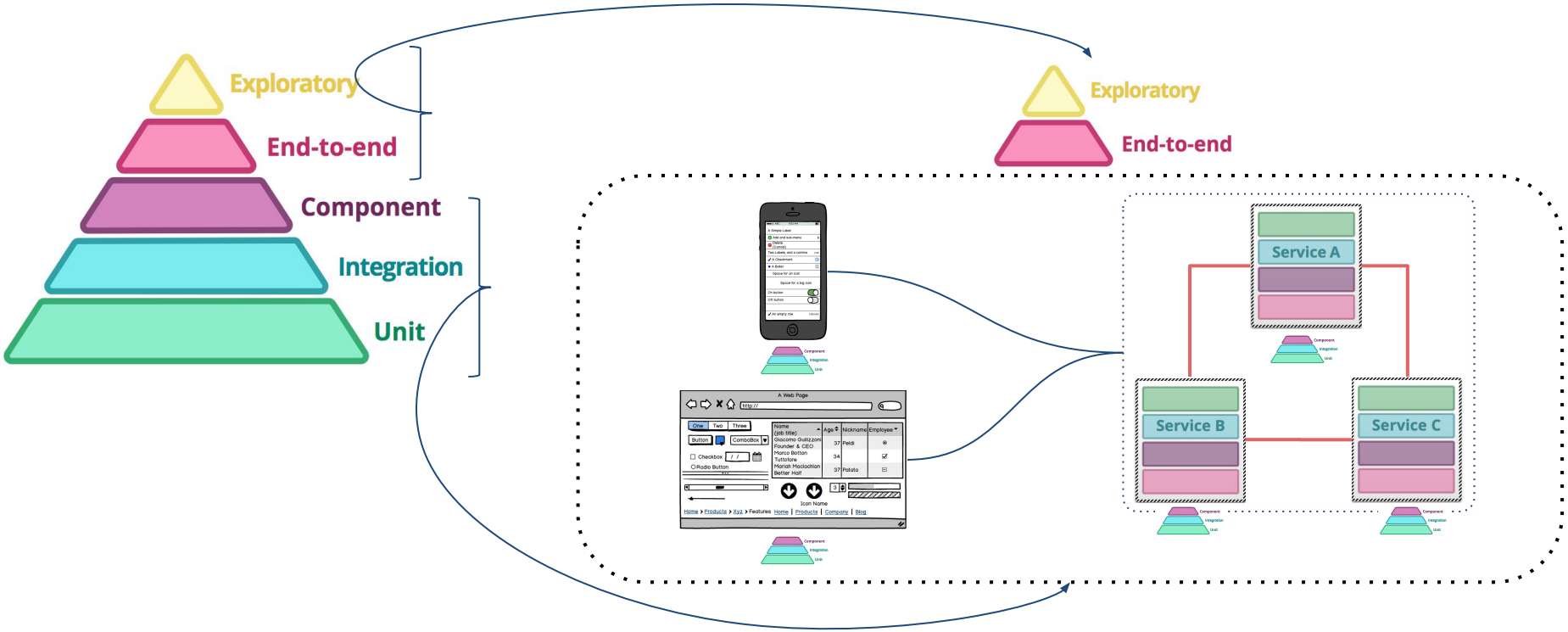
- Dado a complexidade dos testes, se faz necessário também uma análise cuidadosa de como realizar o teste e garantir o cenário para que **não seja data specific** e ao mesmo tempo garanta o resultado.

"The Need For Speed: Drive Velocity And Quality With DevOps"

Forrester, Fevereiro 2017.

THE DEVELOPER'S CONFERENCE

Testes end-to-end e exploratórios estão ligados a jornadas dos usuários e não a módulos ou elementos específicos.



Testes unitários, integração e componente estão conectados a elementos da arquitetura e módulos

Estratégia de Deploy

"The Need For Speed: Drive Velocity And Quality With DevOps"

Forrester, Fevereiro 2017.

