

Microservices

Vinicio Garcia, Saulo Guilhermino, Emerson Victor, Gabriel Pessoa, Ramom Pereira dos Santos Silva, Rafael Mota Alves, Marcos Galvão, Izabel...



41 pessoas

1 administrador(a)

40 membros



421 questões essenciais

104 ferramentas



1694 falas

1260 respostas

434 comentários às respostas

Engajamento por ferramenta

Jornada para Microservice

 29 pessoas
 4 questões
 182 falas
108 respostas
74 comentários

Onboard Microservices

 34 pessoas
 5 questões
 167 falas
166 respostas
1 comentário

Lehman para MBA's

 22 pessoas
 4 questões
 161 falas
82 respostas
79 comentários

Arquitetura Microsserviços

 31 pessoas
 4 questões
 139 falas
109 respostas
30 comentários

Aplicando Microsserviços

 27 pessoas
 4 questões
 132 falas
96 respostas
36 comentários

Casos de Uso de Microsserviços

 24 pessoas
 5 questões
 131 falas
59 respostas
72 comentários

Escalabilidade & Elasticidade

 20 pessoas
 4 questões
 107 falas
64 respostas
43 comentários

Escalonamento automático

 14 pessoas
 3 questões
 52 falas
34 respostas
18 comentários

Auto-organização de microsserviços

 13 pessoas
 4 questões
 49 falas
32 respostas
17 comentários

Observabilidade como função

 12 pessoas
 4 questões
 43 falas
16 respostas
27 comentários

Logs e Monitoramento

 11 pessoas
 5 questões
 33 falas
21 respostas
12 comentários

Matriz CSD

 2 pessoas
 5 questões
 17 falas
17 respostas
0 comentários

Matriz CSD

 2 pessoas
 5 questões
 16 falas
16 respostas
0 comentários

Post-mortem

 2 pessoas
 6 questões
 15 falas
14 respostas
1 comentário

Post-mortem

 2 pessoas
 6 questões
 14 falas
14 respostas
0 comentários

Post-mortem

 2 pessoas
 6 questões
 14 falas
14 respostas
0 comentários

Matriz CSD

3 pessoas

5 questões

13 falas

12 respostas

1 comentário

Post-mortem

1 pessoa

4 questões

13 falas

11 respostas

2 comentários

Post-mortem

1 pessoa

6 questões

12 falas

10 respostas

2 comentários

Arquitetura de Software

1 pessoa

4 questões

11 falas

10 respostas

1 comentário

Arquitetura de Software

1 pessoa

4 questões

11 falas

11 respostas

0 comentários

Matriz CSD

4 pessoas

5 questões

11 falas

11 respostas

0 comentários

Post-mortem

1 pessoa

6 questões

11 falas

11 respostas

0 comentários

Post-mortem

1 pessoa

6 questões

9 falas

9 respostas

0 comentários

desafio de projeto

1 pessoa

4 questões

9 falas

9 respostas

0 comentários

Post-mortem

1 pessoa

5 questões

9 falas

9 respostas

0 comentários

Post-mortem

1 pessoa

6 questões

9 falas

9 respostas

0 comentários

proposta de valor

1 pessoa

5 questões

9 falas

9 respostas

0 comentários

Matriz CSD

2 pessoas

5 questões

9 falas

8 respostas

1 comentário

proposta de valor

1 pessoa

5 questões

9 falas

9 respostas

0 comentários

Dona Deda

7 pessoas

1 questão

8 falas

8 respostas

0 comentários

Arquitetura de Software

1 pessoa

2 questões

8 falas

8 respostas

0 comentários

desafio de projeto

1 pessoa

4 questões

Post-mortem

1 pessoa

6 questões

 8 falas
8 respostas
0 comentários

 7 falas
7 respostas
0 comentários

 **desafio de projeto**
 2 pessoas
 4 questões
 7 falas
7 respostas
0 comentários

 **desafio de projeto**
 2 pessoas
 4 questões
 7 falas
7 respostas
0 comentários

 **Atributos de Qualidade**
 2 pessoas
 3 questões
 7 falas
5 respostas
2 comentários

 **Post-mortem**
 1 pessoa
 5 questões
 7 falas
7 respostas
0 comentários

 **desafio de projeto**
 1 pessoa
 4 questões
 7 falas
7 respostas
0 comentários

 **Matriz CSD**
 2 pessoas
 5 questões
 7 falas
6 respostas
1 comentário

 **Post-mortem**
 1 pessoa
 6 questões
 7 falas
7 respostas
0 comentários

 **desafio de projeto**
 1 pessoa
 4 questões
 7 falas
7 respostas
0 comentários

 **Post-mortem**
 1 pessoa
 6 questões
 6 falas
6 respostas
0 comentários

 **Post-mortem**
 2 pessoas
 5 questões
 6 falas
5 respostas
1 comentário

 **Atributos de Qualidade**
 2 pessoas
 3 questões
 6 falas
6 respostas
0 comentários

 **Arquitetura de Software**
 3 pessoas
 4 questões
 6 falas
6 respostas
0 comentários

 **proposta de valor**
 1 pessoa
 3 questões
 6 falas
5 respostas
1 comentário

 **Post-mortem**
 1 pessoa
 6 questões
 6 falas
6 respostas
0 comentários

 **Hackenge**
 1 pessoa
 3 questões
 6 falas
5 respostas
1 comentário

 **Post-mortem**
 1 pessoa
 6 questões
 6 falas
6 respostas
0 comentários

Matrix CSD

3 pessoas
 5 questões
 6 falas
 5 respostas
 1 comentário

Dúvidas

2 pessoas
 1 questão
 6 falas
 1 resposta
 5 comentários

Arquitetura de Software

1 pessoa
 4 questões
 5 falas
 5 respostas
 0 comentários

proposta de valor

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Equipe 05

2 pessoas
 3 questões
 5 falas
 5 respostas
 0 comentários

proposta de valor

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

proposta de valor

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

proposta de valor

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

3 pessoas
 5 questões
 5 falas
 5 respostas
 0 comentários

Dona Deda

2 pessoas
 1 questão
 5 falas
 4 respostas
 1 comentário

Atributos de Qualidade

1 pessoa
 3 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Post-mortem

1 pessoa
 5 questões
 5 falas
 5 respostas
 0 comentários

Arquitetura de Software

1 pessoa
 0 comentários

Hackenge

1 pessoa
 0 comentários

 4 questões

 4 falas

4 respostas

0 comentários

 2 questões

 4 falas

2 respostas

2 comentários

Arquitetura de Software

 1 pessoa

 4 questões

 4 falas

4 respostas

0 comentários

desafio de projeto

 1 pessoa

 4 questões

 4 falas

4 respostas

0 comentários

Equipe 07

 3 pessoas

 2 questões

 4 falas

4 respostas

0 comentários

Atributos de Qualidade

 1 pessoa

 2 questões

 4 falas

4 respostas

0 comentários

Equipe 03

 2 pessoas

 2 questões

 3 falas

3 respostas

0 comentários

Equipe 04

 2 pessoas

 3 questões

 3 falas

3 respostas

0 comentários

Hackenge

 1 pessoa

 3 questões

 3 falas

3 respostas

0 comentários

Atributos de Qualidade

 1 pessoa

 3 questões

 3 falas

3 respostas

0 comentários

Atributos de Qualidade

 1 pessoa

 2 questões

 3 falas

2 respostas

1 comentário

Hackenge

 1 pessoa

 2 questões

 3 falas

3 respostas

0 comentários

Atributos de Qualidade

 2 pessoas

 3 questões

 3 falas

3 respostas

0 comentários

Equipe 01

 2 pessoas

 2 questões

 3 falas

3 respostas

0 comentários

Equipe 06

 2 pessoas

 2 questões

 3 falas

3 respostas

0 comentários

Equipe 02

 2 pessoas

 2 questões

 2 falas

2 respostas

0 comentários

Hackenge

 1 pessoa

 2 questões

 2 falas

2 respostas

0 comentários

Dúvidas

 2 pessoas

 1 questão

 2 falas

1 resposta

1 comentário

 Dúvidas
 1 pessoa
 1 questão
 1 fala 1 resposta 0 comentários

 Dona Deda
 1 pessoa
 1 questão
 1 fala 1 resposta 0 comentários

 Dúvidas
 1 pessoa
 1 questão
 1 fala 1 resposta 0 comentários

O projeto conta com **41 pessoas**, tendo a seguinte composição: **1 administrador e 40 membros**.

No período de **20/08/2020** até **30/11/2020** foram aplicados nos mapas modulares um total de **104 ferramentas** (hexágonos). Ao todo, foram registradas **1694 falas**. Uma análise de estatística descritiva indica que o projeto possui uma média de **16.29 falas por ferramenta** e de **41.32 falas por membro**.

Na plataforma strateegia.digital existem dois tipos de falas: as respostas diretas às questões e os comentários a essas respostas. Do total de **1694 falas** houve **1260 respostas (74.38% do total de falas)** e **434 comentários às respostas (25.62% do total de falas)**. Entre as 1260 respostas, 294 delas foram provocativas o suficiente para gerar comentários respostas.

2020.3

necessário **comunicação forma** utilizar **dados onde exemplo** pois caso **capacidade arquitetura** bem apenas funcionalidades **desenvolvimento APIs** **microserviços** **escalabilidade sistemas** Postem **requisições uso** empresa **padrão sistema** além **complexidade outros** **aplicações sendo diferentes API serviços** assim **contexto podem processo** fazer **tempo sobre microserviços cada software serviço** **aplicação lei recursos maior deve possível**

O projeto conta com **41 pessoas**, tendo a seguinte composição: **1 administrador e 40 membros**.

No período de **20/08/2020** até **30/11/2020** foram aplicados nos mapas modulares um total de **33 ferramentas** (hexágonos). Ao todo, foram registradas **1238 falas**. Uma análise de estatística descritiva indica que o projeto possui uma média de **37.52 falas por ferramenta** e de **30.2 falas por membro**.

Na plataforma strateegia.digital existem dois tipos de falas: as respostas diretas às questões e os comentários a essas respostas. Do total de **1238 falas** houve **819 respostas (66.16% do total de falas)** e **419 comentários às respostas (33.84% do total de falas)**. Entre as 819 respostas, 282 delas foram provocativas o suficiente para gerar comentários respostas.

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar



Vinicius Garcia 24/08/2020

Postem aqui as suas dúvidas :D

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Links compartilhados

 **Vinicius Garcia**

Podcast Futuros Digitais (spotify)

https://open.spotify.com/show/75J0gYMewiTw1Y7r4yQLMM?si=8kmpBxN5Tia8BpXzBJZ_8w

 **victor sena de lima attar**

podcast - caelum -> B-A-BÁ do microsserviços

<https://hipsters.tech/microservicos-hipsters-17/>

 **victor sena de lima attar**

Podcast - caelum -> Microsserviços na caelum

<https://hipsters.tech/microsservicos-na-caelum-hipsters-on-the-road-6/>

 **victor sena de lima attar**

Poddcast - caelum -> Monolitos na caelum

<https://hipsters.tech/monolitos-cultura-e-dia-a-dia-na-alura-hipsters-on-the-road-31/>

 **victor sena de lima attar**

Podcast - caelum -> Tecnologias na stackoverflow

<https://hipsters.tech/tecnologias-na-stackoverflow-hipsters-46/>

 **Ricardo Ebbers Carneiro Leão**

Ebooks disponíveis de graça pela RedHat

<https://developers.redhat.com/ebooks>

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

 **Vinicius Garcia** 24/08/2020

Adicionei o link para o podcast Futuros Digitais da TDS Company no Spotify

 **victor sena de lima attar** 24/08/2020

Adicionei alguns podcastas do hipsters que falam de microsserviços!

 **Roberto Oliveira** 26/08/2020

Seria interessante que a listagem dos comentários priorizasse o comentários do usuário logado, passei um tempo pra achar meus comentários.

 **Saulo Guilhermino** 26/08/2020

Sugestão pra plataforma: Melhorar o sinal de notificação (bolinha vermelha ao lado do nome do projeto) para que ela possa direcionar o usuário ao conteúdo novo postado desde a ultima visualização

 **Gabriel D'Luca** 30/08/2020

Sugestão pra plataforma: Não possui funcionalidade pra editar um comentário. É preciso copiar o texto, apagar e colar novamente em um novo comentário.

 **Marcos Galvão** 31/08/2020

Sugestão pra plataforma: Eu não sei se é devido a resolução do meu PC mas quando estou digitando um comentário e a caixa com o número de caracteres aparece, ela encobre o que está escrito mais a direita da parte superior..

 **Gabriel D'Luca** 31/08/2020

Sugestão pra plataforma: Ao clicar no link do email onde diz fui selecionado pra um novo ponto de conversação, não há nenhum aviso ou feedback visual na plataforma que indica >qual< é esse novo ponto de conversação... e o email em si tbm não informa sobre isso.

 **Izabella Nascimento** 02/09/2020

 **Rua na plataforma:** ao querer deletar um link em Links compartilhados o botão de Deletar para <https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Drog na plataforma, ao querer deletar um link com links compartilhados o botão de Deletar para de funcionar apóis usá-lo uma vez, se quiser apagar novamente o link o botão fica como já clicado e não é possível deletar, só volta ao recarregar a página.

Onboard Microservices

Métodos [de trabalho]

Método [de imersão] para identificar informações na construção de uma jornada SUFICIENTE para o estudante entender o contexto e EFICIENTE para ser usada como base para o processo de ENSINO e APRENDIZAGEM na disciplina

Referências

Desenvolvimento de Aplicações Nativas da Nuvem com Arquitetura Baseada em Microservices
<https://bit.ly/vcg-microservices>

Qual sua EXPECTATIVA para esta jornada de aprendizagem?

RE Ricardo Ebbers Carneiro Leão 24/08/2020

Minha expectativa é ao final da disciplina ao menos estar confortável em fazer refatorações de monolitos para microserviços, e desenhar uma arquitetura de complexidade média em microserviços

RO Roberto Oliveira 24/08/2020

Entender e arquitetar futuros projetos utilizando microservicos.

HM Hiro Miyakawa 24/08/2020

Aprender um pouco mais sobre essa área que não faço ideia do que seja. Estou aberto para entender mais sobre essa parte de desenvolvimento.

RJ Renato Joaquim de Miranda Ferreira 24/08/2020

Minha expectativa nessa cadeira é entender como funciona o mundo de microserviços, uma vez que sou leigo nesse assunto, e como pode ser usado em diversas aplicações para os mais diversos fins.

Marcílio Freitas 24/08/2020

Minha expectativa é voltar a um ritmo de estudos e aprendizagem que faz um tempo que não tenho, mas que necessito para poder terminar o curso e aprender sobre o assunto para saber bem o que vou ser além do "básico" desenvolvedor de sw.

RM Rafael Mota Alves 24/08/2020

Aprender conceitos sobre microserviços e obter experiência prática nas tecnologias

SG Saulo Guilhermino 24/08/2020

Compreender melhor a teoria, aprimorar a prática do desenvolvimento de microservices e otimizar sua aplicação profissionalmente

LB Lucas Barros 24/08/2020

Aprender sobre os casos de uso para aplicar o método de microservices, aprender novas tecnologias e métodos que auxiliem nessa maneira de desenvolvimento

MG Marcos Galvão 24/08/2020

Minha expectativa ao fim da disciplina é estar apto a entender as características e o funcionamento de aplicações em microserviços

CP Claudio Pacheco 24/08/2020

Conhecer sobre o que é o modelo de microservices e sua aplicabilidade prática na arquitetura de software

DL Danilo Lira 24/08/2020

Entender quais as vantagens da utilização de microservices quando comparados com monólitos e como desenvolver sistemas desta forma

Dairon Eugênio Martins 24/08/2020

Ter uma maior compreensão sobre sistemas arquitetados por meio de microserviços.

C LC Lucas Cardoso 24/08/2020

Entender como funcionam microsserviços e ser capaz de implementar uma aplicação que siga esse padrão

AR **Antonio Rodrigues** 24/08/2020

Entender mais sobre a arquitetura de microsserviços e sobre a área em geral.

RP **Ramom Pereira dos Santos Silva** 24/08/2020

Aprender temas relacionados a cultura de Microserviços e DevOps, além de experimentar uma nova abordagem de aprendizado!

TM **Talyta Maria Rosas Pacheco** 24/08/2020

Aprender na prática sobre a arquitetura de microsserviços

Izabella Priscylla da Costa Nascimento 24/08/2020

Minha expectativa é ter mais conhecimento sobre microsserviços, pois sei pouco relacionado ao tema.

Gabriel Pessoa 24/08/2020

Entender mais sobre microsserviços e saber aplicar esses conhecimentos em aplicações reais

VS **victor sena de lima attar** 24/08/2020

Entender conceitos mais avançados de microsserviços, pois já sei o básico dessa arquitetura. Ainda quero ter mais discussões pois sei que grandes empresas ainda usam monolíticos.

AF **Arthur Frade de Araújo** 24/08/2020

Aprender sobre arquitetura Serverless

João Vasconcelos 24/08/2020

Entender sobre a teoria, prática, técnicas, ferramentas e oportunidades em micro serviços.

EV **Emerson Victor** 24/08/2020

Aprender, de forma prática, como funciona e como aplicar a arquitetura microsserviços

HC **Heitor Carvalho** 24/08/2020

Entender a arquitetura e aplicabilidade de microsserviços, as vantagens e desvantagens dos impactos de se utilizar esse tipo de arquitetura e ter uma base de conhecimentos suficiente para implementar ou manter um sistema que a utilize.

VG **Victor Gabryel** 24/08/2020

A expectativa que eu tenho para essa jornada é compreender e conseguir por em prática a arquitetura de microservices.

Luca 24/08/2020

Conseguir entender os cenários onde microsserviços são interessantes e aplicáveis, assim como entender na prática sobre a implementação da arquitetura

JV **Josenildo Vicente** 24/08/2020

Entender e aprender sobre microsserviços

LB **Luan Brito** 24/08/2020

Aprender sobre microsserviços e experienciar um modelo novo de aprendizado

IT **Isac Tomaz da Silva** 24/08/2020

Minha expectativa é aprender sobre temas importantes de microservices e sobre como trabalhar com ele

Wellington Oliveira 24/08/2020

Aprender Microserviços, desenvolver um projeto explorando seus recursos e apresentar esse projeto com um artigo como Trabalho de Conclusão de Curso.

IF **Igor Fernandes** 24/08/2020

Aprender sobre conceitos de microsserviços, desenvolver algum projeto com esses aprendizados

Daniel Silva 24/08/2020

desenvolver conhecimento prático em microservices e tentar encaixar o uso desse conhecimento em projetos futuros.

GC **Gabriel Cavalcanti de Melo** 24/08/2020

Aprender mais sobre microsserviços, pois ainda não tenho conhecimento sobre a área, e tbm outras habilidades que posso adquirir no decorrer da cadeira



Gabriel D'Lucca 24/08/2020

Aprofundar os conceitos sobre arquiteturas de microserviços (vs. monolitos) e fazer um paralelo sobre como posso aplicar esses conceitos de modularização em mobile.

 **Elverson Melo** 24/08/2020

Compreender os fundamentos da arquitetura de microserviços para auxiliar na construção de projetos futuros

Qual área dentro da computação te interessa atualmente (que você pretende se especializar, construir carreira)?

 **Ricardo Ebbers Carneiro Leão** 24/08/2020
Engenharia e arquitetura de software

 **Roberto Oliveira** 24/08/2020
estou lendo muito sobre usabilidade ultimamente.

 **Hiro Miyakawa** 24/08/2020
AI e Big Data (desenvolvimento) - Analytics e Produto (PM/PO)

 **Renato Joaquim de Miranda Ferreira** 24/08/2020
Desenvolvimento de software e Analytics.

 **Lucas Barros** 24/08/2020
Engenharia de Software

 **Marcos Galvão** 24/08/2020
Deep Learning

Marcílio Freitas 24/08/2020
Desenvolvimento de sw e análise de decisão.

 **Antonio Rodrigues** 24/08/2020
Redes

 **Lucas Cardoso** 24/08/2020
Engenharia de software (mobile atualmente)

Dairon Eugênio Martins 24/08/2020
Estudando Back End e Design Patterns

 **Rafael Mota Alves** 24/08/2020
Engenharia de Software e Computação em Nuvem

 **Claudio Pacheco** 24/08/2020
Qualidade de software

 **Gabriel D'Luca** 24/08/2020
Desenvolvimento Mobile (iOS)

 **Saulo Guilhermino** 24/08/2020
Cybersegurança / Segurança da Informação

 **Ramom Pereira dos Santos Silva** 24/08/2020
Machine Learning, Data Science e Computação em Nuvem

 **Luan Brito** 24/08/2020
Engenharia de Software, arte computacional e informática e sociedade

 **Danilo Lira** 24/08/2020
Desenvolvimento de sistemas

Izabella Priscylla da Costa Nascimento 24/08/2020
Desenvolvimento Front End

 **Talyta Maria Rosas Pacheco** 24/08/2020
Engenharia de dados

- AF** Arthur Frade de Araújo 24/08/2020
Backend e cloud
- EV** Emerson Victor 24/08/2020
Desenvolvimento mobile (iOS)
- HC** Heitor Carvalho 24/08/2020
Engenharia de Software, Arquiteturas, boas práticas e padrões de projeto, motivação de engenheiros de software e produtividade.
- Lucca** 24/08/2020
Desenvolvimento Mobile (iOS)
- João Vasconcelos** 24/08/2020
Engenharia de Software, backend, database, sre, engenhos de busca, machine learning
- IF** Igor Fernandes 24/08/2020
Segurança em Software (mais especificamente Web, por enquanto)
- VG** Victor Gabryel 24/08/2020
Desenvolvimento backend.
- VS** victor sena de lima attar 24/08/2020
Desenvolvimento front-end e mobile
- JV** Josenildo Vicente 24/08/2020
Iot, IA
- IT** Isac Tomaz da Silva 24/08/2020
Desenvolvimento backend
- GC** Gabriel Cavalcanti de Melo 24/08/2020
Desenvolvimento Backend
- Daniel Silva** 24/08/2020
Desenvolvimento web, front e back, com a stack do react.
- Wellington Oliveira** 24/08/2020
Desenvolvimento WEB Back. Stack .Net
- Gabriel Pessoa** 24/08/2020
Big Data e AI
- EM** Elverson Melo 24/08/2020
Data science

Você já se envolveu ou está envolvido em algum projeto de construção de software (aplicativo, aplicação, placaforma, sistema...)?

- RE** Ricardo Ebbers Carneiro Leão 24/08/2020
Sim, atualmente sou desenvolvedor backend senior e diariamente escrevo e mantendo microserviços que compõem algumas features do app do PagSeguro.
- RJ** Renato Joaquim de Miranda Ferreira 24/08/2020
Atualmente sou desenvolvedor de um sistema de monitoramento, relatórios e comissionamentos.
- RO** Roberto Oliveira 24/08/2020
Já atuei em muitos projetos, tanto frontend como backend, mas ultimamente meu foco está sendo em react native, usando ele pra integrar módulos em app nativos.
- LC** Lucas Cardoso 24/08/2020
Já participei do desenvolvimento de apps nativos, e atualmente trabalho no ecossistema mobile
- LB** Lucas Barros 24/08/2020
Sim, atualmente trabalho como desenvolvedor full stack, e escrevo e mantendo serviços que compõe os produtos da Inloco

HM

Hiro Miyakawa 24/08/2020
 A última foi do projetão em 2019.1 (co.junto)

Marcilio Freitas 24/08/2020
 Atualmente na fase inicial de um projeto de uma aplicação de controle de chamados e inventário de TI da empresa em que trabalho.

RP

Ramon Pereira dos Santos Silva 24/08/2020
 Sim, sou Co-Founder da Prepi. Construo serviços, back-end prioritariamente

LB

Luan Brito 24/08/2020
 Desenvolvo projeto de engenharia de dados e web

Izabella Priscylla da Costa Nascimento 24/08/2020
 Estou atuando em um desenvolvimento de um sistema web, na parte Front End.

RM

Rafael Mota Alves 24/08/2020
 Atualmente trabalho como engenheiro de software, desenvolvendo aplicações backend em Cloud

AF

Arthur Frade de Araújo 24/08/2020
 Atualmente atuo como dev backend na Concrete! Lá atuo desenvolvedo o serverside de aplicações mobile!

SG

Saulo Guilhermino 24/08/2020
 Sim, atualmente trabalho na Inloco como Analista de Segurança construindo e estruturando sistemas de proteção e monitoramento de aplicações

DL

Danilo Lira 24/08/2020
 Sim, recentemente participei do desenvolvimento de um app voltado para fila digital. Além disso estou desenvolvendo um site pessoal

CP

Claudio Pacheco 24/08/2020
 Desenvolvimento mais puramente foi em disciplinas passadas (Eng. SW, Projetão). No estágio, meu envolvimento está relacionado à parte de testes

HC

Heitor Carvalho 24/08/2020
 Trabalhei dois anos desenvolvendo projeto em Rails voltado para a área de EAD.

EV

Emerson Victor 24/08/2020
 Sim, já trabalhei em alguns projetos pessoais e atualmente estou trabalhando como desenvolvedor iOS na Concrete

Lucca 24/08/2020
 Atualmente atuo como dev iOS na Concrete

VS

victor sena de lima attar 24/08/2020
 Sim, trabalho no desenvolvimento de um e-commerce voltado para pets, chamado OBAPET.

IF

Igor Fernandes 24/08/2020
 Sim, desenvolvi a dois anos atrás um aplicativo gerenciamento de notas no IFPE (uma versão mobile para "SIGA" de lá).

João Vasconcelos 24/08/2020
 Trabalho como estagiário de solutions engineer no facebook

TM

Talyta Maria Rosas Pacheco 24/08/2020
 Sim, já desenvolvi aplicações em minhas experiências passadas, através de competições, cadeiras na faculdade e trabalho

IT

Isac Tomaz da Silva 24/08/2020
 Sim, mas no momento não estou atuando em nenhum

JV

Josenildo Vicente 24/08/2020
 Atualmente não, os últimos foram em 2019 em projetos das disciplinas do CIn

GD

Gabriel D'Luca 24/08/2020
 Sim, atualmente trabalho em uma app nativa que possui bastante problemas com "módulos" enormes embutidos, e que precisam ser desacoplados por demandas da cliente (os módulos

precisam ser independentes e passíveis de acoplamento em outras apps).

Gabriel Pessoa 24/08/2020

Sim, atualmente trabalho na Inloco como desenvolvedor backend

VG

Victor Gabryel 24/08/2020

Atualmente trabalho em um projeto como QA.

Wellington Oliveira 24/08/2020

Sim, Projeto pessoal, Também alguns projetos que atuo como Desenvolvedor.

Daniel Silva 24/08/2020

Atualmente faço estagio na area de testes do sistema android em devices da motorola e trabalho como freelancer em desenvolvimento de sites.

MG

Marcos Galvão 24/08/2020

Atualmente faço estágio como desenvolvedor no TRE, trabalhando com os sistemas internos.

GC

Gabriel Cavalcanti de Melo 24/08/2020

Sim, já trabalhei no desenvolvimento de aplicações relacionadas as cadeiras do CIn. E atualmente trabalho no desenvolvimento e manutenção de uma ferramenta interna navegação em dispositivos Android

GC **Gabriel Cavalcanti de Melo** 24/08/2020

interna para navegação*

Quais PONTOS CRÍTICOS podem interferir no sucesso do seu aprendizado?

RE

Ricardo Ebbers Carneiro Leão 24/08/2020

Faltar foco e/ou organização

GD

Gabriel D'Luca 24/08/2020

Gestão do tempo

RJ

Renato Joaquim de Miranda Ferreira 24/08/2020

Falta de foco/engajamento com a cadeira e seu assunto.

RO

Roberto Oliveira 24/08/2020

flexibilidade do tempo.

AR

Antonio Rodrigues 24/08/2020

Conciliar o tempo do estudo com o do trabalho

MG

Marcos Galvão 24/08/2020

Gestão de tempo

LB

Lucas Barros 24/08/2020

Gestão de tempo

HM

Hiro Miyakawa 24/08/2020

Trabalho em equipe (encontrar pessoas para fazer junto) / disciplina para me organizar e entregar as tarefas

LC

Lucas Cardoso 24/08/2020

Motivação, e conciliar o estudo com o trabalho

Marcilio Freitas 24/08/2020

flexibilidade, gestão de tempo e foco.

RP

Ramom Pereira dos Santos Silva 24/08/2020

Desafios semanais é uma motivação muito boa e gestão de tempo pode impactar negativamente no engajamento

Izabella Priscylla da Costa Nascimento 24/08/2020

Gestão de tempo e trabalho em equipe

João Vasconcelos 24/08/2020

tempo

C

EV

Emerson Victor 24/08/2020

Gestão de tempo (trabalho, vida pessoal e estudos)

SG

Saulo Guilhermino 24/08/2020

Gestão de tempo, complexidade das atividades e motivação

Lucca 24/08/2020

Conciliação trabalho + aulas

HC

Heitor Carvalho 24/08/2020

Gestão de tempo, motivação e as demandas paralelas das demais cadeiras de 2020.3

AF

Arthur Frade de Araújo 24/08/2020

Gestão de tempo e trabalho em equipe!

LB

Luan Brito 24/08/2020

Motivação e gestão de tempo

CP

Claudio Pacheco 24/08/2020

Curiosidade, gestão de tempo, motivação

DL

Danilo Lira 24/08/2020

Principalmente gestão do tempo

RM

Rafael Mota Alves 24/08/2020

Gestão de Tempo e Conteudos abordados

VS

victor sena de lima attar 24/08/2020

Discussões que consigam exemplificar temas, conceitos, mostrando casos de uso, pontos positivos, negativos. A interação é importante.

IF

Igor Fernandes 24/08/2020

Assuntos interessantes na área da computação comentados nas aulas, me fazem buscar mais aprender sobre, se for uma tarefa me motiva mais.

IT

Isac Tomaz da Silva 24/08/2020

Motivação, gestão de tempo, e novos aprendizados na área

VG

Victor Gabryel 24/08/2020

Gestão de tempo (estudos e trabalho) e engajamento

JV

Josenildo Vicente 24/08/2020

Distração e procrastinação

TM

Talyta Maria Rosas Pacheco 24/08/2020

Tempo e motivação

Gabriel Pessoa 24/08/2020

Manter o foco (procrastinação) e gestão de tempo

GC

Gabriel Cavalcanti de Melo 24/08/2020

Gestão de tempo

Wellington Oliveira 24/08/2020

Foco, Gestão de Tempo (Trabalho)

Daniel Silva 24/08/2020

Sobrecarga

EM

Elverson Melo 24/08/2020

Adaptação ao novo modelo ensino e gestão do tempo

Qual CENÁRIO espera ver na sua formação depois desta disciplina?

RE

Ricardo Ebbers Carneiro Leão 24/08/2020

Assumir o papel de arquiteto de software num futuro próximo

C

Renato Joaquim de Miranda Ferreira 24/08/2020

Entender como funcionam os microsserviços e de que forma pode ser aplicado no decorrer da minha carreira.

Roberto Oliveira 24/08/2020

Entender arquiteturas feitas em microsserviços

Dairon Eugênio Martins 24/08/2020

Ter capacidade de participar de projetos envolvendo microsserviços.

Hiro Miyakawa 24/08/2020

Entender melhor a parte de desenvolvimento, já que normalmente minha participação nos projetos e no trabalho é de PM/PO.

Lucas Cardoso 24/08/2020

Entender arquiteturas de microsserviços e ser capaz de me envolver em projetos relacionados

Marcos Galvão 24/08/2020

Ter a capacidade de desenvolvedor aplicações utilizando microsserviços

Ramom Pereira dos Santos Silva 24/08/2020

Aprimorar habilidades em Microserviços e implementar com maior propriedade na minha Empresa

Antonio Rodrigues 24/08/2020

Ter um entendimento fundamental de microsserviços para ter a capacidade de aplicar a arquitetura em situações práticas

Arthur Frade de Araújo 24/08/2020

Adquirir um conhecimento mais teórico e formal sobre o tema :)

Lucas Barros 24/08/2020

Entender melhor como funciona e quando aplicar arquitetura de microsserviços

Marcilio Freitas 24/08/2020

Entender como funciona microsserviços e colocar em prática os conhecimentos vistos na disciplina.

Heitor Carvalho 24/08/2020

Saber projetar e/ou manter um projeto que utilize a arquitetura de microsserviços.

Izabella Priscylla da Costa Nascimento 24/08/2020

Saber mais sobre desenvolvimento em questão de obter uma melhor qualidade nos projetos.

Lucca 24/08/2020

Ter um maior discernimento sobre os momentos ideais para se aplicar a arquitetura, assim como ter noção prática do seu uso

Emerson Victor 24/08/2020

Ser capaz de explicar, discutir sobre e implementar microsserviços

Saulo Guilhermino 24/08/2020

Boa compreensão de microsserviços e melhor técnica de implementação

Claudio Pacheco 24/08/2020

Conhecer, na prática, sobre desenvolvimento com base em microsserviços

Luan Brito 24/08/2020

saber como funciona e saber que direção tomar caso precisa usar uma arquitetura em microsserviços

João Vasconcelos 24/08/2020

entender melhor sobre micro serviços e ser capaz de trabalhar na infra de aplicações e produtos

Danilo Lira 24/08/2020

Ser capaz de discutir o conteúdo, entender suas vantagens e desenvolver sistemas utilizando essa abordagem

victor sena de lima attar 24/08/2020

Espero conseguir pontuar os pontos positivos e negativos dessa arquitetura, conseguir discernir como, quando e por que usar.

RM **Rafael Mota Alves** 24/08/2020

Ser capaz de entender em que situações utilizar o padrão e ter conhecimento das técnicas usadas

IF **Igor Fernandes** 24/08/2020

Ter <ais conhecimento sobre mudalizaração de serviços em nuvem

IT **Isac Tomaz da Silva** 24/08/2020

ter um bom conhecimento de base sobre microservices, e conseguir trabalhar de forma produtiva com microservices

VG **Victor Gabryel** 24/08/2020

Entender e compreender todos os pontos/características da arquitetura de microservices e usa-la para desenvolver alguns projetos pessoais.

JV **Josenildo Vicente** 24/08/2020

Ser capaz de entender e implementar projetos em microsserviços

Gabriel Pessoa 24/08/2020

Entender de microserviços, compreender em qual situação utilizar o modelo e conseguir implementar projetos usando esse modelo

TM **Talyta Maria Rosas Pacheco** 24/08/2020

Compreender os conceitos base

GC **Gabriel Cavalcanti de Melo** 24/08/2020

Aprender a desenvolver projetos utilizando microserviços

Wellington Oliveira 24/08/2020

Se atualizar quanto aos conceitos e novidades relacionados a MicroServiços e utilizar no Trabalho de Conclusão de Curso.

Daniel Silva 24/08/2020

Fazer o uso dos conceitos aprendidos em projetos no futuro.

GD **Gabriel D'Luca** 24/08/2020

Possuir um bom entendimento do que exatamente classifica uma arquitetura baseada em microsserviços e como podemos aplicar essa abordagem (ou uma abordagem compatível) trabalhando com modularização em desenvolvimento mobile nativo.

EM **Elverson Melo** 24/08/2020

Conhecer e está apto a utilizar as ferramentas disponíveis para construção de software utilizando microsserviços

◆ Arquitetura Microsserviços

Tópicos [de aprendizagem]

Tópico de [aprendizagem] cujo foco é discutir essa maneira particular de projetar aplicativos de software como conjuntos de serviços implantáveis independentemente

🔗 Referências

Microservices

<https://martinfowler.com/articles/microservices.html>

The Twelve-Factor App

<https://12factor.net/>

Introduction to Microservices, this is a seven-part series of articles

<https://www.nginx.com/blog/introduction-to-microservices/>

🔗 Links compartilhados

AF **Arthur Frade de Araújo**

afa4-reference-monolith-x-microservices

<https://www.n-ix.com/microservices-vs-monolith-which-architecture-best-choice-your-business/>

SG **Saulo Guilhermino**

O que são Microsserviços

<https://aws.amazon.com/pt/microservices/>

Emerson Victor

Livro: Building Microservices: Designing Fine-Grained Systems

https://books.google.com.br/books?hl=pt-BR&lr=&id=jjl4BgAAQBAJ&oi=fnd&pg=PP1&dq=microservices&ots=_APUWrbXmK&sig=rGlej5H-CPn5UVi8w8AjfW26-Bw#v=onepage&q&f=false

Renato Joaquim de Miranda Ferreira

microservices with 12 factor

<https://blog.scottlogic.com/2017/07/17/successful-microservices-with-12factor-app.html>

Gabriel Cavalcanti de Melo

Microservices vs Web Services

<https://blog.dreamfactory.com/microservices-vs-web-services/#Comparing-Microservices-and-Web-Services>

Talyta Maria Rosas Pacheco

What are Microservices? A Complete Introduction for Beginners

<https://www.katalon.com/resources-center/blog/microservices-introduction/>

Lucas Cardoso

Fearless Monolith to Microservices Migration – A guided journey

<https://www.dynatrace.com/news/blog/fearless-monolith-to-microservices-migration-a-guided-journey/>

Gabriel Cavalcanti de Melo

Monolithic vs. Microservices Architecture

<https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59>

Claudio Pacheco

Como migrar um aplicativo monolítico para microserviços no Google Kubernetes Engine

<https://cloud.google.com/solutions/migrating-a-monolithic-app-to-microservices-gke>

Heitor Carvalho

Pattern: Microservice Architecture

<https://microservices.io/patterns/microservices.html>

Marcos Galvão

matg: IBM Microservices point of view guide

https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.ibm.com/downloads/cas/ORNMYNRW&ved=2ahUKEwi6_p3a6LfrAhVmHLkGHSKbCE4QFjASegQICRAB&usg=AQ

Isac Tomaz da Silva

What are Microservices?

<https://aws.amazon.com/microservices/>

Victor Gabryel

Monolithic vs. Microservices Architecture: How the Software Structure Can Influence the Efficiency

<https://magora-systems.com/monolithic-architecture-vs-microservices>

JOSE SHELDON BRITO FEKETE

Padrões de arquitetura Web - Monolítica ou Micro serviços?

<https://medium.com/@lgertel/padr%C3%A3es-de-arquitetura-web-monol%C3%ADtica-ou-micro-servi%C3%A7os-7b3f0c9394fe>

Elverson Melo

What's wrong with the monolith?

<http://www.the-data-wrangler.com/whats-wrong-with-the-monolith/>

Ramom Pereira dos Santos Silva

rpss - Why use a microservices approach to building applications

<https://docs.microsoft.com/pt-br/azure/service-fabric/service-fabric-overview-microservices>

Ramom Pereira dos Santos Silva

DDD e Microserviços

<https://www.infoq.com/br/presentations/ddd-e-microservices-do-negocio-a-arquitetura/>

victor sena de lima attar

The What, Why, and How of a Microservices Architecture

<https://medium.com/hashmapinc/the-what-why-and-how-of-a-microservices-architecture-4179579423a9>

Roberto Oliveira

Arquitetura de microserviços X arquitetura monolítica: 7 diferenças

<https://blog.vinco.com.br/arquitetura-de-microservicos-x-arquitetura-monolitica/>

Roberto Oliveira

Afinal. O Que São Microserviços?

<https://medium.com/@virgs/afinal-o-que-s%C3%A3o-microservi%C3%A7os-c1133e9c8618>

 Roberto Oliveira

Projetar um Aplicativo Baseado em Microserviços
<https://docs.oracle.com/pt-br/solutions/learn-architect-microservice/design-microservices-based-application1.html#GUID-AB1383C6-1A5D-4320-B420-151E69BC45D6>

 Roberto Oliveira

O que são os microserviços?
<https://www.redhat.com/pt-br/topics/microservices/what-are-microservices>

 Izabella Nascimento

what is microservices architecture
<https://medium.com/fintechexplained/what-is-microservices-architecture-1da41a94a29b>

 Danilo Lira

Monolithic vs. Microservices: Why Decoupled and Headless Architectures Are the Future
<https://www.contentstack.com/cms-guides/decoupled-cms/monolithic-vs-microservices-cms-architectures>

 Josenildo Vicente

Micro Serviços: Qual a diferença para a Arquitetura Monolítica?
<https://www.opus-software.com.br/micro-servicos-arquitetura-monolitica/>

 Antonio Rodrigues

Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation
https://www.researchgate.net/profile/Davide_Taibi/publication/319187656_Processes_Motivations_and_Issues_for_Migrating_to_Microservices_Architecture_and-Issues-for-Migrating-to-Microservices-Architectures-An-Empirical-Investigation.pdf

 Hiro Miyakawa

Service-Oriented Architecture vs. Microservices | The Comparison
<https://www.xenonstack.com/insights/service-oriented-architecture-vs-microservices/>

Considera a afirmação: "Se considerarmos os aplicativos monolíticos como um conjunto de subsistemas lógicos abrangidos por um limite físico, os microserviços são um conjunto de subsistemas independentes sem limite físico." Qual a sua opinião? Justifique e encontre um artigo (post) que fundamentam sua resposta e adicione ele nas referências.

 Arthur Frade de Araújo 25/08/2020

Acredito que a proposição é válida e verdadeira, já que a arquitetura de microserviços permite o desacoplamento de infraestrutura. Um serviço pode estar hospedado em um servidor próprio e se comunicar com outro serviço hospedado em um provedor de cloud por exemplo. Por se tratarem de aplicações disjuntas, ambas podem ser publicadas em diferentes plataformas/ máquinas virtuais/ containers. referencia: afa4-reference-monolith-x-microservices

 Arthur Frade de Araújo 25/08/2020

Citação do artigo: "[Strengths of the Monolithic Architecture] Simple to deploy. Another advantage associated with the simplicity of monolithic apps is easier deployment. When it comes to monolithic applications, you do not have to handle many deployments – just one file or directory."

 Saulo Guilhermino 25/08/2020

Concordo com a afirmativa. A utilização de microserviços também agrega diversas vantagens no processo de desenvolvimento e deploy de aplicações, visto que, como cada subsistema é executado de forma independente, uma atualização de funcionalidade do sistema maior pode ser feita sem que todas as partes sejam envolvidas no processo, apenas os subsistemas concernentes. Isto agrega flexibilidade e escalabilidade ao sistema.

 Renato Joaquim de Miranda Ferreira 25/08/2020

Concordo com a proposição. Os sistemas monolíticos tradicionais se dão por meio de um único grande sistema englobando diversas funcionalidades, tornando o processo de escalamento do sistema muito mais difícil e custoso. Utilizando os conceitos de microserviços coisas como o escalamento de um sistema e até mesmo utilização de um de seus blocos em outras aplicações é muito mais viável. referencia:
<https://dzone.com/articles/microservices-architecture-what-when-how>

 Ricardo Ebbers Carneiro Leão 25/08/2020

A principal característica que diferencia uma arquitetura baseada em microserviços de uma monolítica é que microserviços podem ser implantados independentemente um do outro. Embora seja possível fazer deploy de múltiplos microserviços no mesmo meio físico (num mesmo servidor), fazer isso só traria o pior dos dois mundos: tanto o aumento de complexidade quanto a ausência de flexibilidade e escalabilidade. fonte:
<https://devops.com/microservices-vs-monoliths-which-is-right-for-your-enterprise/>

RM

Rafael Mota Alves 25/08/2020

Concordo, pois em uma arquitetura de microserviços os subsistemas são separados como aplicações que podem ser implantadas de forma independente, portanto permitindo que cada um desses subsistemas execute sem estar necessariamente fisicamente juntos, eliminando o chamado limite físico. Já no monolito, toda a aplicação é um sistema único, que pode estar dividido internamente em subsistemas, mas que não operam separados, e portanto há um limite físico.

RM

Rafael Mota Alves 25/08/2020

<https://www.n-ix.com/microservices-vs-monolith-which-architecture-best-choice-your-business/>

JV

Josenildo Vicente 26/08/2020

Concordo, a arquitetura monolítica embora tenha aplicações em modularização, ela é executada em uma única máquina, sendo esse o limite físico. Já a arquitetura de microserviços tem o objetivo de desenvolver sistemas em que cada serviço seja executado em máquinas diferentes de forma independente abrindo a possibilidade de uma manutenção individual de cada serviço sem interrupção no sistema e a escalabilidade mais flexível.