THE DEVELOPER'S CONFERENCE

Desenvolvimento e teste de features e bugs em ambiente local;

Criação testes unitários, Integração e componentes automatizados;

Definições de feature toggle e *analytics* para acompanhamento da feature em produção;

AWS CodeCommit

Versionamento em repositório na cloud para garantia de disponibilidade;

Uso de SCM que garanta o processo desenvolvimento e o que sobe para produção através de Trunk Based Development;

AWS CodePipeline

Toda geração de pacote é feita de forma automatizada e a cada commit;

Validação das features utilizando a piramide de testes automatizados;

Análise estática de código, cobertura de testes configurados para o fluxo;

Processos documentados através de código, e tornam-se parte da aplicação;

amazon web services™

Validação das features/bugs em ambiente similar ao produtivo (Infra, componentes de arquitetura);

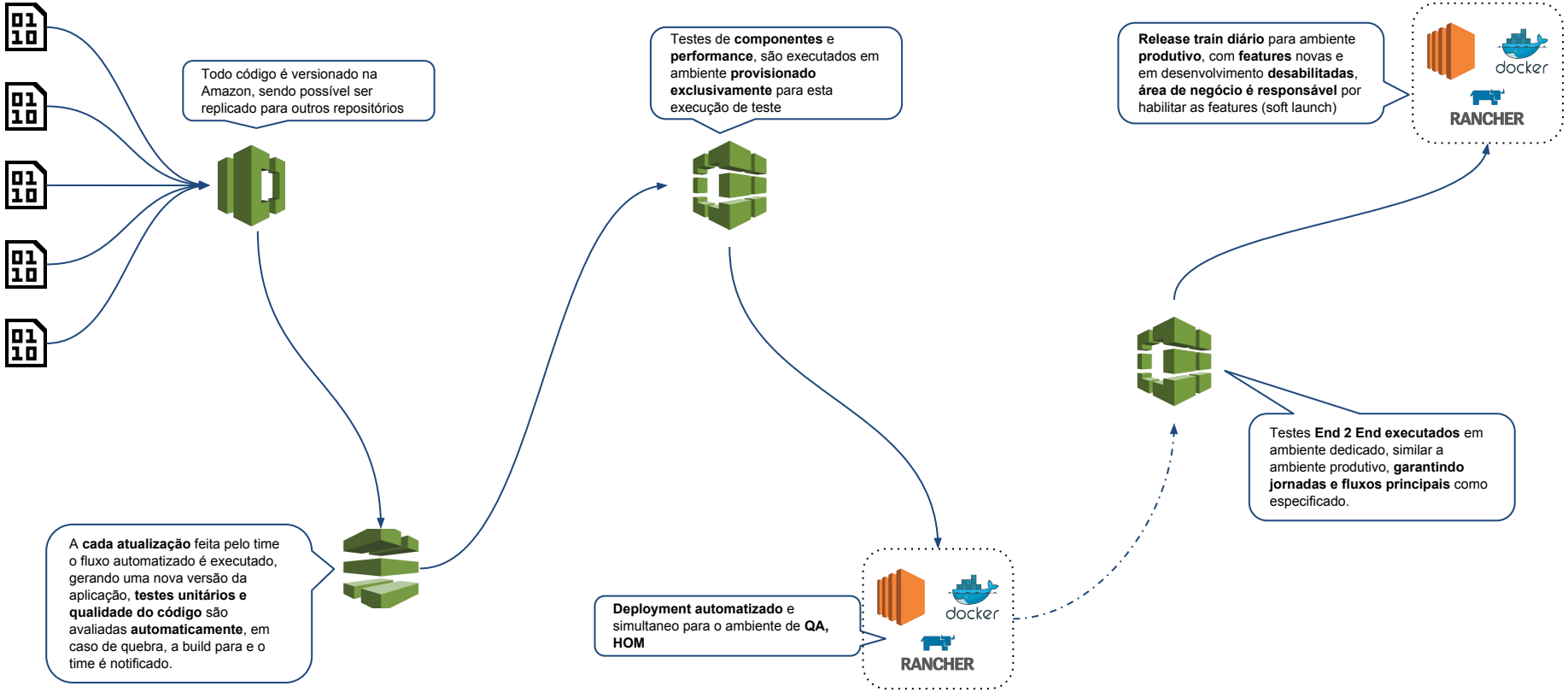
Testes exploratórios realizados por membro do time dedicado a QA;

Gestão da qualidade, com acompanhamento de retrabalho, impacto para o negócio e causa raiz;

"The Need For Speed: Drive Velocity And Quality With DevOps"

Forrester, Fevereiro 2017.

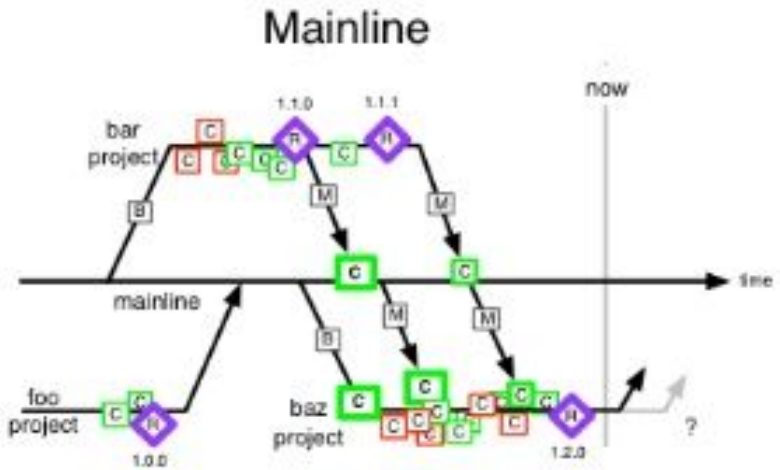
THE DEVELOPER'S CONFERENCE

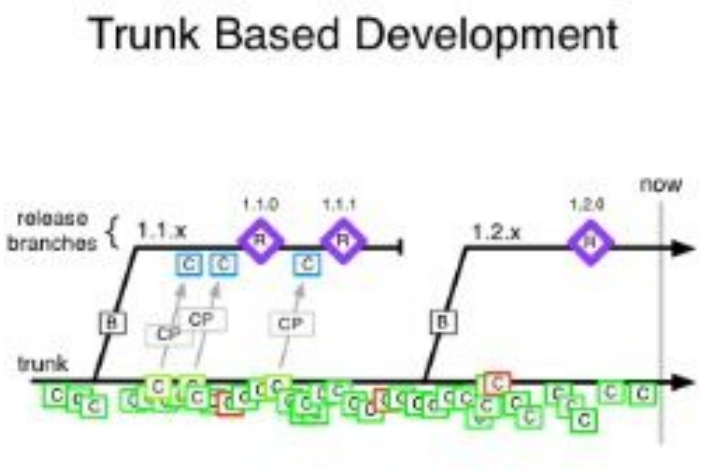


"The Need For Speed: Drive Velocity And Quality With DevOps"

Forrester, Fevereiro 2017.