JV

Josenildo Vicente 26/08/2020

referência: <https://www.opus-software.com.br/micro-servicos-arquitetura-monolitica/>

LC

Lucas Cardoso 26/08/2020

Concordo com a proposição. Acredito que a metáfora de tamanho na frase se refere a capacidade de escalabilidade de microserviços. Capacidade que torna o limite de um sistema usando essa arquitetura corretamente virtualmente infinito. Já que pode ser rapidamente escalados em plataformas na nuvem. Referência: <https://www.dynatrace.com/news/blog/fearless-monolith-to-microservices-migration-a-guided-journey/>

LB

Lucas Barros 26/08/2020

Concordo com a afirmação. Na arquitetura monolítica, os subsistemas são deployados de forma unitária e portanto possuem um limite físico, não sendo possível escalar cada subsistema de forma distinta. Na arquitetura de microserviços cada subsistema pode ter seu ciclo de desenvolvimento e operar na sua forma, sendo cada funcionalidade sendo escalada de acordo com suas necessidades. Fonte: <https://www.geeksforgeeks.org/monolithic-vs-microservices-architecture/>

TM

Talyta Maria Rosas Pacheco 26/08/2020

Sim, concordo com a afirmação. O principal objetivo da arquitetura microserviços é desenvolver um sistema complexo por meio da subdivisão de sistemas menores tratados como independentes, os quais são mantidos em repositórios diferentes e ainda distribuídos em diferentes datacenters. Referência: What are Microservices? A Complete Introduction for Beginners

GC

Gabriel Cavalcanti de Melo 26/08/2020

Concordo, pois como no padrão arquitetural de microserviços as pequenas partes que compõe o sistema estão divididas e independentes, eles podem estar sendo executadas em servidores diferentes, em qualquer lugar do mundo, fazendo com que não exista um limite físico. Já na arquitetura monolítica, como a aplicação precisar rodar em conjunto, existe uma limitação de tamanho e complexidade. Referência: Monolithic vs. Microservices Architecture

CP

Claudio Pacheco 26/08/2020

Pelo o que li sobre microserviços, a afirmação caracteriza bem a diferença entre essa arquitetura e a monolítica. Aplicativos em microserviços são compostos por "mini-aplicativos" tão independentes a ponto de poderem ter linguagens, implementação, ciclo de vida e times diferentes. Já o aplicativo monolítico, mesmo com sub-sistemas, é um ente só (por isso a afirmação cita um limite físico de todo esse ente). Microserviços, por sua vez, se comunicam por rede, excluindo as limitações físicas.

CP Claudio Pacheco 26/08/2020

Referência: Como migrar um aplicativo monolítico para microserviços no Google Kubernetes Engine (<https://cloud.google.com/solutions/migrating-a-monolithic-app-to-microservices-gke>)

HC

Heitor Carvalho 26/08/2020

A afirmação faz sentido e condiz com a ideia geral da arquitetura de microserviços. Enquanto na arquitetura monolítica tem-se uma série de subsistemas interconectados entre si em um mesmo espaço físico e algumas vezes fortemente acoplados entre si, a arquitetura de microserviços propõe que o sistema seja composto de vários subsistemas fracamente acoplados e independentes, muitas vezes não limitados por esse espaço físico. Isso permite inclusive que diferentes times trabalhem em cada subsistema

MG

Marcos Galvão 26/08/2020

Concordo, considerando-se duas características principais de diferenciação das duas

arquiteturas. Um sistemas monolítico, composto por subsistemas, é responsável por diversas "business functions", em contra partida um dos conceitos fundamental da divisão da aplicação por serviços é justamente que cada unidade seja responsável por apenas uma "business

function". O segundo fator é o conceito de "Service per Container" utilizado em microservice, onde cada serviço roda individualmente em seu ...

 MG Marcos Galvão 26/08/2020

próprio container, os quais podem, e provavelmente estarão, rodando em lugares fisicamente distintos, diferentemente das aplicações monolíticas que rodam como um monstruoso e único ser em um mesmo local físico.

 MG Marcos Galvão 26/08/2020

referencia: IBM Microservices point of view guide

 IT Isac Tomaz da Silva 26/08/2020

Concordo. Já que define bem a diferença dos 2 estilos, e os microsserviços permitem que os processos sejam divididos com o objetivo de ser resolvidos de forma independente para resolução do da sua repectiva demanda. O que proporciona mensurar de forma mais precisa o custo de cada recurso. Referência: What are Microservices?

 VG Victor Gabryel 26/08/2020

Essa afirmação é correta, pois a arquitetura de microsserviços nasceu justamente dessa limitação da arquitetura monolítica. A princípio, a diferença mais visível entre ambas é que na arquitetura monolítica temos o desenvolvimento de uma estrutura única com várias camadas que se dependem. Na microsserviços temos de diferente a possibilidade de desenvolver pequenos serviços que funcionam de forma independente, podendo cada módulo ser executado de forma autônoma sem necessidade do todo.

 VG Victor Gabryel 26/08/2020

Referencia: Monolithic vs. Microservices Architecture: How the Software Structure Can Influence the Efficiency

 JS JOSE SHELDON BRITO FEKETE 26/08/2020

Concordo sim, pois enquanto aplicativos monolíticos são desenvolvidos seguindo a lógica de uma estrutura que possui várias camadas e componentes que se interligam entre si, com um deploy feito em um mesmo espaço físico. O aplicativo de microsserviços possuem subsistemas responsáveis por funções diferentes dentro do sistema, funcionando de forma autônoma e podem possuir deploy em diferentes máquinas, tornando se independentes.

 RP Ramom Pereira dos Santos Silva 26/08/2020

Concordo plenamente com a afirmação, pois a ideia da arquitetura em Microsserviços é, de fato, construir subsistemas independentes, possivelmente com tecnologias diferentes, bancos de dados diferentes, arquiteturas diferentes (e.g. Arquitetura Hexagonal) e com mecanismos de deployment automático individuais, ou seja, o software é descentralizado, pode escalar partes individuais, apenas. Já os Monolíticos, todo os subsistemas são agrupados em uma única unidade,

 RP Ramom Pereira dos Santos Silva 26/08/2020

utilizando os mesmos recursos computacionais. Outro aspecto importante é o deploy, enquanto nos Monolíticos tudo está em um mesmo processo, para escalar o serviço, é necessário escalar toda a aplicação, já utilizando Microsserviços, cada unidade pode escalar individualmente, inclusive podem executar em máquinas diferentes, Data Centers diferentes ou até mesmo utilizar Ferramentas Serveless (e.g. AWS Lambda).

 RP Ramom Pereira dos Santos Silva 26/08/2020

Fonte: Why use a microservices approach to building applications

 EM Elverson Melo 26/08/2020

A afirmativa é apropriada pois aplicativos monolíticos rodam como apenas um processo, que possui os limites físicos da máquina em que é executado. Já em aplicativos construídos usando microsserviços, cada componente roda como um processo independentemente escalável, e que possui sua própria base de dados. Fonte: What's wrong with the monolith?

 João Vasconcelos 26/08/2020

Concordo. A ideia da arquitetura de microsserviços é separar as partes "independentes" de uma aplicação monolítica. Essa divisão parte da ideia de modularização de código, que é ter pedaços de códigos independentes e responsabilidade única e, da mesma forma que é possível escalar aplicações monolíticas replicando aplicações e balanceando com load balance, a mesma ideia é ainda mais poderosa com microsserviços, que possibilita replicar apenas os serviços sobrecarregados, reduzindo custo.

 João Vasconcelos 26/08/2020

<https://techbeacon.com/app-dev-testing/challenges-scaling-microservices>

 João Vasconcelos 26/08/2020

<https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59>

VS

victor sena de lima attar 26/08/2020

Concordo com a afirmação, sistemas monolíticos agrupam todas as regras da aplicação em apenas uma unidade. Em contrapartida, sistemas baseados em microsserviços se baseiam na componentização das partes do negócio. Uma forma bem conhecida de divisão em subsistemas é definida por Robert C. Martin como Single Responsibility Principle, "Junte todas as coisas que mudem pela mesma razão e separe as coisas que mudem por motivos diferentes".

vs victor sena de lima attar 26/08/2020

Sendo assim, microsserviços foram sendo construídos para que houvesse separação em nível físico desses componentes, auxiliando no seu desenvolvimento como células únicas e auto-sustentáveis. Fonte: The What, Why, and How of a Microservices Architecture

RO

Roberto Oliveira 26/08/2020

Concordo totalmente, de acordo com minhas pesquisas nos sistemas monolíticos as funções são executadas numa mesma máquina, compartilhando processos, recursos de processamento, memória, arquivos etc, já os microsserviços cada parte do sistema deve ficar separado em ambientes diferentes, cada um com seus próprios recursos, podendo se resumir em uma palavra "especialização".

IF

Igor Fernandes 26/08/2020

Concordo. O limite físico que a afirmação coloca é o container/máquinas em que o sistema monolítico está em execução. Um sistema monolítico não é "fragmentado", ou seja, todos as partes do sistema são inseparáveis e fazem parte da mesma base código. Uma pequena correção ou adição de uma funcionalidade necessita que um novo deploy. Diferentemente da arquitetura de microsserviços, onde todas as regras de negócios (business logic) podem ser fragmentados em diferentes componentes

IF Igor Fernandes 26/08/2020

"deployable independently". Exemplo prático desse limite físico: Em momentos de pico de uso da aplicação, toda a aplicação pode sofrer latência na transmissão de dados por se tratar de um único processo lidando com todas as requisições. Numa arquitetura de microsserviços somente as partes da aplicação mais demandas estarão sofrendo com a latência de transmissão de dados.

IN

Izabella Nascimento 26/08/2020

Concordo com a afirmação, como em microsserviços cada serviço é hospedado separadamente, se houver ver algum problema como o de consumo de muitos recursos, é possível esse serviço ser movido para outra máquina e não custar impacto nos outros serviços.

DL

Danilo Lira 26/08/2020

Concordo com a afirmação. Segundo o artigo Monolithic vs. Microservices: Why Decoupled and Headless Architectures Are the Future, os microsserviços são aplicações enxutas e independentes que podem ser implantadas separadamente. Esse último fator é bastante relevante quando pensamos em um limite físico, pois como as aplicações são hospedadas de forma independente, as mesmas podem ser escaladas individualmente sem interferir no sistema como um todo.

Gabriel Pessoa 26/08/2020

Concordo, pois enquanto é possível e natural dividir um monolito em subsistemas lógicos, estão na mesma máquina e portanto dentro do limite físico da máquina, enquanto na arquitetura de microsserviços os subsistemas são independentes e em máquinas diferentes, sem ter o limite físico e assim sendo facilmente escalável Referencia: <http://www.the-data-wrangler.com/whats-wrong-with-the-monolith/>

AR

Antonio Rodrigues 26/08/2020

Concordo. A natureza modular dos microsserviços permite que a complexidade e outras limitações, que eventualmente podem ser físicas, seja reduzida em comparação aos monolitos. Quebrar um sistema em serviços independentes permite a escalabilidade mais simples e fácil do mesmo, diferente de monolitos, que requerem um alto investimento em hardware.

HM

Hiro Miyakawa 26/08/2020

Sim, considerando o servidor onde o código está armazenado para ser executado. No próprio artigo "Introduction to Microservices" disponibilizado pelo professor, diz as características dos serviços monolíticos, e dos baseados em microsserviços que estão ligados "independentes" com APIs com outras partes do serviço.

Quais são as relações entre microsserviços e aplicações Twelve Factor?

LB

Luan Brito 25/08/2020

Metodologia Twelve Factor possibilita a construção de software no padrão de serviços de modo a: minimizar custos, aumentar portabilidade, aumentar compatibilidade com plataformas

na nuvem, aumentar agilidade de desenvolvimento e escalabilidade

TM

Talyta Maria Rosas Pacheco 25/08/2020

Tanto Microserviços quanto Twelve Factor possuem as mesmas finalidades: aumentar agilidade no desenvolvimento, deploy e teste do software, facilitar integração de novos membros no time, independência de linguagens para implantação de diferentes funcionalidades de um sistema, facilitam o processo de manutenção e são tolerantes a erros, já que uma falha em uma funcionalidade não implicará à falha do sistema como um todo.

AF

Arthur Frade de Araújo 25/08/2020

Acredito que o fator IX. Disponability, tem muito haver com esse estilo arquitetural. De acordo com o documento, essa característica define que as aplicações devem ser robustas no sentido de suportar desligamentos ou indisponibilidade de features. Em um sistema monolítico fica muito mais difícil manter a robustez já que toda a lógica de negócio se encontra rodando em uma única máquina. Caso essa máquina seja desligada por um bug, ou problema técnica TODA a aplicação fica indisponível.

GC

Gabriel Cavalcanti de Melo 25/08/2020

Ambos consistem em uma governança descentralizada, pois permitem que as partes do sistema sejam desenvolvidas com diferentes linguagens e de forma independente, o que agiliza o processo de desenvolvimento, testes e deploy, e evitam que um erro inesperado interrompa o funcionamento do sistema.

RJ

Renato Joaquim de Miranda Ferreira 25/08/2020

Ambos possuem os mesmos propósitos, a entrega continua, reusabilidade e boas práticas. De forma que, seguindo as práticas definidas pelo 12 factor, o uso de microserviços se torna ainda mais eficaz para as mais diversas aplicações. referência: microservices with 12 factor.

RE

Ricardo Ebbers Carneiro Leão 25/08/2020

Os doze fatores formam uma diretriz de boas práticas para avaliar e assumir ao construir softwares como serviço escaláveis, declarativos, com contratos claros e com implantação contínua. Tais características se assemelham aos principais benefícios de se implantar uma arquitetura em microserviços e há uma sinergia ao aplicar os twelve factors ao criar sistemas distribuídos.

RM

Rafael Mota Alves 25/08/2020

Algumas características do Twelve Factor App são inerentes da arquitetura de microserviços, como: Processes e Concurrency. Outras facilitam a aplicação de microserviços como: Disposability (Pode ser facilitado pelo modelo arquitetural também, já que os serviços são menores) e Logs (Para facilitar o monitoramento de múltiplos serviços). No entanto existem características que são dificultadas pelo padrão, como a Dev/prod parity

LB

Lucas Barros 25/08/2020

Os doze fatores dessa metodologia podem permitir ao desenvolvedor uma maior facilidade de escalar, declarar, portar, automatizar processos de build e deploy de um serviço. Como um desenvolvedor precisa lidar com vários serviços na arquitetura de microserviços, essa metodologia se torna bastante valiosa.

Gabriel Pessoa 26/08/2020

A metodologia twelve factor traz alguns fatores que deixa muito mais prático e eficiente o uso de microserviços, e vice-versa, uma arquitetura de microserviços já acaba trazendo alguns pontos por sua própria natureza, acredito que pelo fato deles terem o mesmo objetivo, e portanto casam muito bem.

CP

Claudio Pacheco 26/08/2020

Aplicações twelve-factor seguem 12 diretrizes estabelecidas pela Heroku. Essas diretrizes foram organizadas como uma metodologia pela qual os desenvolvedores podem seguir melhores práticas na construção de aplicações na nuvem. Visto que aplicações microserviços estão essencialmente baseadas na nuvem, os doze pontos elencados pela metodologia relacionam-se com as necessidades e desafios trazidos pela arquitetura baseada em microserviços.

HC

Heitor Carvalho 26/08/2020

A metodologia 12-factor prega alguns princípios que também são pontos fortes da arquitetura de microserviços, como: - minimizar tempo e custo de novos desenvolvedores entrando no projeto - deploy em plataformas modernas de cloud - habilitar continuous deployment com facilidade para maximizar agilidade - escalabilidade sem impactos significativos em ferramentas, arquitetura, etc.

MG

Marcos Galvão 26/08/2020

Ambas possuem as mesmas finalidades, entretanto diferentemente de microservices que consiste de diferentes práticas e que podem ser usadas ou não, o twelve-factor determina diretrizes/recomendações que devem ser seguidas em conjunto para a produção de software como um serviço. Ademais elas buscam prover escalabilidade de operação e desenvolvimento, uma alta robustez durante a execução e grande flexibilidade para integrar-se com novas

VG

Victor Gabryel 26/08/2020

Basicamente, Twelve Factor é uma metologia que abrange um conjunto de boas práticas para que sirva de suporte para o desenvolvimento de aplicativo de software que é baseado em serviços e que tenha um crescimento "saudável". Como os microserviços e o Twelve Factor possuem características idênticas como a escalabilidade, a implantação da aplicação para modelo em nuvem, portabilidade e implantação continua faz o desenvolvimento dos microserviços necessitar dessas boas práticas como orientação.

IT

Isac Tomaz da Silva 26/08/2020

Facilitar as etapas no desenvolvimento de software, reduzir a complexidade dos projetos, e também diminuir a dificuldade inicial que novos colaboradores do projeto possam ter.

JS

José Sheldon Brito Fekete 26/08/2020

Twelve factors ou Doze fatores é um conjunto de boas práticas que auxiliam no desenvolvimento de aplicações que rodam na nuvem, tornando assim mais agradável a utilização dessas práticas em desenvolvimento de aplicativos microservices, criando um ambiente de desenvolvimento e a busca de erros mais fáceis e rápidos.

RP

Ramon Pereira dos Santos Silva 26/08/2020

Aplicações Twelve Factor seguem uma metodologia com 12 princípios para construção de SaaS (Software as a Service), onde são utilizadas várias abordagens de automação de processos, escala de serviços, deployments automáticos, e claro, deployment em plataformas de Cloud (AWS, GCP, Azure). Microserviços têm um relacionamento muito próximo disso, companhias que utilizam essa arquitetura fazem diversos deployments por dia (Mercado Livre fazem 3k), e padronizar o desenvolvimento é um requisito interessante.

Ramon Pereira dos Santos Silva 26/08/2020

para trabalhar com Microserviços, além da evolução do Software futuramente ser facilitada, visto que cada Microserviço tem suas responsabilidades bem definidas.

EM

Elverson Melo 26/08/2020

Ambos são adequados a implantação em nuvem, permitem implantação contínua, são facilmente escaláveis e independentes de linguagem de programação.

RO

Roberto Oliveira 26/08/2020

Em ambos existe a separação de responsabilidade que resulta em portabilidade, a autonomia também permite que existam menos falhas no ecossistema inteiro.

João Vasconcelos 26/08/2020

Inicialmente o manifesto Twelve Factor Apps foi projetado para aplicações na nuvem afim de ser um conjunto de boas práticas. Aplicações em microserviços são bastante semelhantes, fazendo bastante sentido utilizar essas práticas na hora de desenvolver uma.

João Vasconcelos 26/08/2020

<https://imasters.com.br/desenvolvimento/twelve-factor-app-boas-praticas-para-microservicos> (português)

João Vasconcelos 26/08/2020

https://dev.to/simon_sugob/the-twelve-factor-app-a-successful-microservices-guideline-3a1h

IF

Igor Fernandes 26/08/2020

O Twelve-Factor propõe boas práticas de desenvolvimento de software, práticas essas que podem ser seguidas numa arquitetura de microserviços. Dois de seus fatores que considero fortemente ligados à arquitetura de microserviços são: "VI. Processos" e "IX. Descartabilidade". Eles propõem a disponibilidade da aplicação é muitos processos "stateless", "share-nothing" e facilmente descartáveis.

Igor Fernandes 26/08/2020

Ou seja, componentes independentes capazes de fazer seu papel na aplicação, sem afetar qualquer outra parte do software durante seu funcionamento ou desligamento. Além disso reforçam a importância da escalabilidade do sistema, assim que como a arquitetura de microserviços.

DL

Danilo Lira 26/08/2020

Há uma relação quando falamos do fator IX, serviços de apoio, pois para a metodologia esses serviços são vistos como partes anexas e não como partes integrantes da aplicação, por isso podem ser trocadas a qualquer momento sem precisar de mudanças no código, mas sim nas configurações.

IN

Izabella Nascimento 26/08/2020

Como o The Twelve Factor foi criado para ser um guia de boas práticas para serem utilizadas ao criar aplicações voltadas para nuvem, essas práticas são importantes a serem seguidas na construção de uma aplicação com arquitetura de microserviços onde se tem muitos serviços

VS

victor sena de lima attar 26/08/2020

O The Twelve Factor é um conjunto de 12 boas práticas de desenvolvimento de softwares SaaS, onde os microserviços se encaixam, dentre as regras temos a XI - processos, nela descreve que a aplicação deve ser dividida em processos (ou serviços), e que os mesmos não devem guardar estado, para isso devem guardar os dados em bancos, ou seja, é um padrão de boas práticas que se encaixam muito bem com a arquitetura de microserviços.

JV

Josenildo Vicente 26/08/2020

A metodologia 12factor são doze 'pontos' de boas práticas que uma aplicação SaaS deve seguir para poder ter uma escalabilidade orgânica ao longo do tempo, e, os microserviços também têm esse objetivo de ter uma escalabilidade fácil com cada serviço sendo feito de forma independente. Os 12 fatores são importantes na base de uma implementação em microserviços para evitar problemas futuros.

AR

Antonio Rodrigues 26/08/2020

Além de ser um conjunto de orientações e práticas para o desenvolvimento de projetos no geral, alguns pontos concordam diretamente com os princípios de microserviços, em especial o sexto, sobre processos assíncronos e a importância das aplicações serem stateless. Acho que alguns combinam mais se forem um pouco alterados (por ex, o primeiro pode ser lido como "Uma codebase para cada serviço").

HM

Hiro Miyakawa 26/08/2020

12 factor são boas práticas / metodologia para desenvolvimento de apps na nuvem como saas. Há uma similaridade nos valores em deixar independentes (microservices) e menos "robusto" (monolith). Haja vista VI. Processes "Twelve-factor processes are stateless and share-nothing." e X. Dev/prod parity que bebe indiretamente da filosofia da independência.

Lucca 26/08/2020

A Arquitetura de microserviços tem como um dos seus pilares o uso de ampla e completa infraestrutura externa (à aplicação em si), como cloud e CI/CD, para que o desenvolvimento seguro e escalável de diversos serviços seja viável e funcione de maneira de maneira fluída, sem que os desenvolvedores sintam os gargalos das etapas de testes, deploy e integração inerentes a qualquer projeto.

Lucca 26/08/2020

Aplicações Twelve Factor partem dessa premissa de que o desenvolvimento deve estar cercado por um alto nível de automação e abstração, baseado em soluções de nuvem, permitindo que os desenvolvedores otimizem o tempo de exploração e execução no projeto. Outro ponto em que ambos se assemelham é o design voltado a escalabilidade, fator essencial para que essas soluções possam ser idealizadas inicialmente de uma maneira mais simples, evoluindo e se expandindo frente

Lucca 26/08/2020

à novas com complementares necessidades. A clareza nas interfaces também é um ponto comum em ambos, dado que o desenho e encapsulamento de serviços (lógica, interface etc) por meio de API's REST traz abstração e isolamento para diversos e diferentes contextos possíveis em uma aplicação.