THE DEVELOPER'S CONFERENCE



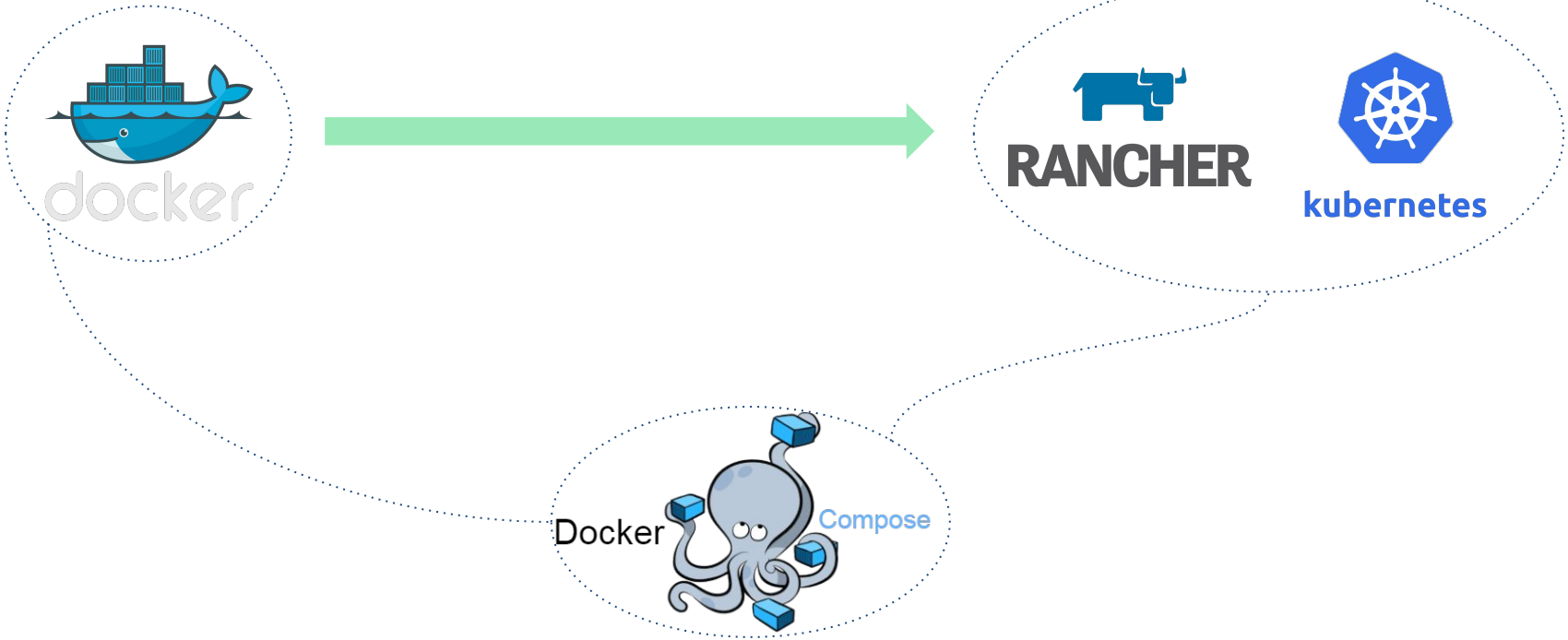


Operação

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



1. Quais são as **funcionalidades mais acessadas**? Qual o tempo de resposta dessas funcionalidades **críticas**?

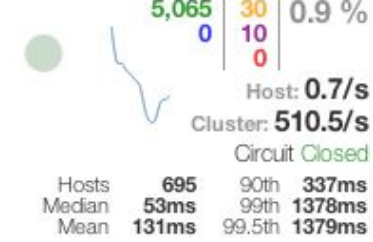
2. Quantos **usuários estão acessando o sistema** simultaneamente e de qual região (IP)?

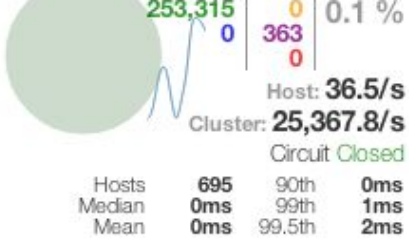
3. Qual foi a quantidade de **erros não tratados** nos últimos dias (instabilidade)? **Onde estão os logs**?

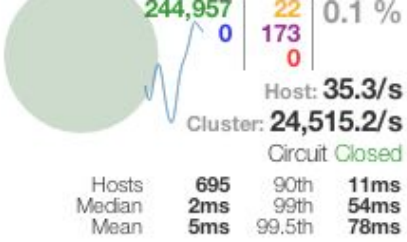
4. Qual é o **tempo gasto para salvar um carregamento**?

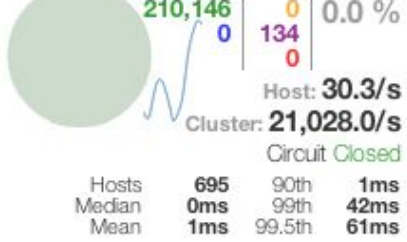
5. Quantos **eventos são gerados por dia**? Qual é o horário de pico de geração de eventos?

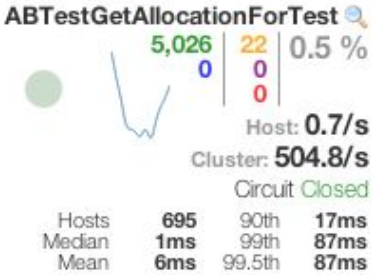
6. Existe **degradação no tempo de resposta** nos períodos de pico?

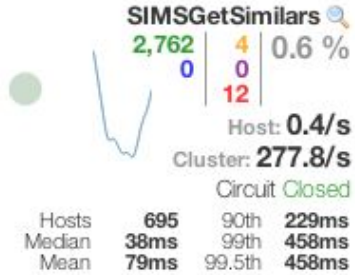

5,065
30
0.9 %
0
10
0
Host: 0.7/s
Cluster: 510.5/s
Circuit Closed
Hosts 695 90th 337ms
Median 53ms 99th 1378ms
Mean 131ms 99.5th 1379ms


253,315
0
0.1 %
0
363
0
Host: 36.5/s
Cluster: 25,367.8/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 1ms
Mean 0ms 99.5th 2ms

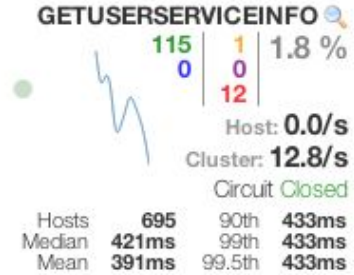

244,957
22
0.1 %
0
173
0
Host: 35.3/s
Cluster: 24,515.2/s
Circuit Closed
Hosts 695 90th 11ms
Median 2ms 99th 54ms
Mean 5ms 99.5th 78ms

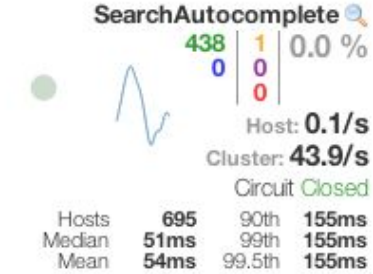

210,146
0
0.0 %
0
134
0
Host: 30.3/s
Cluster: 21,028.0/s
Circuit Closed
Hosts 695 90th 1ms
Median 0ms 99th 42ms
Mean 1ms 99.5th 61ms

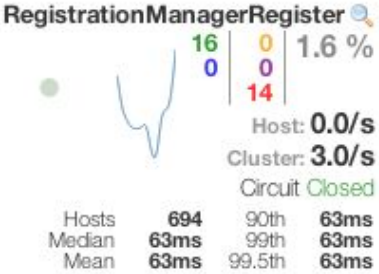

ABTestGetAllocationForTest
5,026
22
0.5 %
0
0
0
Host: 0.7/s
Cluster: 504.8/s
Circuit Closed
Hosts 695 90th 17ms
Median 1ms 99th 87ms
Mean 6ms 99.5th 87ms

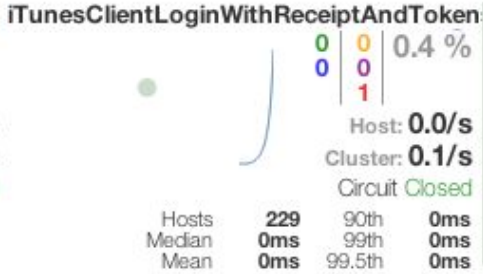

SIMSGetSimilars
2,762
4
0.6 %
0
0
12
Host: 0.4/s
Cluster: 277.8/s
Circuit Closed
Hosts 695 90th 229ms
Median 38ms 99th 458ms
Mean 79ms 99.5th 458ms