Quais são os relacionamentos com outros estilos de arquitetura de software?

LC

Lucas Cardoso 25/08/2020

Existem pessoas que colocam microserviços como um subset ou até um outro nome de Service Oriented Architecture (SOA). Nos artigos de referência, Fowler e Lewis argumentam que SOA é uma definição que acaba agrupando muitos estilos e práticas, por isso parte dos defensores de microserviços negam esse rótulo

EV

Emerson Victor 25/08/2020

De acordo com o livro Building Microservices as maiores vantagens de uma arquitetura baseada em microserviços vêm da sua granularidade e poder resolver problemas de diversas formas. No entanto, existem outras técnicas que podem ser usadas para tornar um software mais granular, como bibliotecas compartilhadas e módulos. E, assim como microserviços, essas técnicas estão sujeitas a possuírem desvantagens, por isso é necessário observar cada caso e entender qual a melhor solução para cada situação.

AF

Arthur Frade de Araújo 25/08/2020

Por se tratar de uma estrutura mais fragmentada, a arquitetura de microserviços permite que cada serviço possua seu próprio estilo arquitetural. Por exemplo: em um processo de migração gradual de arquitetura, novas funcionalidades podem ser desenvolvidas em pequenos serviços enquanto a versão inicial do projeto (monolito por exemplo) ainda existe. Nesse caso, as coisas não se anulam, a arquitetura começa a mudar na medida em que alguns "serviços" ainda são monólicos.

C

RE

Ricardo Ebbers Carneiro Leão 25/08/2020

Existem dois pitfalls relacionados a "tamanho" ao desenhar uma arquitetura orientada a microsserviços: criar um conjunto descentralizado de "Microlitos", cada um com várias responsabilidades nebulosas e muitas vezes reinventando a roda, ou microsserviços anêmicos, que não representam capacidades de negócio bem definidas. Para evitar situações como essa o padrão arquitetural de Domain Driven Design é crucial para a definição de responsabilidades e fronteiras dos microsserviços a implantar.

GC

Gabriel Cavalcanti de Melo 25/08/2020

Como Microsserviços são pequenas aplicações independentes que executam tarefas específicas, elas podem ser integradas em outra aplicações. Essa integração pode ser feita por meio de Web Services, que conectam Microsserviços com outras aplicações de diferentes estilos arquiteturais, formando uma aplicação maior. Referência: Microservices vs Web Services

LB

Luan Brito 26/08/2020

Acredito que seja preferível usar outra metodologia se a necessidade for uma arquitetura no estilo monólito, mas ainda assim é possível aproveitar algumas características no twelve factor

LB

Lucas Barros 26/08/2020

O estilo arquitetural microsserviços é muitas vezes confundido com o estilo arquitetural SOA (Service-oriented Architecture). Sua principal diferença é o escopo: enquanto microsserviços é referente a uma aplicação, SOA é referente à empresa como um todo (conjunto de aplicações). referência: <https://www.ibm.com/blogs/cloud-computing/2018/09/06/soa-versus-microservices/>

Gabriel Pessoa 26/08/2020

Uma arquitetura muito comparada com a de microsserviços eh a arquitetura service-oriented, pois elas tem o mesmo propósito e portanto tem bastantes semelhanças, com sua principal diferença sendo na comunicação entre os serviços, pois na arquitetura SOA os serviços geralmente dividem o mesmo componente de comunicação, enquanto que na de microsserviços ela é feita de maneira independente por chamadas na API, e portanto os serviços acabam ficando mais independentes e sem um unico ponto de falha.

TM

Talyta Maria Rosas Pacheco 26/08/2020

Como comentado no artigo de referência, Microservices, muitos acabam entendendo microsserviços como um outro nome para a arquitetura mais antiga, SOA, já que os dois são orientados a serviços. Porém, SOA tem muitos padrões definidos, o que acaba limitando na escolha das linguagens no desenvolvimento dos serviços, já no microserviços a equipe tem mais autonomia na escolha das ferramentas necessárias para a implementação <https://www.luiztools.com.br/post/microservices-vs-soa-entenda-as-diferencias/>

HC

Heitor Carvalho 26/08/2020

Apesar da diferença estrutural, a ideia das aplicações de microsserviços podem seguir as mesmas ideias, boas práticas e padrões de projeto de arquiteturas monolíticas (com algumas adaptações). Tal arquitetura não é algo díspar de uma monolítica, mas sim uma outra forma de organização com vantagens e benefícios. No artigo "Microservices - a definition of this new architectural term" de Martin Fowler, ele recomenda que se comece com sistema monolítico para depois aplicá-la gradualmente.

CP

Claudio Pacheco 26/08/2020

Como citado em dois dos artigos de referência, a arquitetura microsserviços é comumente associada à Arquitetura Orientada a Serviços (SOA) devido às diversas similaridades entre ambas (as duas são orientadas a serviços). Entretanto, há algumas diferenças principais, como dados (SOA compartilha um banco, enquanto que cada microsserviço possui o próprio) e comunicação (SOA utiliza ESB, enquanto microsserviços usam protocolos agnósticos).

Claudio Pacheco 26/08/2020

Leitura principal: <https://www.tiempodev.com/blog/microservices-vs-soa/>

MG

Marcos Galvão 26/08/2020

Pela ideia de microservices ser justamente os serviços independentes e interoperantes, responsáveis por apenas uma "business function", a arquitetura se torna muito flexível para trabalhar inclusive com sistemas, ou parte dele, que foram construídas em outras arquiteturas, assim como é comentado na referência "Microservices", quando o autor fala sobre o The Guardian, onde sua estrutura principal, o site, é um grande monólito, entretanto novas funcionalidades são desenvolvidas em ...

Marcos Galvão 26/08/2020

microservices devido a melhor escalabilidade, rapidez de produção e, caso necessário, desligamento da mesma

RP

Ramom Pereira dos Santos Silva 26/08/2020

Antes da arquitetura baseada em Microserviços ficar tão em alta no mercado, a arquitetura orientada a serviços (SOA) era bastante utilizada, antes dessa o RUP também já foi utilizada com bastante frequência. Microserviços tem muita relação com SOA, dado que ela trouxe a ideia de decompor o Software em vários serviços que comunicavam entre si. porém ainda com

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Isso só descreve o conceito em várias versões que conseguem unir os sistemas em uma ideia uma unidade lógica, o que difere da arquitetura em Microserviços, porém, vale salientar, ainda não existe um consenso

 **Ramom Pereira dos Santos Silva** 26/08/2020

em como construir e definir os limites de cada Microserviço, muitos utilizando algumas metodologias para isso, como o DDD (Domain Driven Design).

 **Ramom Pereira dos Santos Silva** 26/08/2020

<https://www.infoq.com/br/presentations/ddd-e-microservices-do-negocio-a-arquitetura/>

 **Isac Tomaz da Silva** 26/08/2020

Possui alguns aspectos semelhantes com SOA(o que deixa algumas pessoas confusas se não seria outra nomenclatura), mas possui divergências. Relacionado a monolítico possui uma relação de migração, quando se percebe que continuar com a abordagem monolítica é algo inviável, fazendo essa mudança de forma gradual.

 **Victor Gabryel** 26/08/2020

Os microsserviços tem uma grande similaridade com o SOA, onde o conceito das duas arquiteturas são bem semelhantes. É possível notar que devido a essas semelhanças, existe um adoção de que as os microsserviços seria um novo nome para o SOA, devido sua recente popularização. É possível checar na referencia abaixo os gráficos de pesquisa das duas arquiteturas nos últimos anos, mostrando essa relação das duas arquiteturas. Porém, nos microsserviços é evitado o uso do ESB, diferentemente do SOA, +

 **Victor Gabryel** 26/08/2020

+ além de existir uma maior complexidade no uso de um padrão de comunicação entre os serviços. referencia dos gráficos:

<https://www.infoq.com/br/news/2017/08/soaandmicroservices/>

João Vasconcelos 26/08/2020

Pesquisando um pouco podemos ver que microsserviços se assemelha bastante com SOA, porém a maior diferença está na forma em que os serviços são disponibilizados. SOA se aproxima mais da ideia de várias aplicações monolíticas se comunicando através de um ESB.

João Vasconcelos 26/08/2020

<https://searchapparchitecture.techtarget.com/definition/Enterprise-Service-Bus-ESB>

João Vasconcelos 26/08/2020

<https://www.luiztools.com.br/post/microservices-vs-soa-entenda-as-diferencias/>
(português)

 **Roberto Oliveira** 26/08/2020

A arquitetura orientada a serviços como já mencionada, tem o objetivo mais parecido com o microservices. Ajustar, testar e criar os serviços ao mesmo tempo, porém ainda assim existe uma estrutura monolítica.

 **Igor Fernandes** 26/08/2020

Muitas padronizações de desenvolvimento de software querem melhorar a mantinabilidade do sistema. Uma das formas mais propostas para resolver esse problema é a componentização. A componentização torna o código mais enxuto para manutenção. Porém há várias formas de fazer componentização, como proposto pelo SOA: cada recurso/componente do sistema pode ser uma rota da API.

 **Igor Fernandes** 26/08/2020

Ou como o microsserviços: cada componente do sistema deve ser rodado isoladamente das outras partes da aplicação (em processos diferentes).

 **Igor Fernandes** 26/08/2020

Ou como o MVC: Os componentes do sistema são três Model, View e Controller.

 **José Sheldon Brito Fekete** 26/08/2020

A arquiteta de microsserviços é bastante semelhante ao SOA e muitas vezes confundidas, embora o SOA possua diversos significados e até contraditórios para diferentes profissionais. Uma das semelhanças é a separação de serviços autônomos. Porém enquanto o SOA ainda utiliza a estrutura monolítica, se comunicando utilizando o ESB, microsserviços utilizam endpoints inteligentes com protocolos burros.

 **Izabella Nascimento** 26/08/2020

Microsserviços são semelhantes a arquitetura orientada a serviço, falado até algumas vezes como evolução do SOA, tendo como algumas diferenças cada microsserviço possuir seu próprio bando de dados, além de utilizam chamadas com a implementação do REST tornando as chamadas mais simplificadas.

 **Danilo Lira** 26/08/2020

Como citado na referência do NGINX, a arquitetura de microsserviços se parece bastante com o SOA, pois as duas se baseiam na utilização de um conjunto de serviços. Entretanto há uma diferença entre os dois estilos, já que o SOA é dividido em processos de negócios e os

microsserviços em funções básicas.

 VS victor sena de lima attar 26/08/2020

Microsserviços são comparados a outros estilos de arquitetura de software como monolítico, sendo seu oposto em vários pontos e com o SOA, sendo um modelo bem semelhante, dito até como o predecessor do microsserviços. Diferenças básicas entre os dois: Banco de dados é compartilhado entre todos os serviços, no microsserviços, cada serviço tem o seu. Necessitava de um ESB, já no microsserviços, utiliza-se o padrão rest, entre outras diferenças.

 JV Josenildo Vicente 26/08/2020

Como cada micro serviço é implementado e modificado de forma independente para ser responsável por apenas uma função, os microsserviços podem ser integrados em outras aplicações de outras arquiteturas sem prejudicá-las como por exemplo num sistema monolítico ter novas funcionalidades em microsserviços.

 HM Hiro Miyakawa 26/08/2020

Me lembrou a o Service-Oriented Architecture (SOA). Segundo o texto "Service-Oriented Architecture vs. Microservices | The Comparison", as duas estão na moda por ter "Independently deployable and consist of a set of dissociated services" e consequentemente ser mais robusto, persistente e poder evoluir.

 AR Antonio Rodrigues 26/08/2020

Como visto nas referências, a arquitetura de microsserviços pode parecer similar a SOA, porém é diferenciada pelo seu foco em ESBs usadas em aplicações monolíticas.

 RJ Renato Joaquim de Miranda Ferreira 26/08/2020

Tem uma grande semelhança com a SOA porém se difere no ponto que a SOA é mais predominantemente monolítica enquanto os microsserviços não seguem o mesmo propósito.

Não há um consenso quanto uma definição precisa desse estilo arquitetônico denominado microservices (do português, microsserviço), mas existem certas características comuns em torno da organização em torno da capacidade de negócios, implantação automatizada, inteligência nos terminais e controle descentralizado de linguagens e dados. Reflita sobre o que vocês estudou e escreva a SUA definição para o que é o estilo arquitetural microsserviços.

 AF Arthur Frade de Araújo 25/08/2020

A arquitetura de microsserviços pode ser definida como um conjunto de aplicações disjuntas, independentes de plataforma, que possuem a capacidade de comunicarem-se entre si através de um protocolo comum, compreendido por ambos os sistemas comunicantes, e que atuam juntos em prol de um objetivo maior.

 EV Emerson Victor 25/08/2020

Uma arquitetura que tem como base a divisão de um software em serviços pequenos e autônomos que trabalham juntos

 TM Talyta Maria Rosas Pacheco 25/08/2020

É uma alternativa para o desenvolvimento de sistemas complexos, cujo o principal objetivo é subdividir o sistema maior em sistemas menores independentes e que se comunicam entre si.

 RM Rafael Mota Alves 25/08/2020

O padrão arquitetural de microsserviços seria um arquitetura em que a uma aplicação é dividida em vários subsistemas de forma que esses subsistemas podem ser desenvolvidos e implantados de forma independente.

 RE Ricardo Ebbers Carneiro Leão 25/08/2020

Organizações suficientemente complexas, assim como sistemas orgânicos, possuem subsistemas com funções claramente definidas, que crescem ou diminuem em importância de forma independentes e demandam recursos nos mais variados e ritmos. Dado que hoje temos capacidades físicas e lógicas suficientes para desenvolver sistemas tão flexíveis quanto as organizações que eles modelam, a arquitetura de microsserviços é a forma que mais aproxima o modelo de software das demandas de negócios.

 LC Lucas Cardoso 25/08/2020

Um padrão arquitetural que consiste na divisão da complexidade de um sistema em pequenos módulos independentes, que se comunicam entre si

 LB Luan Brito 26/08/2020

Um padrão de arquitetura que permite um alto nível de modularidade dos serviços, garantindo: reuso, escalabilidade e melhor gerencia da aplicação. Padrão possibilitado pelo caminho que a rede vem tomando, com servidores potentes e avançados protocolos de comunicação.

 LB Lucas Barros 26/08/2020

O estilo arquitetural microsserviços seria, para mim, uma forma de diminuir a complexidade atribuída apenas a um subsistema, e dividir a para vários subsistemas, permitindo desenvolvimento e escalabilidade independente.

Gabriel Pessoa 26/08/2020

Uma arquitetura que consiste do software ser dividido em vários serviços essenciais que funcionam de forma independente e se comunicam entre si.

GC

Gabriel Cavalcanti de Melo 26/08/2020

Um padrão arquitetural que tem como principal característica a divisão da aplicação em pequenos serviços independentes, que executam tarefas ou funcionalidades específicas. Agilizando os processos e garantindo que uma falha em determinado serviço, não impacte o funcionamento do resto do sistema.

MG

Marcos Galvão 26/08/2020

Padrão arquitetural que visa reduzir a complexidade alocada, nos monólitos, em uma única aplicação em serviços interoperantes, possibilitando maior flexibilidade no desenvolvimento, deploy e manutenção; permitindo que modificações sejam feitas individualmente nos serviços que, em conjunto, compõem a aplicação. De tal forma ações como escalabilidade, testes contínuos e atualizações mais frequentes são facilitadas devido ao nível de independência de cada serviço para com o resto da aplicação.

HC

Heitor Carvalho 26/08/2020

O estilo arquitetural de microsserviços pode ser visto como um caminho a qual se pode levar um sistema monolítico de forma a deixá-lo mais fragmentado e distribuído, permitindo inclusive que serviços diferentes sejam escritos em linguagens diferentes, estejam em máquinas diferentes e se comuniquem por protocolos simples. Isso traz benefícios e malefícios ao sistema, que devem antes ser avaliados com base no contexto para verificar se tal arquitetura é a melhor opção a se considerar.

CP

Claudio Pacheco 26/08/2020

A arquitetura em microsserviços está baseada na ideia de criar uma aplicação formada por microsserviços independentes entre si. O nível de independência chega a permitir que cada microsserviço tenha características próprias (linguagem, banco, ciclo de desenvolvimento etc), e que sejam responsáveis, cada um, por um papel/tarefa específica. A principal preocupação quanto ao sistema como um todo é estabelecer a comunicação do microsserviço com os demais através da internet.

VG

Victor Gabryel 26/08/2020

O estilo arquitetural microsserviços tem como o desenvolvimento de aplicativo de software para que ele se componha de pequenos módulos(serviços) que funcionem de forma independente, podendo eles serem desenvolvidos com diferentes tecnologias, onde se é necessário uma padrão comunicação entre esses serviços tendo como uma maior flexibilidade e a implantação continua.

IT

Isac Tomaz da Silva 26/08/2020

É um estilo que preza pelo desenvolvimento de software de maneira sustentável e escalável, resolvendo o(s) problema(s)/demanda de uma forma independente, permitindo que cada necessidade tenha sua solução específica, ou seja, com somente aquilo que está no contexto de sua respectiva necessidade. Que por sua vez resulta numa entrada facilitada de novo desenvolvedores em projetos que usufruem dessa metodologia.

JS

José Sheldon Brito Fekete 26/08/2020

Um estilo arquitetural que propõe que o software seja subdividido em microsserviços independentes entre si, que se comunicam, assim executando todas as funções do sistema, deixando o mesmo mais flexível e escalável.

RP

Ramom Pereira dos Santos Silva 26/08/2020

Estilo arquitetural de desenvolvimento de aplicações compostas por diversos serviços independentes, fracamente acoplados e com alta coesão, com foco no domínio do problema a ser resolvido, possuindo mecanismos de deploy, linguagens e bancos de dados diferentes, que se comunicam entre si via protocolos bem definidos.

EM

Elverson Melo 26/08/2020

Microsserviços compõe um padrão arquitetural onde softwares são decompostos em componentes menores e separados que rodam em nuvem e se comunicam por APIs. Cada componente pode escalar de uma forma diferente, assim como possui um banco de dados independente. Como são componentes distintos e que se comunicam por APIs, eles podem ser implementados em diferentes linguagens e por times distintos. Em geral os times, para aplicativos como microsserviços, são menores que o normal e ...

Elverson Melo 26/08/2020

possuem profissionais com experiências diversificadas pelas camadas do software. Embora seja mais complexo nos primeiros estágios do desenvolvimento de uma aplicação, esse estilo evita problemas que surgem quando um aplicativo monolítico se torna extremamente grande e complexo.

C

**Igor Fernandes** 26/08/2020

Arquitetura de microserviços pode ser resumida em três pontos fundamentais: 1 - Componentização: Sistemas divididos em múltiplos serviços permitem que a aplicação seja menos dependente e portanto não necessitam do funcionamento das outras partes da aplicação para realizar seu papel.

IF Igor Fernandes 26/08/2020

2 - Escalabilidade: Serviços independentes que são "share-nothing" e "stateless", e portanto não necessitam estar alojados na mesma máquina física. Para aumentar a disponibilidade em outra parte do mundo o serviço pode ser iniciado em outra máquina física com nenhuma dificuldade. 3 - Deploy rápido: Serviços independentes não necessitam do rebuild e teste da aplicação inteira para realizar o deploy de uma nova funcionalidade.

João Vasconcelos 26/08/2020

Microsserviço é um padrão arquitetural que busca modularizar toda a aplicação, fazendo com que o time possa se adaptar da melhor forma: 1 - O time responsável por cada serviço é livre para decidir qual a melhor linguagem, abordagens a serem utilizadas, deploys, etc. Isso incrementa muito a velocidade de desenvolvimento 2 - É mais escalável, sendo possível escalar apenas os serviços mais "requisitados".

**Danilo Lira** 26/08/2020

O estilo arquitetural de microsserviços é a aplicação do conceito dividir para conquistar em uma aplicação, ou seja, é quebrar uma aplicação em várias partes menores que sejam capazes de resolver um problema cada. Essas pequenas partes devem ser independentes e se comunicar para desempenhar uma função maior

**Izabella Nascimento** 26/08/2020

Microsserviços é uma arquitetura que onde os serviços são separados de forma que se tornam independentes podendo serem monitorados, implantados, testados sem interferir no serviço como um todo.

**victor sena de lima attar** 26/08/2020

A arquitetura de microsserviços é um padrão que busca componentizar aplicações em pequenos serviços que são Independentes e que se comunicam entre si. Esse tipo de organização dá mais flexibilidade a empresas para montar times menores para cada serviço obtendo maior produtividade, serviços menores mais organizados possibilitando deploys mais rápidos e mais constantes.

**Roberto Oliveira** 26/08/2020

é uma arquitetura que se divide em componentes e que funcionam de forma de independentes, evitando que um erro comprometa toda a estrutura. Otimizando o desenvolvimento e a escalabilidade do negócio

**Hiro Miyakawa** 26/08/2020

É uma arquitetura de software, qual divide o serviço em pequenas partes que são implantadas de forma independentes e comunicam entre si, viabilizando o desenvolvimento do serviço de maneira modular e a escalabilidade do negócio sem comprometer o todo.

**Antonio Rodrigues** 26/08/2020

Uma arquitetura que promove a divisão do sistema em partes assíncronas e independentes, permitindo flexibilidade em alterações e na manutenção do sistema.

**Josenildo Vicente** 26/08/2020

Microsserviços é o conjunto de serviços que são implantados de forma independentes em ambientes diferentes se comunicando com mecanismos leves formando uma aplicação que seja mais flexível, escalável e com uma manutenção simples sem dependência entre os serviços.

**Renato Joaquim de Miranda Ferreira** 26/08/2020

Microsserviços é um estudo de arquitetura de software que se utiliza da independência entre suas funcionalidades, modularizando o sistema em várias micro-aplicações, por assim dizer, para os mais diversos usos nos mais diversos sistemas desde que haja uma forma de comunicação comum.

Dairon Eugênio Martins 26/08/2020

É uma arquitetura dividida em várias aplicações independentes de tecnologia e ou ambientes que "conversão" entre si.