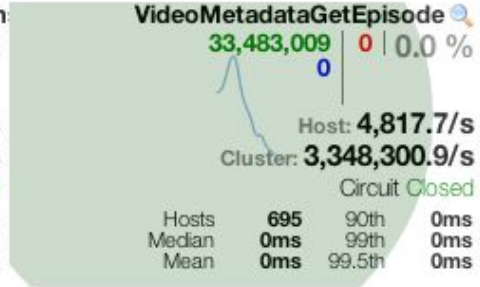

SocialGetHiddenStatus
1,749
0
0.5 %
0
0
0
Host: 0.3/s
Cluster: 174.9/s
Circuit Closed
Hosts 694 90th 156ms
Median 25ms 99th 150ms
Mean 40ms 99.5th 150ms

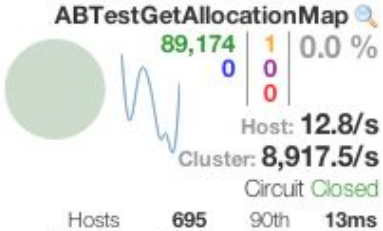

GETUSERSERVICEINFO
115
1
1.8 %
0
0
12
Host: 0.0/s
Cluster: 12.8/s
Circuit Closed
Hosts 695 90th 433ms
Median 421ms 99th 433ms
Mean 391ms 99.5th 433ms

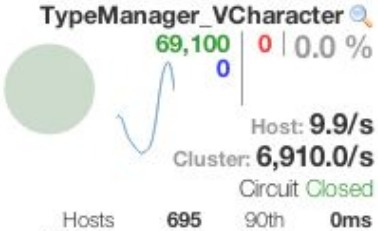

SearchAutocomplete
438
1
0.0 %
0
0
0
Host: 0.1/s
Cluster: 43.9/s
Circuit Closed
Hosts 695 90th 155ms
Median 51ms 99th 155ms
Mean 54ms 99.5th 155ms

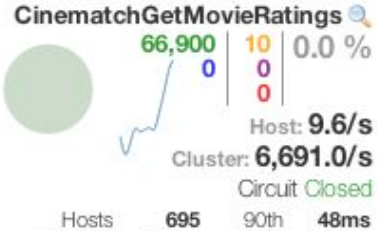

RegistrationManagerRegister
16
0
1.6 %
0
0
14
Host: 0.0/s
Cluster: 3.0/s
Circuit Closed
Hosts 694 90th 63ms
Median 63ms 99th 63ms
Mean 63ms 99.5th 63ms

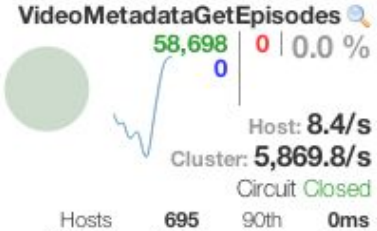

iTunesClientLoginWithReceiptAndToken
0
0
0.4 %
0
0
1
Host: 0.0/s
Cluster: 0.1/s
Circuit Closed
Hosts 229 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms


VideoMetadataGetEpisode
33,483,009
0
0.0 %
0
Host: 4,817.7/s
Cluster: 3,348,300.9/s
Circuit Closed
Hosts 695 90th 0ms
Median 0ms 99th 0ms
Mean 0ms 99.5th 0ms


ABTestGetAllocationMap
89,174
1
0.0 %
0
0
0
Host: 12.8/s
Cluster: 8,917.5/s
Circuit Closed
Hosts 695 90th 13ms


TypeManager_VCharacter
69,100
0
0.0 %
0
Host: 9.9/s
Cluster: 6,910.0/s
Circuit Closed
Hosts 695 90th 0ms