JORNADA PARA MICROSERVICE

Tópicos [de aprendizagem]

Migrar para um arquitetura de microservices pode ser uma estratégia de sucesso para sua aplicação ou pode ser um suicídio. Como escolher o caminho certo?

🔗 Referências

A pattern language for microservices

<https://microservices.io/patterns/index.html>

The cloud native application: Microservices with Spring Boot and Spring Cloud

<https://www.infoq.com/br/presentations/the-cloud-native-application-microservices-with-spring-boot-and-spring-cloud/>

🔗 Links compartilhados

 Emerson Victor

Adopting an API-first approach

<https://swagger.io/resources/articles/adopting-an-api-first-approach/>

 Emerson Victor

O que é API e qual sua importância para o cenário digital?

<https://inteligencia.rockcontent.com/o-que-e-api/>

 Saulo Guilhermino

The Facebook/Cambridge Analytica Data Scandal, Visually Explained

https://overthinkgroup.com/facebook-cambridge-analytica/?utm_campaign=Submission&utm_medium=Community&utm_source=GrowthHackers.com

 Saulo Guilhermino

What Does An API Gateway Do

<https://www.redhat.com/pt-br/topics/api/what-does-an-api-gateway-do>

 Rafael Mota Alves

Uma abordagem de desenvolvimento API-First

<https://blog.mandic.com.br/artigos/uma-abordagem-de-desenvolvimento-api-first/>

 Rafael Mota Alves

Equipe sua API com a melhor armadura

<https://sensedia.com/api/equipe-sua-api-com-a-melhor-armadura/>

 Rafael Mota Alves

AWS API Gateway

<https://aws.amazon.com/pt/api-gateway/>

 Danilo Lira

API pública, privada e entre parceiros: qual a diferença?

<https://vertigo.com.br/apis-publicas-privadas-e-entre-parceiros-qual-a-diferenca/>

 Igor Fernandes

REST vs RESTful

<https://blog.ndepend.com/rest-vs-restful/>

 Arthur Frade de Araújo

YAGNI Concept

<https://www.martinfowler.com/bliki/Yagni.html>

 Arthur Frade de Araújo

SOLID Pattern

<https://pt.wikipedia.org/wiki/SOLID>

 Lucas Cardoso

What is wrong with API-first Microservices?

<https://blogs.sap.com/2018/02/21/what-is-wrong-with-api-first-microservices/>

 Marcos Galvão

An API-First Development Approach

<https://dzone.com/articles/an-api-first-development-approach-1>

 Marcos Galvão

API Gateway

<https://microservices.io/patterns/apigateway.html>

 Renato Joaquim de Miranda Ferreira

API Integrations: 5 Ways they can benefit your business

<https://mydatascope.com/blog/en/api-integrations-5-ways-they-can-benefit-your-business/>

 Izabella Nascimento

API gateway com microsserviços

<https://www.zup.com.br/blog/api-gateway-com-microsservicos>

 José Sheldon Brito Fekete

API-First: Uma abordagem de desenvolvimento

<https://blog.mandic.com.br/artigos/uma-abordagem-de-desenvolvimento-api-first/>

Ramom Pereira dos Santos Silva

Qual é a função de um gateway de API?

<https://www.redhat.com/pt-br/topics/api/what-does-an-api-gateway-do>

Hiro Miyakawa

Web API Design | Azure

<https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design>

Qual o valor que as API's trouxeram para os negócios digitais hoje em dia? O que significa e qual a relevância do COMPROMISSO COM API's ou o termo API FIRST?

Emerson Victor 27/08/2020

API's facilitam tanto no aspecto de desenvolvimento, por manter funcionalidades em um único local, independente de outros sistemas e mais fácil de realizar manutenção, quanto no aspecto de mercado, por facilitar a integração entre diversas plataformas e empresas. Exemplos disso são: localização geográfica no próprio site do estabelecimento e login em uma plataforma com contas de outras plataformas.

Emerson Victor 27/08/2020

API first é uma abordagem onde um negócio cria primeiramente uma API com as suas necessidades e em seguida desenvolve o seu produto com base dessa API. Essa abordagem traz diversos benefícios, como times podendo trabalhar em paralelo, redução do custo de desenvolvimento de apps, novas possibilidades de negócios e modelos de receitas baseados no consumo de serviços, adicionar novos serviços sem precisar refazer todo o sistema, entre outros.

Vinicius Garcia 31/08/2020

quais os maiores desafios nessa abordagem?

Saulo Guilhermino 27/08/2020

APIs também possibilitaram o avanço da cultura de Inovação: Um desenvolvedor, através de uma API, consegue interagir com uma infraestrutura pronta e focar mais esforços na manipulação e exibição das informações obtidas por meio das requisições de API

Saulo Guilhermino 27/08/2020

Mas isto também pode representar um risco de segurança à empresa caso os acessos permitidos através da API não sejam bem controlados e monitorados. Um enorme problema relacionado aconteceu com o Facebook no caso Cambridge Analytica e sua campanha durante as eleições presidenciais dos Estados Unidos em 2016

Vinicius Garcia 31/08/2020

o que você acha que faltou no processo do uso da API do Facebook pela CA?

Luan Brito 27/08/2020

Acredito que a abordagem API First funciona bem pois é muito mais fácil desenvolver novas aplicações a partir de um modelo de API bem definido, em vez de trabalhar apenas com "acordos" prévios que podem gerar muitos ruídos de comunicação entre a equipe de desenvolvimento da API e das aplicações que dela vão utilizar.

Rafael Mota Alves 27/08/2020

APIs possibilitam a comunicação entre programas, ou seja, podem ser usadas tanto por sistemas internos como um cliente oficial para uma aplicação. Mas também podem ser usadas por desenvolvedores externos ao projeto em questão para estender a funcionalidade oferecida para atender melhor as suas necessidades, ou até mesmo criar novos produtos usando a mesma API. O API first é uma técnica de desenvolvimento, que diz que se um projeto planeja oferecer uma API para os seus usuários,

Rafael Mota Alves 27/08/2020

ela deve ser a primeira coisa a ser desenvolvida, e os demais produtos da empresa, como uma página web ou um aplicativo, devem usar esse API, a ideia por trás dessa ideia é tornar essas API otimizadas para o uso de desenvolvedores, além de fazer com que a API seja testada pelos próprios desenvolvedores dos produtos da empresa.

Lucas Barros 27/08/2020

Desenvolvedores podem utilizar-se do conceito de API para desenvolvimento em paralelo, manipular dados de formas diferentes com mais contexto e assim estender funcionalidades de várias formas. API First é uma estratégia que desenvolve-se as APIs primeiro e o produto é construído em cima delas. Isso é relevante pois pode significar que a API é provida de acordo com o uso dos desenvolvedores e assim tende a ser mais consistente e documentada.

Ricardo Ebbers Carneiro Leão 28/08/2020

APIs são contratos que expõem como um consumidor pode usar um serviço específico. Definir o contrato primeiro permite que equipes distintas possam entrar em acordo sobre o que cada um precisa produzir e consumir, ajuda a definir melhor as fronteiras de contextos e não

precisam de esforço no sentido de definir detalhes de implementação antecipadamente.

 **Ricardo Ebbers Carneiro Leão** 28/08/2020

O termo API first só me lembra o Memo de Jeff Bezos: <https://api-university.com/blog/the-api-mandate/>

 **Heitor Carvalho** 29/08/2020

O valor das APIs hoje em dia está na modularização e exposição de serviços específicos de uma solução/programa/plataforma para serem utilizados de forma agnóstica por outras aplicações, sejam elas feitas internamente ou por terceiros. Isto é, uma API de checkout de uma empresa pode ser reutilizada por vários serviços internos e também disponibilizada para

que seja utilizada por outras soluções de comércio desenvolvidas por outras pessoas/organizações.

 **Heitor Carvalho** 29/08/2020

O significado e relevância do termo API FIRST reside em justamente tirar vantagem do valor das APIs hoje em dia, fazendo com que elas sejam de certa forma o "núcleo" do projeto, para serem consumidas por outras aplicações, serem consistentes e reutilizáveis. Elas são a parte primordial e principal pelo qual o projeto revolve em torno.

 **Danilo Lira** 29/08/2020

As API's trouxeram vantagens para os negócios por possibilitar integração de novas funcionalidades. Um bom exemplo disso são as API's da Google como o Maps, que possibilitam a inserção de um mapa a um site/app de forma simplificada sem a necessidade de entender o seu funcionamento interno. Já API First é o termo designado a estratégia de se pensar na construção de um software começando por sua API, isso é bastante relevante, pois o produto pode ganhar vantagem competitiva por ser mais flexível

 **Igor Fernandes** 30/08/2020

O principal valor das API's hoje em dia é facilidade que tornou integrar diferentes aplicações. O exemplo mais fácil de visualizar isso é a integração do Google/Facebook como forma de login em uma aplicação. Aplicações podem fazer uso de informações dessas plataformas para melhorar o ecossistema de sua aplicação, como importar lista de amigos, fotos e vídeos da plataforma.

 **Igor Fernandes** 30/08/2020

A API First se tornou boa prática no desenvolvimento de uma aplicação, pois não é necessário se preocupar qual tipo de linguagem ou plataforma o cliente vai funcionar. Uma vez construída a API, o negócio pode se movimentar para uma aplicação Web ou mobile sem a necessidade de fazer um desenvolvimento "from the scratch".

 **Arthur Fraude de Araújo** 30/08/2020

As API's representam o "core concept" de SaaS. Através delas, uma mesma lógica de negócio pode ser acessada por várias interfaces de acesso. O Termo API FIRST trás a ideia de priorizar a implementação da lógica no backend, para que esses recursos possam ser consumidos como um serviço para qualquer outro serviço ou aplicação. A estratégia permite que tecnologias de frontend sejam mais facilmente atualizadas ou substituídas, já que a lógica principal do produto está preservada em aplicações...

 **Arthur Fraude de Araújo** 30/08/2020

desacopladas.

 **Gabriel D'Luca** 30/08/2020

Adicionando ao que foi comentado pelo Heitor, uma abordagem API First também pode ser vista como uma estratégia de negócio, pois para muitas empresas faz total sentido que a API seja vista como o "core" ou o principal produto do negócio justamente por conta dos benefícios. Por exemplo, temos a Netflix: um claro exemplo de uma empresa que aderiu ao multiplataforma muito bem, e que hoje está presente em praticamente qualquer dispositivo que tenha tela. (continua)

 **Gabriel D'Luca** 30/08/2020

E isso só aconteceu por que a Netflix soube satisfazer essas necessidades de consumo diante de tantas plataformas diferentes (desde um celular Android até uma Apple TV). E isso seria impossível sem APIs prontas para consumo que garantissem esse multiplataforma. Ou seja, as APIs sendo toda a base para essa estratégia de negócio.

 **Gabriel D'Luca** 30/08/2020

Hoje também há casos de empresas que só possuem uma única API como produto (o negócio é a API). Enfim, acho que ficou bem claro que além desses benefícios em termos de desenvolvimento que muitas pessoas já trouxeram, como a própria manutenção, temos também muitos benefícios para o negócio (dinheiro!) onde a API First é algo completamente atrelado à estratégia da empresa.

 **Lucas Cardoso** 30/08/2020

Concordo com Gabriel e Heitor, e gostaria de adicionar como API's são importantes também para sistemas que utilizam microserviços, já que podem ser a melhor maneira de estabelecer uma comunicação entre eles. E nesse sentido, o uso da abordagem API First pode trazer diversas vantagens em termos de desenvolvimento paralelo e facilitar o desenvolvimento de

ferramentas que usem os serviços.



Talyta Maria Rosas Pacheco 30/08/2020

O uso das API's trouxe vantagens nos negócios tanto no âmbito de desenvolvimento quanto na experiência do usuário. Por se tratar de uma facilidade de integração de funcionalidades já desenvolvidas anteriormente, o desenvolvedor entrega a solução com mais agilidade. Com relação à experiência de usuário, muitas soluções oferecem a possibilidade de logar na plataforma por meio de outras plataformas, como Google/Facebook, ou ainda adquirir um produto digital por diversas opções de pagamento.



Talyta Maria Rosas Pacheco 30/08/2020

Como Emerson e Gabriel comentaram, a estratégia de API's First traz muitos benefícios tanto para os negócios que farão uso das API's quanto para os que desenvolvem, pois a API pode se tornar um produto de venda, e serem utilizadas para criar cada vez mais novos produtos inovadores



victor sena de lima attar 31/08/2020

O uso de API permitiu que o desenvolvimento fosse mais distribuído, pois qualquer serviço ou interface pode chamar outro serviço apenas pelo uso da API dele, assim facilitando o desenvolvimento multiplataforma. Como já haveria essa construção das API's para os serviços multiplataformas, a idéia do API FIRST vem para que se evite o retrabalho da mudança da api caso ela possa ser usada por fontes externas usada como um SAS, sendo assim ela é pensada como core da aplicação +



victor sena de lima attar 31/08/2020

sendo desenvolvida em paralelo com as equipes de frontend, para que não se espere até o fim do desenvolvimento da api para que seja integrada, caso havendo falha ou correção, não tenha que reiniciar todo o fluxo de espera .



Marcos Galvão 31/08/2020

Definitivamente as API's permitiram uma série de inovações no mundo digital atual, para começar a facilidade de consumir serviços de terceiros, já que as API's são regidas por contratos de comunicação (o que eu envio / o que eu recebo) é extremamente fácil terceirizar funcionalidades complexas que muitos negócios pequenos não conseguiram ou teriam dificuldade de manter e desenvolver, tendo apenas que pagar valores razoáveis para utilizar tais funcionalidades ...



Marcos Galvão 31/08/2020

Outro ponto interessante é pra quem produz a API, é fácil adaptá-la para possíveis mudanças de tecnologias ou utilizá-las no desenvolvimento de outros sistemas, permitindo que as mesmas funcionalidades estejam disponíveis em várias plataformas diferentes e sistemas diferentes, utilizando-se da mesma estrutura por trás dos panos.



Marcos Galvão 31/08/2020

O termo API First está relacionado ao processo de pensar/ desenvolver a API primeiro, enxergando desde o inicio que suas aplicações, independente de onde estejam, serão consumidoras das suas API's, de tal forma as interfaces podem ser descritas detalhadamente inicialmente, utilizar mocks, por exemplo, e permitir um desenvolvimento em paralelo sem muitos problemas e, além do mais, viabilizando o desenvolvimento facilitado de funcionalidades futuras reutilizando o que já foi criado.



Renato Joaquim de Miranda Ferreira 31/08/2020

API's permitem a integração de múltiplos sistemas, banco de dados e tarefas além de automatizar diversos serviços, aumentando muito a eficiência nos mais diversos meios. A abordagem de API first consiste em estabelecer um contrato de como aquela API deve funcionar, estabelecendo regras de negócios e design e, dessa forma, fazendo com que ela seja consistente e reusável para os mais diversos fins.

Lucca 31/08/2020

API's com interfaces "maduras" e bem definidas ajudam negócios digitais a implementarem seus produtos e plataformas seguindo alguns princípios de Design, como o princípio da responsabilidade única e o princípio de substituição de Liskov. O uso extensivo desses princípios permite que esses negócios consigam isolar a entrega de valor dos mesmos em interfaces concisas, tanto para stakeholders internos quanto externos, assim como trazem uma maior facilidade para a expansão do mesmo, por meio de ...

Lucca 31/08/2020

mudanças e melhorias na lógica de cada serviço em específico, que acabam ficando isoladas e não gerando impacto para seus utilizadores externos.

Lucca 31/08/2020

Produtos e serviços concebidos baseados em APIFirst acabam direcionando o desenho dessas aplicações, centrando elas em serviços concisos e independentes, trazendo aspectos de escalabilidade para o início do projeto



Izabella Nascimento 31/08/2020

Através das API's podemos ter as ligações nos dias de hoje entre diversos serviços, havendo uma integração entre um sistema maior e terceiros, como exemplo clássico temos a API

Google Maps. A API FIRST é vista como uma estratégia com a API sendo a base da estratégia de negócio, sendo desenvolvida primeiro com o produto sendo desenvolvido em cima da API, possibilitando que o sistema seja mais flexível e obtenha uma resposta mais rápida a novos planejamentos, também permite possuir um benefício...

 Izabella Nascimento 31/08/2020

de economia de tempo de desenvolvimento, permitindo as diferentes equipes do projeto trabalharem de forma simultânea.

 Gabriel Cavalcanti de Melo 31/08/2020

Um ponto muito importante que me vem à cabeça quando penso no valor que as API's trazem para os negócios digitais, é a agilidade no desenvolvimento da aplicação, pois como podemos integrar, com muita facilidade, um serviço dentro do sistema, os desenvolvedores podem utilizar o tempo que eles iam passar implementando um serviço que já é executado pela API, desenvolvendo outros aspectos da aplicação.

 Gabriel Cavalcanti de Melo 31/08/2020

E sobre a API FIRST, concordo com o que Danilo comentou sobre tornar as aplicações mais flexíveis, pois usar API's desde o começo, faz com que a aplicação se adapte a finalidades não planejadas inicialmente.

Gabriel Ramos 31/08/2020

O poder do desenvolvimento com a "divisão" possibilitada pelas APIs é um dos maiores valores no contexto API, junto a isso a disponibilidade e controle de acesso a dados se tornou trivial muitos projetos e empresas conseguem desenvolver ideias pelo simples fatos de que eles não são obrigado desenvolver tudo, consumir APIs com total segurança e estabilidade. Essa facilidade toda atrelado a ideia do API first se mantém relevante por características muito boas que ao se desenvolver uma API você +

Gabriel Ramos 31/08/2020

consegue ter uma melhor e mais eficiente manutenção de código, a possibilidade do trabalho em paralelo, uma forte redução de custos durante o desenvolvimento entre algumas coisas características que torna relevante o termo API first

 José Sheldon Brito Fekete 31/08/2020

O uso de API's evita o retrabalho, para que eu irei criar algo que já foi criado por outros? Criando assim integrações de sistemas mais facilmente e com segurança, identificando quem está usando seu sistema. API-First é uma estratégia de desenvolvimento em que a primeira ordem de negócio é desenvolver a API que corresponde aos interesses dos desenvolvedores para depois construir o produto.

 Vinicius Garcia 31/08/2020

quais são os principais riscos na abordagem API First? como ela dá errado?

 Ricardo Ebbers Carneiro Leão 31/08/2020

Se você mapear comunicações demais relacionadas a regras de negócio que ainda não estão 100% definidas você pode ser levado a ter que mudar o contrato posteriormente. Caso esse contrato já esteja em produção, vc pode ser forçado a refatorar para uma API incompatível com a primeira e ter que dar manutenção a duas versões distintas (especialmente se seu front for um app mobile, onde vc n tem controle sobre qual versão do app o cliente estará usando)

 Gabriel D'Luca 31/08/2020

Puxando a sardinha da qualidade aqui também, acho que uma abordagem API First tem que sempre considerar todos os cenários pra testes. Então ter uma estratégia de testes mtb bem definida, com cenários de permissão, segurança, destrutivos... tudo isso pensando no cliente e no contrato q foi definido. Do contrário, em algum momento o cliente não conseguirá cobrir todos esses cenários com testes de integração e isso pode comprometer a entrega (ou pior, ir a produção sem esses testes...), enfim.

 Rafael Mota Alves 31/08/2020

Eu diria que o risco da estratégia API First, é talvez um maior dificuldade em fazer um produto viável mais rápido. Já que o foco será desenvolver a API, antes de implementar um cliente por exemplo.

 Rafael Mota Alves 31/08/2020

Acho que outro risco também do API First é não entender o padrão de uso da API antets do desenvolvimento, o que pode levar a muitas modificações na API.

 Ramom Pereira dos Santos Silva 01/09/2020

Ao meu ver, tem o risco de focar no início em coisas técnicas de desenvolvimento ao invés de planejar a solução com base no cliente, além disso, existe a questão que no começo do desenvolvimento muitas questões da solução são desconhecidas, o que pode trazer muitas mudanças na API.

 Claudio Pacheco 01/09/2020

O principal benefício que as APIs trouxeram aos negócios digitais é a flexibilidade conferida na forma de criar, disponibilizar e manter um serviço. Ou seja, não é preciso criar do zero todos os

termo de API, disponibilizar o menor conjunto de dados, não é preciso usar os dados de componentes de uma aplicação: podemos utilizar serviços terceiros em comunicação com o nosso através de APIs (ex: Uber utilizando Google Maps). Essa flexibilidade acaba por conferir também maior eficiência através da abstração.

 Claudio Pacheco 01/09/2020

Já o termo "API First" refere-se a uma estratégia em que prioriza-se o desenvolvimento da API para, posteriormente, desenvolver a aplicação em si. Esse tipo de abordagem permite que APIs sejam bem estabelecidas para serem utilizadas por outros serviços que venham a consumi-la posteriormente.

 Ramom Pereira dos Santos Silva 01/09/2020

O principal benefício das APIs é a possibilidade de integração entre softwares e serviços diferentes, automatizando processos e auxiliando na produtividade de diversos desenvolvedores. O reaproveitamento de funcionalidades, sem a necessidade de novo desenvolvimento é um grande valor. Como exemplo, temos: API do Google Maps e API de Gateways de Pagamento para Ecommerce.

 Ramom Pereira dos Santos Silva 01/09/2020

Plataformas que precisam da utilização de localização, podem tranquilamente utilizar o Google Maps para esse fim, com uma simples integração entre os serviços. Lojas virtuais utilizam o serviço de Gateways de pagamento, com a utilização de APIs. O grande benefício de unificar serviços de empresas grandes e pequenas.

 Ramom Pereira dos Santos Silva 01/09/2020

Sobre a abordagem API First, é uma maneira de desenvolver um software baseando-se, primeiramente, em desenvolver a API de fornecimento de serviços, para depois desenvolver o clients (web ou mobile). Como hoje em dia, as aplicações são feitas para execução em nuvem, construindo APIs primeiro, favorece o desenvolvedor ao colocar seus interesses em primeiro lugar, além de desenvolver uma API capaz de servir quaisquer clients que se integrem a si.

João Vasconcelos 02/09/2020

Faço da sua a minha resposta!

João Vasconcelos 02/09/2020

APIs flexibilizou bastante como os negócios são feitos. Hoje um programador/empresa pode não ser mais focado apenas em uma aplicação, e sim em desenvolver uma API e monetizá-la; ou uma empresa pode expandir a sua marca disponibilizando APIs que se integrem com seu produto. Além do financeiro, APIs agilizaram o desenvolvimento de software permitindo reutilizar APIs próprias ou de outros devs (login do Facebook ou Google, "crawler" de dados do Spotify, etc)

 Roberto Oliveira 02/09/2020

um dos principais benefícios da API é a flexibilidade do negocio, podendo dividir as equipes em frentes diferentes e tocar em paralelo as demandas, isso vendo no lado do time e para o cliente o melhor beneficio seria a utilização dessa api de modo desacoplado e independente do device utilizado (mobile , desktop, apps etc).

 Roberto Oliveira 02/09/2020

por experiencia própria API first pode trazer bastante agilidade do desenvolvimento do projeto, uma vez que o front end pode começar suas demandas de forma independente se baseando no contrato da api.

 Elverson Melo 02/09/2020

As APIs permitiram uma maior integração (segura) de aplicativos, mais que um conceito técnico, elas são estratégias de negócios, conhecido como API Economy. Ficou mais fácil criar novos negócios, uma vez que a quantidade de APIs disponíveis é grande, não é preciso criar um negócio do zero. Apesar disso para o maior sucesso nos negócios é necessária a construção e o planejamento das APIs necessárias para integração a terceiros para só então começar a projetar a aplicação.

 Elverson Melo 02/09/2020

Isso dá certo foco nos desenvolvedores que por ventura queiram integrar os seus serviços aos deles e é conhecido como API-First.

 Victor Gabryel 02/09/2020

As API's abriram um mar de oportunidades para as grandes empresas e também pequenos desenvolvedores, onde elas podem ter suas API's é possível utiliza-las até mesmo para integrar com outros sistemas. API FIRST significa que a própria API será o foco inicial do desenvolvimento da aplicação, podendo ela ser a única a ser desenvolvida. Nela será planejada quais e como as funcionalidades serão projetadas e como serão utilizadas sendo relevantes podendo ter essas integrações com outros sistemas.

 Hiro Miyakawa 09/09/2020

A possibilidade de integração, absurda, de até mesmo criar novos serviços em cima dela. Por exemplo, há empresa em Recife que otimizou ferramentas de publicação para instagram. Isso é bom tanto para a provedora quanto para o usuário do API. API First é a estratégia de focar em criar API, mesmo antes de ter o serviço rodando, com o foco total de integração (ex: netflix)

Qual o melhor exemplo, na sua opinião, de API's Open, ReRestrita e Privada? Como escolher qual utilizar na minha arquitetura?

EV Emerson Victor 27/08/2020

Privadas são utilizadas para operações internas de uma organização (ex: internet banking), públicas estão disponíveis para qualquer pessoa utilizar (ex: registro de ocupação hospitalar COVID-19 oferecido pelo governo) e as restritas podem ser utilizadas por pessoas específicas (ex: empresas que oferecem seus serviços por meio de APIs). A escolha depende do objetivo do negócio: oferecer um serviço, disponibilizar dados para desenvolvimento de soluções, estruturar seus serviços internos, etc.

RM Rafael Mota Alves 27/08/2020

A API Open (Pública) está disponível para qualquer indivíduo utilizar, esses tipo de API é recomendado para APIs que requisitam dados não sensíveis e portanto podem ser usadas por qualquer pessoa ou desenvolvedor, um exemplo seria a API que retorna o catálogo de e-commerce, nesse tipo de API devemos nos preocupar com coisas como ataques de negação de serviço e também em ter uma boa documentação disponível. Já a API Restrita, o acesso é limitado para usuários/desenvolvedores específicos

RM Rafael Mota Alves 27/08/2020

normalmente a identificação é feita por tokens de acesso e podem requisitar dados pessoais ou fazer alterações, um exemplo seria a API que edita os dados de um usuário em uma rede social. Já a API privada, é um tipo de API destinada apenas para o uso interno na empresa, e podem fazer ações administrativas, ou regras de comunicação entre serviços, um exemplo é uma API que retorna métricas sobre o funcionamento de uma aplicação, geralmente o controle de acesso é feito a nível de Rede.

LB Lucas Barros 28/08/2020

API's open qualquer um pode fazer uso e possui dados não sensíveis, abertos ao público. Exemplo: APIs de geocoding, que retornam informações sobre locais. API's restritas têm seu uso limitado a algumas pessoas, através de autenticação. Um exemplo seria a API do Spotify, por exemplo, que como possui dados que podem ferir privacidade, utiliza-se de OAuth para garantir que o acesso aos dados seja dado a pessoas através de permissões.

LB Lucas Barros 28/08/2020

APIs privadas são utilizadas a nível interno, provavelmente limitadas através de uma VPN, ou coisa do tipo. Um exemplo seria APIs de backend de uma empresa que se comunicam entre si.

RE Ricardo Ebbers Carneiro Leão 28/08/2020

APIs open são APIs públicas que não necessitam de autenticação, normalmente limitadas no volume de uso de forma a não sobrecarregar os servidores onde elas estão rodando. APIs restritas possuem algum sistema de autenticação, que é útil para identificar os consumidores e possivelmente monetizar. Por fim, as APIs privadas são aquelas protegidas por barreiras físicas ou lógicas e utilizadas normalmente entre sistemas de uma mesma organização. A escolha vai depender totalmente do tipo de uso da API.

HC Heitor Carvalho 29/08/2020

Um exemplo de API OPEN seria: <https://pokeapi.co/> - Esse tipo de API é totalmente aberto para consulta pública e utilização em aplicações de terceiros. Geralmente o consumo (número limite de requisições em um período de tempo) é cobrado, então esse tipo de API é uma boa escolha quando os DADOS são o produto principal e há uma demanda por sua consulta.

HC Heitor Carvalho 29/08/2020

Um exemplo de API RESTRICTA seria: <https://developer.spotify.com/documentation/web-api/> - Esse tipo de API fornece dados ou serviços parciais de sua aplicação. No exemplo do Spotify, é possível obter uma miríade de informações sobre músicas e álbuns, mas as restrições da API não permitem que alguém recrie o player do Spotify do zero, por exemplo. Entretanto geralmente, o provedor da API detém o "core" do negócio mas fornece recursos secundários para desenvolvedores externos.

HC Heitor Carvalho 29/08/2020

Um exemplo de API PRIVADA seriam APIs que não conseguem colocar o link aqui, devido a sua própria natureza: Geralmente fornecem acesso a recursos e serviços internos a uma organização, de forma que ela possa reutilizar em múltiplos produtos ou plataformas. Como por exemplo, uma empresa de entregas que pode ter uma API acessível apenas pelos seus produtos internos Web e Mobile, com versões para cliente e entregador.

DL Danilo Lira 29/08/2020

Uma API pública ou Open é uma interface disponibilizada para todos, ou seja, terceiros podem interagir com essa API. Ex: Compras públicas do governo federal que contêm informações das compras públicas realizadas no sistema SIASG/Comprasnet. Já as APIs privadas são utilizadas internamente pelas empresas para suas operações. Ex: Interfaces para serviços críticos das empresas. As APIs restritas são aquelas utilizadas para facilitar a comunicação entre empresas parceiras.

 **Danilo Lira** 29/08/2020

A escolha da utilização depende do intuito da empresa e para quem ela pretende disponibilizar aquilo.

 AF

Arthur Frade de Araújo 30/08/2020

Um bom exemplo de api open está disponível em <https://pokeapi.co/>. Lá, qualquer pessoa do mundo, a qualquer momento, pode consultar informações sobre o diversos pokemons existentes. Ela é enquadrada como open por ser de acesso público e irrestrito. Um exemplo de api restrita são as fornecidas pelo DATAPREV do SERPRO. Que é um orgão nacional de manutenção e fornecimento de dados sociais. Através de um cadastro e pagamento de mensalidade é possível consultar esses serviços. Já APIs privadas...

 AF **Arthur Frade de Araújo** 30/08/2020

Não são publicamente conhecidas e encontram-se no âmago das empresas. Esse tipo de api geralmente transfere dados muito sensíveis.

 IF

Igor Fernandes 30/08/2020

API abertas são aquelas que não necessitam de autenticação para ser consumida. São utilizadas quando os dados que serão acessados não são sensíveis (são públicas). Exemplo: Resgatar CEP de um Endereço Já API restritas fornecem acesso aos dados da plataforma, mas necessitam de credenciais para serem utilizadas. Exemplo: Resgatar pedidos de um usuário num e-commerce.

 IF **Igor Fernandes** 30/08/2020

E por fim, as API's privadas são utilizadas internamente pelos serviços de uma plataforma, funcionalidades que só devem ser acessadas pelo back-end de um serviço ou pelo administradores da plataforma. A escolha de qual categoria utilizar depende de que tipo de dados serão manipulados. Muitas vezes um negócio pode ter os três tipos.

 GD

Gabriel D'Luca 30/08/2020

Como já trouxeram bastante, exemplos rápidos de cada uma: - API aberta: A PokeAPI, como já trouxeram aqui, algo completamente público, mas com acesso completo por \$\$. - API restrita: A API do Twitter, que permite consumo após autenticação, mas com restrições. - API privada: Algo completamente interno de uma organização. A escolha deve levar em conta vários fatores. Por ex, uma API pública pode ser vista como uma dependência, pois possui sim um risco de ser descontinuada.

 GD **Gabriel D'Luca** 30/08/2020

Uma API aberta também pode ser consumida totalmente por qualquer um, então se o "core" de sua aplicação ou serviço estiver revolto nessa API aberta, qualquer competidor tem acesso (basta pagar por ela). Já uma API restrita vai ter suas restrições, então não é possível pegar tudo dela, a empresa "dona" da API é a única que detém tudo... enfim, tudo depende do objetivo, levando em consideração os riscos e o caso pra garantir a melhor aplicabilidade do tipo escolhido.

 LC

Lucas Cardoso 30/08/2020

API Aberta - Public APIs, uma API aberta de APIs abertas (mais informações: <https://github.com/davemachado/public-api>) API Restrita - Twilio possui APIs para mensagens (SMS, Whatsapp, etc) API Privada - Comunicação interna entre serviços de uma empresa. A escolha se dá em como se planeja que a API seja utilizada. Se a API lida com dados internos da empresa, deveria ser fechada. Caso o sistema possa ter vantagem de um ambiente aberto mais inovador, deve ser aberta. (Continua)

 LC **Lucas Cardoso** 30/08/2020

Caso se tenha interesse em controlar o acesso aos dados, ou monetizar o uso, deve se utilizar uma API restrita

 VS

victor sena de lima attar 31/08/2020

Api públicas: Apis onde qualquer pessoa pode ter acesso Api restrita: Apis abertas ao público, mas que necessitam de credenciais de acesso. Ex: Pagarme Api privada: Api fechada apenas para o uso interno da empresa. Para escolher qual o melhor tipo: Depende qual o público que irá usar, se somente a empresa irá usar, deve ser fechada, caso o público externo possa usar sem nenhuma limitação, pode ser aberta, caso não, restrita.

 MG

Marcos Galvão 31/08/2020

As API's públicas basicamente possuem acesso liberado para qualquer indivíduo, sendo assim uma pessoa pode requisitar informações nela a qualquer momento, um ótimo exemplo é a <https://corona.lmao.ninja/>, API disponibilizada pouco tempo depois da pandemia se espalhar pelo mundo, ela fornece diversas informações demográficas acerca dos países e qual o estado do vírus no local, desde seu lançamento já recebeu mais de 20 bilhões de requests e é disponibilizada gratuitamente ...

 MG **Marcos Galvão** 31/08/2020

Uma API restrita, como já falaram, são abertas ao público, entretanto o fornecimento de credenciais é necessário, em grande parte da vezes por terem custos por requests ou limitações de volume, por exemplo, as API's do IBM Watson permitem utilizar as mais variadas funcionalidades de Machine Learning com um custo muito baixo, por vezes sendo até de graça. Por fim as API's privadas são de uso interno da empresa, não sendo acessíveis por aplicações externas.

 C

RJ**Renato Joaquim de Miranda Ferreira** 31/08/2020

API's Abertas: Mapas do Waze, uma vez que trás uma série de informações a respeito da localidade sem que se pague um custo por isso. API's Restritas: Telegram, disponível para os mais diversos usos para um desenvolvedor, como os chatbots, sem comprometer a integridade das mensagens entre diferentes usuários, ou seja, permitindo apenas parte da aplicação acessível.

RJ Renato Joaquim de Miranda Ferreira 31/08/2020

API's Privadas: Cálculo de transações dentro de uma companhia, uma vez que é necessário para os fins jurídicos e internos faz total sentido que ninguém que não seja da companhia tenha acesso. A resposta certa sobre a escolha é: Depende do sistema que deseja ser desenvolvido e se haverá tráfego de informações sensíveis.

Lucca 31/08/2020

API's públicas são API's, que como já foi citado acima, tem uso livre por qualquer pessoa. Um exemplo de API pública aberta é a boredapi (<https://www.boredapi.com/>) que seleciona uma atividade para você fazer quando entediado. (Bom para a quarentena) API's restritas são aquelas que são disponibilizadas para o público geral, porém que necessitam de algum tipo de verificação do usuário. Um exemplo desse tipo de API é a do Pinterest, que necessita de um cadastro para o consumo de seus serviços.

Lucca 31/08/2020

API's Privadas são as utilizadas internamente por uma companhia, como era o caso do BigTable do Google, que até o ano de 2015 era um serviço de armazenamento acessível somente pelos serviços e operações da empresa. (Hoje já se tornou uma API Restrita). A escolha de um dos tipos de API varia de acordo com o seu negócio, ou das partes do mesmo. Muitas empresas possuem API's que, por expor funcionalidades exclusivas e prioritárias, são construídas privadas, e API's relacionadas a um mesmo negócio.

Lucca 31/08/2020

sendo que por não afetarem o diferencial de valor da empresa, podem ser oferecidas como abertas.

IN**Izabella Nascimento** 31/08/2020

API Pública: são as disponibilizadas gratuitamente com acesso liberado a todos, sendo possível implementar funcionalidades com um esforço bem menor, como exemplo o Facebook. API Restrita: geralmente fornece apenas partes de suas funcionalidades, sendo bastante lucrativa ao escolher com quem será disponibilizada, como a empresa de pagamentos Braintree e a empresa de telefonia Twilio. API Privada: é disponibilizada dentro da empresa, protegendo dados, como os sistemas de internet banking.

Gabriel Ramos 31/08/2020

API privada criada para uso apenas de uma organização, carregando, normalmente dados sensíveis. API Restrita, é uma API pública, sua existência é conhecida mas seu acesso é restrito. API Pública, qualquer pessoa pode consumir e ter acesso. Sem muito enrolar, o super resumo delas é isso, agora, para se escolher qual utilizar vai depender do contexto que essa API vai ser desenvolvida e para que ela vai ser utilizada, se você julga que os dados presentes nela são dados sensíveis, o ideal é que seja

Gabriel Ramos 31/08/2020

privada ou restrita, dependendo do contexto. Caso os dados possam ser úteis para outras pessoas, talvez a ideia de uma API pública seja a sua.

JS**José Sheldon Brito Fekete** 31/08/2020

API's open ou públicas estão disponíveis livremente para o uso e integração em outra aplicação.

JS José Sheldon Brito Fekete 31/08/2020

Um exemplo é o Google Maps, API's restritas são conhecidas porém apenas liberada para parceiros de negócios. API's privadas são aquelas restritas dentro de uma empresa, integrando processos internos de negócios. Para saber qual tipo de API utilizar é importante saber os tipos de dados que irão trafegar pela sua API, de forem dados importantes que não podem cair nas mãos de concorrentes, é recomendado o uso da API privada ou até mesmo restrita, se for feito para uso futuro de outras aplicações

JS José Sheldon Brito Fekete 31/08/2020

Sua ou de outros desenvolvedores, utilizar a política de API's open ou aberta.

CP**Claudio Pacheco** 01/09/2020

API's open estão disponíveis para utilização sem a necessidade de autenticar. Um dos principais exemplos é a do Google Maps, que atualmente opera no modelo "freemium": <https://cloud.google.com/maps-platform/pricing> Já as API's Restritivas, como têm um maior número de dados sensíveis envolvidos, exige que haja alguma autenticação durante o uso. Um exemplo é a API do Trello: <https://developer.atlassian.com/cloud/trello/>

CP Claudio Pacheco 01/09/2020

Por fim, as API's privadas têm seu uso reservado a uma mesma organização/ecossistema

de negócios devido à sensibilidade dos dados e negócios envolvidos. O tipo API a ser usado na minha arquitetura dependerá do quê é a minha solução e quem precisará consumir os dados via API.

TM

Talyta Maria Rosas Pacheco 01/09/2020

A escolha dentre esses tipos de API vai depender do quão sensíveis são os dados que serão gerenciados, além dos custos que serão considerados para manter a API. Os bons exemplos para mim de cada uma dessas API's pra mim são: Pública: Google Maps Privada: Utilizada nas assistentes virtuais, como Siri, já que é desenvolvida para utilização dos usuários da empresa. Restrita: Facebook, já que precisa de autenticação para exibir dados de um usuário, por exemplo.

RP

Ramom Pereira dos Santos Silva 01/09/2020

API's Open: APIs abertas, que não necessitam de autenticação para integração, por exemplo, API do Correios para estimar custo de entrega, normalmente utilizada em E-commerce. API's ReRestrita: APIs que precisam de autenticação para integração, por exemplo, APIs de Gateways de Pagamento ou Facebook para acesso ao perfil de usuários. API's Privada: APIs com uso interno na empresa, devido ao alto grau de sensibilidade do negócio.

Ramom Pereira dos Santos Silva 01/09/2020

Quanto a escolha de qual usar, isso deve ser decidido de acordo com a necessidade do negócio a ser modelado, levando em consideração os possíveis custos e restrições que venham impactar o negócio da organização.

GC

Gabriel Cavalcanti de Melo 02/09/2020

API aberta: O primeiro exemplo que me vem à cabeça é a API do Google Maps, que é usada nos mais diversos tipos de aplicações. Desde sites de busca de hotéis ou restaurantes até jogos para celular ou navegador. API Restrita: Serviços que necessitam de autenticação para poderem ser acessadas. API Privada: São aquelas que são desenvolvidas para uso interno da empresa. Como por exemplo a Netflix, que faz uso de suas APIs privadas para manter seus serviços funcionando.

Gabriel Cavalcanti de Melo 02/09/2020

A escolha de qual usar vai depender dos objetivos da aplicação. Podendo usar cada uma das três categorias, de acordo com a tarefa que o serviço vai executar

João Vasconcelos 02/09/2020

Sumarizando o que foi dito acima: open são APIs que podem ser utilizadas sem login. Restritas necessitam um cadastro, um token de acesso e, algumas vezes, vem com monetização. Privadas, como o nome diz, são APIs fechadas para uma empresa, onde diversos setores da mesma pode usar a mesma API.

João Vasconcelos 02/09/2020

A escolha de qual utilizar vem de acordo com a necessidade: público ou restrito se o dev precisar de uma lógica já implementada por alguma API (evitar reinventar a roda). Privado seria uma abordagem se a proposta da API pode ser reutilizada por mais aplicações internas.

RO

Roberto Oliveira 02/09/2020

a escolha dessa arquitetura depende muito do projeto e dos dados que vão estar envolvidos, por exemplo a api open não necessita de qualquer chave de autenticação, google maps por exemplo. api restritivas: google cloud, e api privada que são utilizadas internamente exemplo: Netflix que fechou sua api em 2014.

VG

Victor Gabryel 02/09/2020

Acredito que a abordagem que deve ser escolhida depende de alguns fatores, caso eu necessite ter um maior domínio sobre ela (API Privada), se eu acredito que qualquer pessoa possa explorar o uso dela (Pública) ou se eu tenho como objetivo fazer parcerias com outros projetos/empresas para a facilitar a comunicação desses dados (ReRestrita). Alguns exemplos que resumem bem cada abordagem são: CovidAPI: Esse é um ótimo exemplo de API Pública, onde qualquer pessoa pode utilizá-la para informar +

Victor Gabryel 02/09/2020

o quantitativo de casos, mortes, etc por país. referência:
<https://github.com/javieraviles/covidAPI> Internet Banking: Basicamente, como as Internet Banking's trabalham com informações confidenciais e regras de negócios bem restritas é essas empresas precisam que essas informações estejam em controle.

HM

Hiro Miyakawa 09/09/2020

API Open: Sem autenticação. Para qualquer cliente poder acessar. (ex: redes sociais, apesar de algumas partes serem fechadas). API ReRestrita: Que necessita de uma autenticação, cadastro. As partes fechadas da rede social, por ex, precisando do API key. API Privada: apenas para uso interno. Microserviço parece entrar nisso.

O que faz esse tal de API Gateway? Quando utilizá-lo?

C

EV

Emerson Victor 27/08/2020

API gateway é uma forma de gerenciar APIs. Ele intercepta as requisições, agrupa todos os serviços necessários para responder e, em seguida, retorna o resultado. Algumas funções comuns incluem autenticação, roteamento, limitação de taxa, faturamento, monitoramento, análise, políticas, alertas e segurança. Um grande benefício de usar um API gateway é que ele encapsula a estrutura interna de uma aplicação e, em vez de requisitar a serviços específicos, requisita apenas ao gateway.

LB

Luan Brito 27/08/2020

além disso também pode fazer o load balance para não sobrecarregar uma instância da API (caso de ser distribuída) e cache de informações

SG

Saulo Guilhermino 27/08/2020

Um gateway também pode servir a propósitos de proteção e monitoramento da aplicação, permitindo que sejam executadas apenas requisições autenticadas e também aplicando ferramentas de extração de métricas das requisições e respostas para futuras análises de fluxo

RM

Rafael Mota Alves 27/08/2020

O API Gateway serve como um "agregador" de APIs simplificando o seu uso em um único "lugar", ele pode direcionar as chamadas para APIs para outros serviços com também "juntar" dados de vários serviços para dar um resposta única. Outro uso do API Gateway é servir como uma camada de abstração para uma série de funcionalidades usadas em APIs com: autorização e autenticação, tratamento de erros, monitoramento e prevenção de ataques de negação de serviço. Portanto o API Gateway pode ser usado

Rafael Mota Alves 27/08/2020

tanto para agregar as APIs de múltiplos serviços, como também como um "porta-voz" para a sua API, controlando acesso aos recursos e também trazendo informações sobre o uso deles.