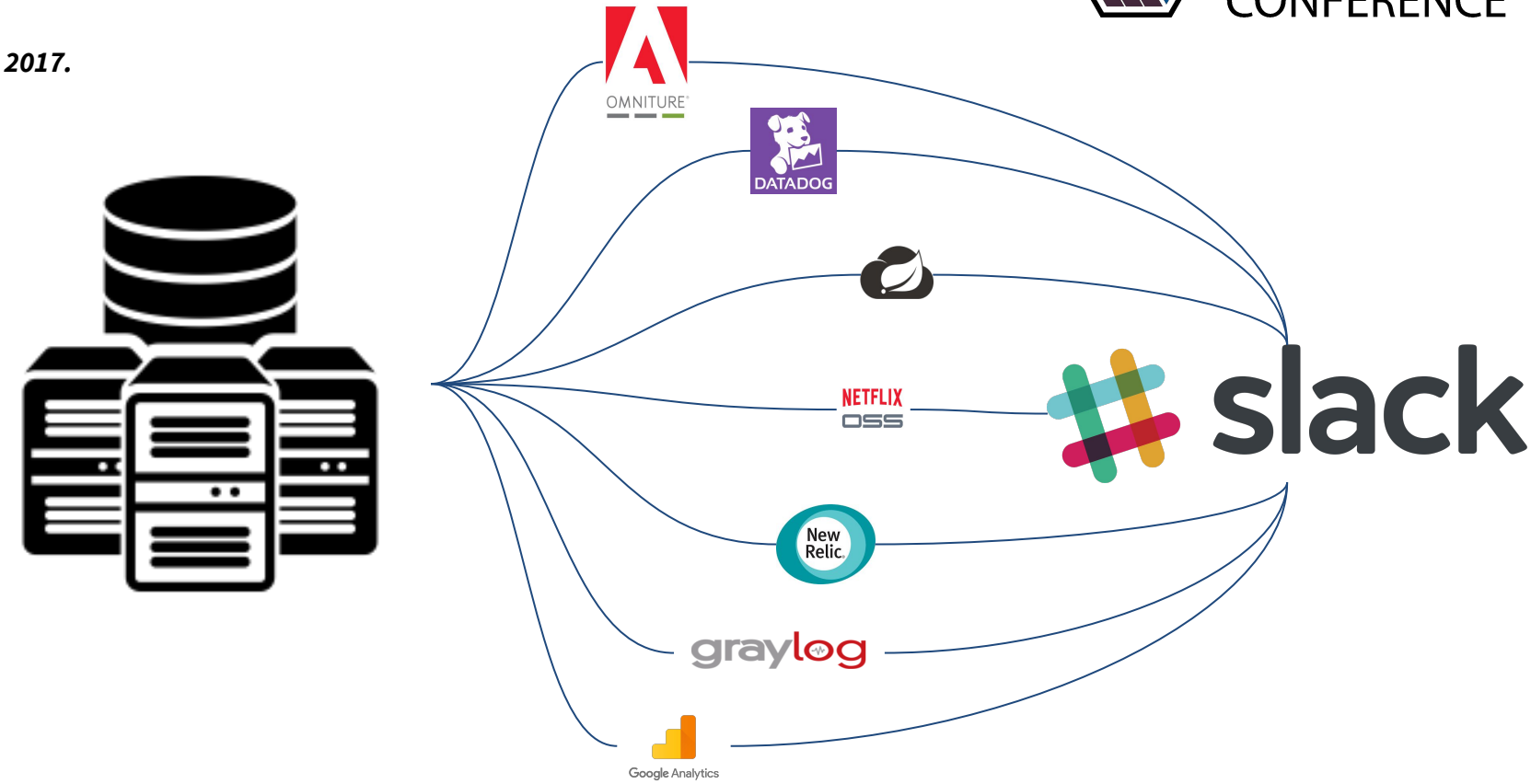

CinematchGetMovieRatings
66,900
10
0.0 %
0
0
0
Host: 9.6/s
Cluster: 6,691.0/s
Circuit Closed
Hosts 695 90th 48ms


VideoMetadataGetEpisodes
58,698
0
0.0 %
0
Host: 8.4/s
Cluster: 5,869.8/s
Circuit Closed
Hosts 695 90th 0ms

"Microservices architecture is an approach for building distributed applications that support agile and scalable delivery."

Gartner, Março 2017.

THE DEVELOPER'S CONFERENCE



Talk is cheap. Show me the code.

— Linus Torvalds

$~: git clone awsome_code

THE DEVELOPER'S CONFERENCE

Thank you!

CI&T

http://vempra.ciandt.com