Rafael Mota Alves 27/08/2020

Uma implementação legal para vê como esse padrão funciona na prática é o serviço API Gateway da AWS, ele provê várias abstrações para as aplicações, além de trazer integrações com os próprios serviços da AWS

LB

Lucas Barros 28/08/2020

API Gateway é uma porta única de entrada para vários serviços. Ele pode ser utilizado para agrupar funcionalidades em comum aos serviços, como autenticação, load balancing, monitoramento, etc. Utiliza-se quando há várias APIs a serem utilizadas pela aplicação, pois diminui a complexidade do código do cliente, otimizar carga de rede e permitir uma refatoração menos trabalhosa dos microserviços (já que os acessos estariam centralizados)

RE

Ricardo Ebbers Carneiro Leão 28/08/2020

O pattern API Gateway fica na fronteira entre o 'mundo externo' e os microserviços e age como um centralizador de requests que chegam, distribuidor desses requests para os respectivos serviços (fazendo papel de Load Balancer), é onde normalmente se implementam sistemas de circuit breaking para evitar sobrecarregar sistemas que estejam falhando, também é um ótimo candidato para gerenciar autenticação e autorização. É favorável sempre ter um serviço de Gateway quando trabalhar com microserviços.

DL

Danilo Lira 29/08/2020

Um API gateway é uma ferramenta para gerenciar a utilização de uma interface. Essa ferramenta é utilizada para interceptar todas as chamadas feitas para a API e, dependendo do seu objetivo, pode desempenhar diversas funções como as que foram citadas pelos colegas. Ele deve ser utilizado quando houve necessidade de, por exemplo, uma autenticação para prevenir possíveis falhas, ou para monitorar o que os usuários da API estão fazendo com ela. A empresa deve analisar a ferramenta e sua necessidade.

HC

Heitor Carvalho 29/08/2020

Um API Gateway é um ponto de entrada único para todos os clientes, que roteia as requisições para a API mais adequada. O uso desse padrão é recomendado quando o cliente geralmente irá fazer requisição a múltiplos microserviços simultaneamente como parte do serviço completo, daí existe vantagem em centralizar todos esses serviços em um ponto único de acesso.

Heitor Carvalho 29/08/2020

Outro tipo de uso é quando os protocolos dos serviços são diferentes sendo necessária uma unificação na padronização, ou então quando a carga é diferente para clientes diferentes (ex: aplicações mobile trabalham melhor com requisições menores e mais rápidas do que um ambiente web, ou podem precisar de dados diferentes)

AF

Arthur Frade de Araújo 30/08/2020

Eu poderia dizer que uma API gateway é uma espécie de implementação alto nível do padrão "facade" para um conjunto de microserviços. Basicamente o gateway será o orquestrador de requisições, ele irá ser um ponto central de acesso a todos os componentes de software da arquitetura. Um bom caso de utilização dessa ferramenta é para a comunicação entre front e back-end. Nesse caso o frontend só precisa consultar um único domínio para ter acesso a qualquer serviço do "ecossistema" de serviços.

 AF Arthur Frede de Araújo 30/08/2020

Caso contrário qualquer front (mobile ou web) teriam que gerenciar múltiplas url para acessar recursos disponíveis em diferentes locais.

 IF Igor Fernandes 30/08/2020

A API Gateway é porta de entrada para os serviços de uma plataforma. Ele guia a rota requisitada para o container que hospeda o serviço que trata esse tipo de requisição. Muitas vezes o papel deles também é gerenciar as permissões de acesso à esses serviços, tratando a autenticação de todos os serviços num único lugar.

 IF Igor Fernandes 30/08/2020

Acredito que um bom cenário para utilizá-lo é quando queremos gerenciar melhor o uso dos serviços, e queremos uma camada a mais de proteção (balanceando o uso das instâncias, e impedindo ataques DDoS/DoS aos serviços).

 GD Gabriel D'Luca 30/08/2020

Acho que já temos comentários suficientes aqui sobre o que seria um API Gateway, então resumindo: um objeto que encapsula o acesso a um sistema ou recurso externo (nesse caso, uma API). Eu queria trazer aqui justamente um pouco sobre quando utilizar. Eles podem fornecer utilidade para lidar com algumas questões mais genéricas - como autenticação e limitação de taxa - mas... (continua)

 GD Gabriel D'Luca 30/08/2020

QUALQUER inteligência de domínio (como transformação de dados ou regras de negócio), deve estar >nos aplicativos ou nos serviços<. Do contrário, temos o que chamamos de "Overambitious API Gateways": software de transporte capazes de executar lógicas críticas de aplicação, onde os próprios API Gateways encorajam um design que é difícil de testar e deployar. <https://www.thoughtworks.com/radar/platforms/overambitious-api-gateways>

 GD Gabriel D'Luca 30/08/2020

Então, sim, API Gateways oferecem vários benefícios, mas ele precisa se concentrar em fazer o que faz de melhor: roteamento de tráfego e tarefas mais comuns/genéricas. Já lógica de domínio, é melhor deixar no código da aplicação ou serviço por questões de testabilidade e compatibilidade com as práticas de desenvolvimento de software definidas pelo time.

 LC Lucas Cardoso 30/08/2020

Um API Gateway faz o gerenciamento de APIs. Pode servir como ponto de entrada para várias APIs internas, podendo fazer controle de carga, gerenciar chamadas para serviços internos, e funcionalidades de segurança, como mecanismos para detecção de DDoS. Deve ser usado para disponibilizar as APIs de um sistema de maneira centralizada, sendo imprescindível em APIs públicas. Para APIs fechadas, apesar de poder ser útil, não precisa ser usado

 VS victor sena de lima attar 31/08/2020

O API gateway é o ponto de correio das APIs da sua aplicação. Ele que vai gerenciar para qual serviço interno a requisição externa vai ser redirecionada. Ele também serve como barreira contra invasões, sendo ele responsável por fiscalizar as credenciais de segurança e redirecionar somente se for autorizado, com essa segurança consegue conter ataques DDOS, por exemplo.

 MG Marcos Galvão 31/08/2020

O objetivo do API Gateway é prover um ponto unificado de acesso (um portão) para as APIs, sendo assim as aplicações consumidoras não enxergam como os serviços estão particionados ou onde eles estão, ademais o gateway provê funcionalidades muito convenientes, assim como Igor comentou ele pode ser utilizado como uma camada de segurança, ofuscando os serviços das aplicações, um balanceador, administrando quais containers receberão as requests, um unificador de protocolos, ou seja, as aplicações

 MG Marcos Galvão 31/08/2020

podem se comunicar com o gateway utilizando diversos protocolos, e não menos importante, pode reduzir o número de chamadas sob responsabilidade das aplicações. Imagine que uma aplicação precisa de três informações que estão espalhadas por três serviços, o gateway poderá receber uma única request, solicitar aos três serviços e retornar para aplicação todas as respostas de uma vez, tornando assim o cliente menos sobrecarregado,

 RJ Renato Joaquim de Miranda Ferreira 31/08/2020

API Gateway coordena e 'orquestra' como as mais diversas requisições dentro da arquitetura do sistema são processadas. Ele pode ser utilizado quando se mostra muito útil quando existe uma relação entre sistemas legados para outros sistemas dentro da mesma arquitetura e também quando se existe uma necessidade de multiplataforma, ao invés de fazer algo gigantesco para englobar todos os casos se torna mais eficiente utilizar o API Gateway para coordenar esses serviços.

 TM Talyta Maria Rosas Pacheco 31/08/2020

Conforme as aplicações vão se tornando mais complexas, que é a realidade de muitas

aplicações, surgem preocupações que antes não tinham, como qual serviço está no ar e qual ip direcionar as inúmeras chamadas de requisições. Desta forma, a API Gateway concentra os acessos, organizando-os, simplificando muitos dos processos necessários para o funcionamento do sistema, como o processo de autorização e autorização e versionamento.

Lucca 31/08/2020

O Gateway de API atua como uma interface única de entrada para os usos de um dado serviço, permitindo o isolamento de comportamentos esperados para todos os endpoints e serviços em um único local. Comportamentos como taxação financeira, autenticação, agregação de microsserviços, e uso e gerenciamento de funções serveless integradas.

IN

Izabella Nascimento 31/08/2020

O API Gateway é uma ponte entre o cliente e o serviço de back-end, sendo responsável por encaminhar as chamadas pros microsserviços correspondentes corretamente, servindo como uma barreira de controle, também padronizando a comunicação de acordo com os diferentes tipos de formatos de respostas. Ela pode ser utilizada para evitar ataques, como forma de monitorar e analisar como sua API está sendo utilizada, para gerenciamento em microsserviços como cada solicitação pode fazer chamadas para...

Izabella Nascimento 31/08/2020

diversos sistemas.

GC

Gabriel Cavalcanti de Melo 31/08/2020

Além de ser utilizada no processo de autenticação e autorização como muitos comentaram acima, entendi que API Gateway é utilizada para facilitar o gerenciamento de mensagens entre os serviços. Configura-se como um monitorador capaz de apresentar a quantidade de requisições, endpoints mais acessados ou quais problemas inesperados apresentados nos endpoints e qual o tempo de resposta nos endpoints, antecipando a necessidade de escalar o serviço que está tendo um tempo de resposta maior.

Gabriel Ramos 31/08/2020

API Gateway, pode ser feito um paralelo como a portaria de um condomínio, assim como na portaria é normalmente o local de acesso a função do API Gateway é similar, direcionado às requisições feitas e direcionando para "onde" devem ir. Ainda aproveitando a ideia da portaria, para passar pela portaria, você precisar ser "autenticado" assim podemos dizer que API Gateway funciona, aumentando a segurança, autenticando as requisições.

JS

José Sheldon Brito Fekete 31/08/2020

API Gateway faz parte do sistema de gerenciamento da API, interceptando as solicitações de entrada e as envia por meio desse sistema, onde executa diversas funções, tais como autenticação se for necessário, ou seja ele conecta o cliente com uma coleção de serviços do back-end. Sua utilização é recomendada pela segurança que ele proporciona controlando o tráfego de solicitações e simplificando as chamadas para uma arquitetura de microsserviços.

CP

Claudio Pacheco 01/09/2020

De acordo com o que pude ler na internet e nos comentários postados acima, um API Gateway é uma ferramenta responsável pelo recebimento das requisições exteriores, seu processamento, comunicação com os serviços backend e, por fim, retorno do resultado requerido. Pode ser utilizado para diversos fins, como proteger a API de uso excessivo ou entender melhor o uso a partir de ferramentas de analytics. +

Claudio Pacheco 01/09/2020

Na arquitetura microsserviços, a API Gateway também pode ser usada para agregar as requisições que, de outra forma, requeriam diversas chamadas a aplicações diferentes. (<https://www.redhat.com/en/topics/api/what-does-an-api-gateway-do>)

RP

Ramom Pereira dos Santos Silva 01/09/2020

API Gateway é um unificador de APIs, funcionando como ponto único de entrada para diversos Microserviços, ou seja, os clients não precisam saber cada API individualmente, essa responsabilidade é do API Gateway, apenas sabem da existência de uma única API, o API Gateway.

Ramom Pereira dos Santos Silva 01/09/2020

Normalmente, numa Arquitetura de Microsserviços é indicado a utilização de API Gateway, pois evita que os clients conheçam os endereços lógicos e físicos de cada Microsserviços, evitando a necessidade de Service Discovery, por parte dos clients. Manutenção dos serviços da API, inserir ou remover serviços, com API Gateway, isso se torna transparente para os clients.

João Vasconcelos 02/09/2020

API Gateway atua como um reverse-proxy, recebendo todas as requisições a um conjunto de API e mapeia cada chamada a API apropriada. Uma arquitetura de microsserviços é um ótimo caso de uso pois uma única chamada a um único endereço pode lidar com vários serviços ao mesmo tempo. Outros motivos seriam quando é desejado adicionar uma nova camada antes de chamar as apis: sistemas de cobrança, analytics, etc. Com API Gateway podemos facilmente adicionar novas APIs ao sistema

C

João Vasconcelos 02/09/2020

<https://www.redhat.com/en/topics/api/what-does-an-api-gateway-do#:~:text=An%20API%20gateway%20is%20an,return%20the%20appropriate%20result.>

 RO Roberto Oliveira 02/09/2020

é responsável pelo recebimento de todas requisições de uma determinada api trazendo segurança para o usuário.

 EM Elverson Melo 02/09/2020

Um API Gateway é uma ferramenta de gerenciamento de APIs, um ponto de entrada para único para todos os clientes que querem se conectar com os vários serviços back-end,

desacoplando assim essas duas entidades. Ele funciona como um serviço de autenticação, limitação de taxa e roteador de tráfego das APIs, monitoramento, segurança entre outros. Ele deve ser utilizado principalmente em microsserviços e quando o cliente pode representar muitos dispositivos de tipos distintos.

 VG Victor Gabryel 02/09/2020

Usando como exemplo a arquitetura de microsserviços, devido sua modularidade, a API Gateway tem como gerenciar todas as requisições e assim dividir cada requisições em solicitações de cada microsserviço, sendo como um ponto de unificação para cada serviço. É importante usa-la quanto se usa esse tipo de arquitetura, onde existe essa separação entre essas requisições dos microsserviços. Outro ponto, é que o uso da API Gateway é ótimo para o controle e gerenciamento para a segurança de suas API's,+

 VG Victor Gabryel 02/09/2020
contemplando a tríade CID.

 HM Hiro Miyakawa 09/09/2020

É uma ferramenta de gerenciamento de API, onde o cliente acessa os APIs. Isso pode ser útil quando o servidor quer ter mais controle. Se for cobrar por uso, se for restrigir por quantidade de requisição, facilidade na hora de prover novos APIs, etc.

Como fazer um bom Design de API? Qual a relação entre REST e RESTful?

 RE Ricardo Ebbers Carneiro Leão 28/08/2020

REST é um padrão de transferência de dados composto por um verbo HTTP, um header com metadados da requisição, o caminho do recurso no servidor, um corpo de resposta opcional e um status. São denominados RESTful os sistemas que implementam corretamente o padrão REST sem guardar estado e que separam bem os contextos de clientes e servidores. Existem várias formas de desenhar uma API, mas a principal boa prática é usar corretamente as partes do REST e expor recursos através de interfaces idempotent

 RM Rafael Mota Alves 28/08/2020

Para fazer um bom Design de API, eu acho que primeiro de tudo é necessário usar uma ferramenta de documentação como o Swagger, mapear e tratar de forma simples os possíveis cenários de erro (no caso de HTTP usando os status codes corretos) e seguir um padrão de design, como o REST. REST é um padrão de design de APIs, que rege como devem ser usados os métodos e URIs para acessar e modificar recursos de uma dada. Uma API RESTful é uma API que segue os padrões REST.

 DL Danilo Lira 29/08/2020

O design da API é criar uma interface com regras bem definidas, estando essas disponíveis para os usuários. Para um bom design é preciso uma utilização de fácil compreensão e para isso é interessante seguir as melhores práticas de desenvolvimento como a aplicação correta do padrão REST. Esse padrão define as características fundamentais para o desenvolvimento de aplicações Web seguindo as boas práticas. E a relação entre REST e RESTful é direta, pois ser RESTful significa seguir os padrões REST

 HC Heitor Carvalho 29/08/2020

O design de API deve seguir obviamente as regras, padrões e boas práticas já conhecidas tradicionalmente no desenvolvimento de software. Além disso, APIs, pela sua natureza de disponibilizar recursos externos para consumo, devem ter seu desenvolvimento pensado para tal. Por isso devem ser simples de utilizar/consumir, devem ser flexíveis aos vários casos de uso que podem surgir em seu consumo, além de uma documentação robusta para dar suporte aos desenvolvedores que irão utilizá-la.

 HC Heitor Carvalho 29/08/2020

A arquitetura Representational State Transfer é uma arquitetura que dita alguns padrões sobre como recursos devem ser acessados/consumidos/modificados em uma interface web. Ela se baseia em utilizar os verbos HTTP para suas requisições e organizar o acesso aos recursos de forma semanticamente agradável e de fácil utilização, além de não manter estado entre requisições. Serviços web (apis por exemplo) que usam essa arquitetura são ditos serem "RESTful".



 AE Arthur Frade de Araújo 30/08/2020

Arthur Trade de Araújo 30/08/2020

Um bom design de api deve ser feito, primeiramente, observando-se a experiência de mentes que já pensam análise e desenvolvimento de aplicações por um bom tempo. "Reinventar a roda" nunca é um bom caminho. Analisar a experiência de empresas e pessoas experientes é uma ótima opção. Adotar princípios de codificação limpa, como DRY, YAGNI, SOLID são excelentes opções para começar a pensar em design. Bedendo desse tipo de fonte que o padrão REST foi desenvolvido...

AF Arthur Trade de Araújo 30/08/2020

REST é uma espécie de sigla para Representational State Transfer e é composto por uma série de regras de implementação para que o conhecimento sobre o comportamento de uma API seja principalmente intuitivo. A semântica da requisição diz respeito

precisamente à lógica disponibilizada pelo recurso. RESTful é um adjetivo atribuído a uma API que segue os princípios do padrão REST.

IF Igor Fernandes 30/08/2020

Seguindo a estilo de arquitetura REST: Um bom design de API é feito a partir da definição dos limites das funcionalidades de uma aplicação, isso é fase de Decomposição. Há várias formas de fazer essa decomposição: Decompor por capacidade de negócios, Serviço independente, etc. Após isso, definir os nomes dos recursos e os métodos HTTP que serão utilizados em cada um dos recursos. Todas essas definições devem estar documentadas. O RESTful é basicamente dizer que alguma coisa segue o padrão REST.

LC Lucas Cardoso 30/08/2020

Acredito que para um bom design de API é preciso pensar em padronização e na expansibilidade. A padronização é importante para que uma grande gama de clientes possam consumir essa API sem a necessidade de uma introdução de complexidade (por isso REST é importante). A expansibilidade é importante pois serviços evoluem com o tempo e novas funcionalidades devem ser adicionadas sem a necessidade de alterar os clientes (APIs abertas podem nem saber quais clientes a usam)

LB Lucas Barros 30/08/2020

Uma API com um bom design com certeza possui uma interface declarativa e simples, erros bem definidos e coerentes. RESTful é a capacidade de fazer REST, e REST é como um guia de boas práticas para uma API.

VS victor sena de lima attar 31/08/2020

Rest é um guia de comunicação entre serviços. Para fazer um bom design REST deve usar uri + método HTTP + mensagem, entre outros pontos como client-server arquitetura, sem estado, cacheável entre outros. Usando essa padronização de comunicação a web conseguiu promover acesso facilitado a outros serviços pela padronização. Rest é o Padrão, RESTFUL é quando o padrão é utilizado em toda a sua totalidade.

LB Luan Brito 31/08/2020

Acredito que uma boa API possui um padrão bem definido e uma boa documentação para facilitar o uso de seus recursos. Restful é um sistema que usa o padrão Rest.

MG Marcos Galvão 31/08/2020

Uma API bem projetada, consiste, assim como já citaram, inicialmente de uma boa organização de suas interfaces, definindo muito claramente qual a utilidade, limites e padrões de cada uma delas. Assim como Lucas comentou, existir uma padronização na maneira de realizar os requests e suas respectivas resposta, incluindo suas informações de erro, devem ser muito bem estabelecidas e documentadas, isso facilita primeiramente o rápido aprendizado para uso, outrossim a consistência da mesma, pois é

MG Marcos Galvão 31/08/2020

muito fácil saber o que esperar como resposta para cada tipo de request realizado, e não menos importante, permite a escalabilidade da mesma, já que novas funcionalidades ou API's podem ser disponibilizadas seguindo o mesmo padrão e não prejudicando em nada o atual padrão de comunicação já existente. Rest basicamente se trata de um padrão arquitetural que define como a comunicação deve ser estabelecida com as interfaces para, de tal forma, poder acessar ou modificar as informações nelas contidas

MG Marcos Galvão 31/08/2020

já uma uma aplicação Restful se refere a uma aplicação que utiliza a arquitetura Rest

RJ Renato Joaquim de Miranda Ferreira 31/08/2020

Para um bom design de API o objetivo da utilização dela deve ser bem definido, bem como a comunicação entre os sistemas, quais serão os parâmetros passados, segurança utilizada e os padrões entre cada sistema. O REST é um estilo de arquitetura de software bem definido e que segue regras bem determinadas para facilitar a comunicação/programação do sistema, o RESTful é o nome dado às aplicações que seguem todos os padrões REST.

Luca 31/08/2020

Um bom design de API está intrinsecamente ligado a comprehensibilidade da lógica de negócio da aplicação, assim como dos diferentes contextos lógicos, permitindo que elas possam ser desenvolvidas isoladas, com com motivos para alteração bem definidos e responsabilidades "unitárias". Seguindo essa lógica o desenho da interface de comunicação com o

Continuando essa logica, o desenho da interface de comunicação com o produto/serviço (A interface REST) se torna mais simples. Por meio do uso dos padrões REST na comunicação da sua aplicação (sua interface), podemos...

Lucca 31/08/2020

Afirmar que nossa API é RESTful, ou seja, aplica por completo os padrões de arquitetura da World Wide Web em sua comunicação e tratativa.

Gabriel Ramos 31/08/2020

Um bom design de API é algo padronizado, intuitivo e com documentação bem didática. Desenvolvendo métodos para o HTTP padronizados funcionando de forma independente,

pensar de forma que seja possível sem ser muito trabalhoso expandir/aumentar a escalabilidade da API. RESTful é você dizer que segue a ideia do que é o REST.

IN

Izabella Nascimento 31/08/2020

Um bom Design de API deve se preocupar em responder porque utilizar API, quais os resultados que visam alcançar com essa API e como será o planejamento da execução dessa API para obter os resultados esperados, devesse pensar no valor gerado pela API além do próprio valor da API. REST são princípios de arquitetura, como um padrão de design de API, e uma API é chamada de RESTful quando segue os padrões REST.

JS

José Sheldon Brito Fekete 31/08/2020

Um bom design de API é criar uma interface bem definida com regras bem definidas, disponibilizada para que se tenha uma interação com outros sistemas, entregando informações solicitadas ou realizando operações. enquanto REST é uma arquitetura de desenvolvimento que trabalha com protocolo Web, já RESTful é o serviço que utiliza a arquiteta REST.

CP

Claudio Pacheco 01/09/2020

Das características que podemos listar para um bom design de API, as principais concentram-se na noção de que a API deve ser simples de ser utilizada e compreendida - boa documentação ajuda nesse aspecto (sendo esse um outro fator positivo para o design da API), bem como utilizar padrões conhecidos, como o próprio REST. +

CP **Claudio Pacheco** 01/09/2020

REST, portanto, é um padrão de design de APIs. Seus conceitos envolvem seis princípios, como utilizar cache a fim de evitar novas chamadas no servidor e ter a separação cliente/servidor entre as aplicações. Dizemos que um serviço é RESTful quando ele implementa a arquitetura REST.

RP

Ramom Pereira dos Santos Silva 01/09/2020

Antes de qualquer coisa, é necessário analisar quais os impactos que a API a ser desenvolvida trará ao negócio, além de análises sobre os resultados e, se de fato, a API traz valor para o problema que está sendo resolvido. Além disso, estabelecer padrões a serem seguidos para a equipe de desenvolvimento da empresa, organizar o Pipeline de desenvolvimento.

RP **Ramom Pereira dos Santos Silva** 01/09/2020

REST: é um modelo de arquitetura de desenvolvimento de APIs, com regras e princípios bem definidos. RESTful: apenas a capacidade da aplicação de utilizar REST como padrão de API.

TM

Talyta Maria Rosas Pacheco 01/09/2020

O bom design tem a ver com seguir um conjunto de regras definidas para o desenvolvimento, como padrões de nomeação dos endpoints e versionamento de código. Além disso, precisa ser simples e de fácil uso para outros desenvolvedores que farão uso da API. Levando isso em conta, REST é conjunto de padrões e regras de implementação de API e transferência de dados, e RESTFUL faz uso dos padrões REST.

GC

Gabriel Cavalcanti de Melo 02/09/2020

Para fazer um bom design de API, podemos seguir as restrições determinadas pelo estilo arquitetural REST. Como por exemplo: manter as aplicações do servidor e cliente separadas, utilizar cache no cliente para diminuir as comunicações com o servidor e fazer as requisições de forma independente. Uma API RESTful é aquela faz uso completo das restrições presentes no estilo REST.

João Vasconcelos 02/09/2020

Não existe uma implementação padrão de API e sim conceitos centrais como simplicidade e flexibilidade. Quando essas questões forem bem definidas o programador pode começar a pensar em tecnologias ou implementações (pensando, por exemplo, em seguir os padrões REST). REST é um padrão de arquitetura que segue certos princípios, enquanto RESTful é uma forma de dizer que uma api segue os padrões REST: a aplicação x é RESTful

João Vasconcelos 02/09/2020

<https://www.redhat.com/en/topics/api/what-is-api-design#:~:text=API%20design%20refers%20to%20the,products%20to%20their%20partnership%20strategies.>

C

João Vasconcelos 02/09/2020

EM

Elverson Melo 02/09/2020

O desenvolvedor precisa analisar quais as condições ou suposições que cada requisição depende para que seja auto-contida, a API também deve ser simples de usar e ser bem documentada. O padrão REST evoluiu para substituir o SOAP. Ele é um padrão que é livre de estado e que explora os métodos HTTP GET, POST, PUT e DELETE. Ele faz com que uma requisição HTTP contenha todas informações necessárias para identificar o recurso e a ação a ser executada.

EM Elverson Melo 02/09/2020

RESTful é o nome dado as APIs de aplicações que seguem o padrão REST.

RO

Roberto Oliveira 02/09/2020

Um bom design de API está ligado aos padrões de projeto que vc utiliza ao desenvolve-la, um dos padrões mais conhecidos é o REST que inclui um guia de boas práticas, quando um API implementa esse padrão pode ser chamada/se torna API RESTful.

VG

Victor Gabryel 02/09/2020

Para fazer um bom design de API é necessário que ele siga alguns princípios como a simplicidade da API em relação ao uso de formato de dados autenticação e outros, a sua flexibilidade e o tipo de padrão que ela irá usar, como exemplo o REST. A relação entre REST e RESTful é que REST é um estilo arquitetural onde ele compõem de restrições para que as requisições do HTTP possam ser usadas na arquitetura. RESTful são API que são baseadas no estilo REST.

HM

Hiro Miyakawa 09/09/2020

Um bom API foca em "independencia da plataforma" e "evolução do serviço" segundo a Web API Design | Azure. REST é um estilo arquitetural para web services que normalmente utiliza HTTP através da URL. Restful API são princípios de REST aplicados a API.

Casos de Uso de Microsserviços

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar experiências de casos de uso de microsserviços com o time

🔗 Referências

Challenge Based Learning at Windward School
<https://www.youtube.com/watch?v=SDp3xv-WoFw>

framework CBL
<https://www.challengebasedlearning.org/>

🔗 Links compartilhados

GC Gabriel Cavalcanti de Melo
 Service Architectures at Scale: Lessons from Google and eBay
<https://www.infoq.com/presentations/service-arch-scale-google-ebay/>

GD Gabriel D'Luca
 AWS re:Invent 2014 | (ARC308) Nike's Journey into Microservices
<https://www.youtube.com/watch?v=h30ViSEZzW0>

GD Gabriel D'Luca
 What Led Amazon to its Own Microservices Architecture
<https://thenewstack.io/led-amazon-microservices-architecture/>

RO Roberto Oliveira
<https://odetodata.com/2016/04/amazon-microservices-and-the-birth-of-aws-cloud-computing/>
<https://odetodata.com/2016/04/amazon-microservices-and-the-birth-of-aws-cloud-computing/>

RO Roberto Oliveira
<https://www.hys-enterprise.com/blog/why-and-how-netflix-amazon-and-uber-migrated-to-microservices-learn-from-their-experience>
<https://www.hys-enterprise.com/blog/why-and-how-netflix-amazon-and-uber-migrated-to-microservices-learn-from-their-experience>

GC Gabriel Cavalcanti de Melo
 What eBay looks like under the hood
<https://www.infoworld.com/article/3041064/application-development/what-ebay-looks-like-under-the-hood.html>

C

- GC** **Gabriel Cavalcanti de Melo**
The Infrastructure Behind Twitter: Scale
https://blog.twitter.com/engineering/en_us/topics/infrastructure/2017/the-infrastructure-behind-twitter-scale.html
- RM** **Rafael Mota Alves**
Building Twitter's ad platform architecture for the future
https://blog.twitter.com/engineering/en_us/topics/infrastructure/2020/building-twitters-ad-platform-architecture-for-the-future.html
- LC** **Lucas Cardoso**
Enabling Microservices at Orbitz

https://www.youtube.com/watch?v=gV7JSjWDFJc&ab_channel=GOTOConferences&t=2357s
- LC** **Lucas Cardoso**
Airbnb: from Monolith to Service-Oriented
<https://www.infoq.com/presentations/airbnb-soa-migration/>
- RM** **Rafael Mota Alves**
Construindo um arquitetura moderna de microservices na empresa Gilt
<https://www.infoq.com/br/articles/microservices-gilt/>
- AF** **Arthur Frade de Araújo**
Why You Can't Talk About Microservices Without Mentioning Netflix
<https://smartbear.com/blog/develop/why-you-can-t-talk-about-microservices-without-ment/#:~:text=The%20microservices%20architecture%20allowed%20Netflix,rollback%20if%20there%20were%20problems.>
- AF** **Arthur Frade de Araújo**
Entrevista com Fábio Kung
https://www.youtube.com/watch?time_continue=1622&v=uN5l1PKuqJM&feature=emb_title
- RE** **Ricardo Ebbers Carneiro Leão**
Introducing Domain-Oriented Microservice Architecture
<https://eng.uber.com/microservice-architecture/>
- RE** **Ricardo Ebbers Carneiro Leão**
eBay architecture - scalability with agility
<https://www.slideshare.net/tcng3716/ebay-architecture>
- IN** **Izabella Nascimento**
Enabling Microservices @ Orbitz
https://www.youtube.com/watch?v=9LAXaR0_a_E
- IF** **Igor Fernandes**
Segundo o Nike's Journey into Microservices
<https://www.slideshare.net/AmazonWebServices/arc308-nikes-journey-into-microservices-aws-reinvent-2014>
- HM** **Hiro Miyakawa**
Introducing Domain-Oriented Microservice Architecture | Uber Engineering
<https://eng.uber.com/microservice-architecture/>
- HM** **Hiro Miyakawa**
Why We Leverage Multi-tenancy in Uber's Microservice Architecture | Uber Engineering
<https://eng.uber.com/multitenancy-microservice-architecture/>
- HM** **Hiro Miyakawa**
One Team At Uber Is Moving From Microservices To Macroservices
<http://highscalability.com/blog/2020/4/8/one-team-at-uber-is-moving-from-microservices-to-macroservic.html>
- GD** **Gabriel D'Luca**
ESTUDO DE CASO: Amazon e Nike – um estudo de caso sobre microserviços
<https://medium.com/@gdsfvo-contexto-das-aplica%C3%A7%C3%B5es-t%C3%AAM-demonstrado-uma-complexidade-cada-vez-mais-crescente-principalmente-f54929cf78e5>
- SG** **Saulo Guilhermino**
Migração para microserviços: Uma visão Airbnb e Orbitz
<https://medium.com/@sgfl/migra%C3%A7%C3%A3o-para-microservi%C3%A7os-uma-vis%C3%A3o-airbnb-e-orbitz-4952eac65a5d?sk=44e544cf1b10ccdf23f574388e336411>
- RP** **Ramom Pereira dos Santos Silva**
Netflix e Nike: Comparando Casos de Uso de Microserviços
<https://medium.com/@rpss/netflix-e-nike-comparando-casos-de-uso-de-microservi%C3%A7os-6e6a5a596021?sk=dbb6571b874cabaacc7cc12f8919eb07>
- RE** **Ricardo Ebbers Carneiro Leão**
Estudo de caso: Microserviços na Uber e Ebay
<https://www.linkedin.com/pulse/estudo-de-caso-microservi%C2%AD%C2%A7os-na-uber-e-ebay-wellington-oliveira/?trackingId=bAeJLc%2BJTyUFeMdxWyW8GQ%3D%3D>

**Gabriel Cavalcanti de Melo**

A jornada de microserviços no Twitter e eBay

<https://medium.com/@gabrielcavalcanti1310/a-jornada-de-microsservi%C3%A7os-no-twitter-e-ebay-dad1e08b6954>

Estabelecer os casos a serem pesquisados aqui.

**Vinicius Garcia** 03/09/2020

Pessoal, os cases são destas empresas: 1. Netflix (<http://www.netflix.com/>) 2. Uber (<http://www.uber.com/>) 3. Airbnb (<http://www.airbnb.com/>) 4. Twitter (<http://www.twitter.com/>) 5. Amazon (<http://www.amazon.com/>) 6. Gilt (<http://www.gilt.com/>) 7. Orbitz (<http://www.orbitz.com/>) 8. eBay (<http://www.ebay.com/>) 9. Nike (<http://www.nike.com/>)

VG Vinicius Garcia 03/09/2020

1 + 9 [Arthur Frade de Araújo, Matheus Nunes Galdino da Silveira, Ramom Pereira dos Santos Silva] 2 + 8 [Ricardo Ebbers] 3 + 7 [Saulo Guilhermino, Lucas Cardoso, Lucas Barros] 4 + 6 [Gabriel Pessoa, Luan Brito, Rafael Mota, Heitor Carvalho]

VG Vinicius Garcia 03/09/2020

5 + 9 [Roberto Oliveira, Lucca França, Gabriel D'Luca] 6 + 8 [Elverson Soares de Melo, João Vasconcelos de Souza Neto, Josenildo Vicente de Araújo, Renato Joaquim] 7 + 5 [Izabella Nascimento, Victor Gabryel, Daniel Silva] 8 + 4 [Cláudio Pacheco, Gabriel Cavalcanti, Talyta Pacheco]

VG Vinicius Garcia 03/09/2020

9 + 3 [Marcos Galvão, Igor Fernandes e Sheldom Fekete]

VG Vinicius Garcia 04/09/2020

1 + 7 [Victor Sena, Emerson Victor, Danilo Lira, Gabriel Ramos]

RE Ricardo Ebbers Carneiro Leão 09/09/2020

Professor o Wellington Oliveira (wmof) se juntou a mim na análise do Uber vs eBay

**Vinicius Garcia** 09/09/2020

Artigo e Keynote sobre o caso da Airbnb (cortesia de Ricardo Ebbers) no InfoQ sobre a transição deles de um monolito para serviços: <https://www.infoq.com/news/2019/02/airbnb-monolith-migration-soa/>

VG Vinicius Garcia 09/09/2020

Uma excelente palestra com dicas da AirBNB sobre a migração da sua arquitetura: <https://www.youtube.com/watch?v=QQ1QW4boM4Q>

VG Vinicius Garcia 09/09/2020

AirBNB relatando a mudança dessa infraestrutura <https://www.infoq.com/presentations/airbnb-data-infrastructure/> para essa <https://www.infoq.com/br/news/2019/04/airbnb-kubernetes-workflow/>.

**Vinicius Garcia** 09/09/2020

Do Orbitz tem uns slides que já dá pra ter uma noção geral de como foi a transição deles (cortesia de Ricardo Ebbers) : https://gotocon.com/dl/goto-chicago-2016/slides/RickFast_and_SteveHoffman_EnablingMicroservicesAtOrbitz.pdf

VG Vinicius Garcia 09/09/2020

Palestra associada ao link do Orbitz: <https://www.youtube.com/watch?v=gY7JSjWDFJc>

**Vinicius Garcia** 09/09/2020

Palestras da Amazon sobre a cultura e evolução de DevOps na empresa: <https://www.youtube.com/watch?v=mBU3AJ3j1rg> <https://www.youtube.com/watch?v=esEFaY0FDKc>

**Vinicius Garcia** 09/09/2020

Artigo relatando como tudo começou e das dificuldades na migração para MBA da Gilt: <https://dzone.com/articles/gilt-and-microservices>

**Vinicius Garcia** 09/09/2020

Artigo que fala sobre a Uber, Netflix e Amazon: <https://www.hys-enterprise.com/blog/why-and-how-netflix-amazon-and-uber-migrated-to-microservices-learn-from-their-experience/>

**Vinicius Garcia** 09/09/2020

Outro talk sobre o caminho da Amazon de arquitetura monolítica para microserviços: <https://vimeo.com/29719577#at=3>

**Vinicius Garcia** 09/09/2020

Links do blog técnico da Uber: <https://eng.uber.com/multitenancy-microservice-architecture/>
<https://eng.uber.com/building-tincup-microservice-implementation/>

VG **Vinicio Garcia** 09/09/2020

Microservices at Netflix Scale: Principles, Tradeoffs & Lessons Learned
<https://www.youtube.com/watch?v=57UK46qfBLY>

VG **Vinicio Garcia** 09/09/2020

Mastering Chaos - A Netflix Guide to Microservices <https://www.youtube.com/watch?v=CZ3wluvmHeM>

VG **Vinicio Garcia** 09/09/2020

Werner Vogels - Amazon and the lean cloud <https://vimeo.com/29719577>

VG **Vinicio Garcia** 09/09/2020

Twitter: https://blog.twitter.com/engineering/en_us/topics/infrastructure/2017/the-infrastructure-behind-twitter-scale.html

VG **Vinicio Garcia** 09/09/2020

<https://www.infoq.com/presentations/Real-Time-Delivery-Twitter/>

GD **Gabriel D'Luca** 15/09/2020

Link para o estudo de caso (Amazon e Nike): <https://medium.com/@gdsv/o-contexto-das-aplica%C3%A7%C3%B5es-t%C3%A3Am-demonstrado-uma-complexidade-cada-vez-mais-crescente-principalmente-f54929cf78e5>

RM **Rafael Mota Alves** 15/09/2020

Estudo de caso (Twitter e Gilt): <https://medium.com/@rma7/comparando-casos-de-uso-de-microsservi%C3%A7os-twitter-e-gilt-a43726d8a403>

GC **Gabriel Cavalcanti de Melo** 16/09/2020

Link para o estudo de caso (Twitter + eBay): <https://medium.com/@gabrielcavalcanti1310/a-jornada-de-microsservi%C3%A7os-no-twitter-e-ebay-dad1e08b6954>

RP **Ramom Pereira dos Santos Silva** 16/09/2020

Estudo de caso Netflix e Nike: <https://medium.com/@rpss/netflix-e-nike-comparando-casos-de-uso-de-microsservi%C3%A7os-6e6a5a596021>

AR **Antonio Rodrigues** 16/09/2020

Estudo de caso (Uber/Airbnb): https://medium.com/@armn_23684/processo-de-migra%C3%A7%C3%A3o-para-microsservi%C3%A7os-uber-e-airbnb-f13025d7aaf3

IF **Igor Fernandes** 16/09/2020

Estudo de caso (Airbnb/Nike): https://medium.com/@ifc_7168/um-estudo-de-caso-migra%C3%A7%C3%A3o-para-microsservi%C3%A7os-airbnb-nike-d6cc1df69df8

VS **victor sena de lima attar** 16/09/2020

Estudo de caso (Netflix/Orbitz): <https://medium.com/@danilo.lira01/introdu%C3%A7%C3%A3o-a-microsservi%C3%A7os-um-estudo-de-caso-6528a7ff059e>

IN **Izabella Nascimento** 16/09/2020

Estudo de caso (Amazon e Orbitz): <https://medium.com/@ipcn/monol%C3%ADtico-para-microservices-casos-de-uso-da-amazon-e-orbitz-8b1c97bc0ee2>

Identificar similaridades e diferenças no que diz respeito a lições aprendidas?

RO **Roberto Oliveira** 06/09/2020

Uma das lições aprendidas da amazon foi de limitar o numero de desenvolvedores por equipe, no caso deles, eles limitaram esse numero a 10 no maximo, eles chamaram esse metodo de "Metodo de 2 pizzas" onde só era permito ter a quantidade de pessoa que onde 2 pizzas é o suficiente, eles ainda complementam dizendo que o numero 10 é um numero onde não é necessário realizar reuniões complexas para manter todos atualizados do seu progresso, outro regra era a "você constrói você executa"

RO **Roberto Oliveira** 06/09/2020

"desenvolvedores são totalmente responsáveis pelos serviços que constroem - para desenvolver a funcionalidade e para todas as operações, incluindo DevOps."

Lucca 06/09/2020

A nike aprendeu algumas lições com o uso de microsserviços, sendo que a principal foi o quanto mais rápido se torna o processo de desenvolvimento e entrega quando esse é baseado nesse tipo de arquitetura. Eles perceberam também que o investimento no desenvolvimento horizontal de soluções escaláveis baseadas nesse tipo de arquitetura é

desenvolvimento horizontal de soluções escaláveis baseadas nesse tipo de arquitetura é vantajoso a longo prazo.

Lucca 06/09/2020

As empresas tiveram diferentes aprendizados. Enquanto a Amazon percebeu e destacou mais a importância da mudança na dinâmica dos seus times de desenvolvimento, a Nike entendeu uma maior mudança em sua estrutura organizacional como um todo, enaltecedo as melhorias no processo de entrega.



Heitor Carvalho 07/09/2020

GILT: Inicialmente uma aplicação monolítica, enfrentava problemas em picos de acessos, adições de novos produtos e em períodos de compras em massa como Black Friday. Com isso, optaram por migrar para uma arquitetura de microserviços. Uma das principais lições aprendidas foi que apenas distribuindo 'um pouco' os serviços - loja, checkout, separar view dos models e controllers, etc, já resolvia 90% dos problemas de performance que eles tinham. Porém ainda haviam pontos em aberto. (+)



Heitor Carvalho 07/09/2020

Novos serviços criados eram semi monolíticos e ainda haviam muitos controllers/models/jobs no mesmo serviço que podiam ser fragmentados. E o deploy continuava difícil com longos ciclos de integração.



Heitor Carvalho 07/09/2020

Para isso adotaram uso de ferramentas para gerenciamento de deploy, adotaram o padrão Reactive (disponível em The Reactive Manifesto) e também tecnologias como Docker e Amazon AWS, além de um grande foco em code review e em enfatizar a ownership de cada equipe para cada serviço que a mesma era responsável.



Heitor Carvalho 07/09/2020

Principais lições aprendidas - GILT: - Começar distribuindo o sistema aos poucos, para que ele migre lentamente para uma arquitetura de microserviços completa - Enfatizar o ownership das equipes: cada equipe é "dona" de um serviço e a fonte autoritativa de informações relacionadas - Utilizar o padrão Reactive p/ gerenciar dependências - Docker é uma grande ajuda para garantir a configuração dos serviços e a imutabilidade - Foco grande em Code Review - Estar preparado para investir em microserv.



Heitor Carvalho 07/09/2020

Em comparação, não foram encontradas em mesmo teor um relato completo da migração de monolítico para Microserviços no que tange a plataforma do Twitter. No entanto, no blog do twitter existem algumas considerações relevantes sobre o que o time leva em consideração ao construir e implantar novos sistemas de microserviços na arquitetura do Twitter. (+)



Heitor Carvalho 07/09/2020

Aprendizados: - Decomposição do problema não é o bastante. É necessário verificar se ainda há coesão dentro do serviço e se há também boa reusabilidade. - Planeje sua arquitetura para além dos requisitos e specs originais. - Confie em dados e métricas para fazer as decisões técnicas corretas - Uso do Apache MESOS provê grande ganho operacional para crescimento e escalabilidade



Saulo Guilhermino 07/09/2020

Ambas empresas observaram a necessidade de se estabelecer um processo bem definido para essa migração e para os futuros releases de serviços. A Orbitz observou a necessidade de se agilizar o processo eliminando etapas manuais, já o Airbnb observou a necessidade de se envolver o time por completo no processo de migração e também se fornecer ferramental e técnicas para que o desenvolvimento de deploy de serviços seja ágil.



Gabriel Cavalcanti de Melo 07/09/2020

[Twitter + eBay] Uma lição aprendida pelo eBay foi utilizar sinalizadores de recursos para desconectar a implantação de código da implantação de recursos, e poder realizar uma integração contínua. Pois dessa forma, é possível integrar o código na aplicação com o recurso desativado para os clientes e visível apenas para os desenvolvedores e testers.



Gabriel Cavalcanti de Melo 07/09/2020

Já no caso do Twitter, como os sistemas numa arquitetura de microserviços ficam distribuídos, entenderam a necessidade de utilizar orquestradores e containers, pois a comunicação e monitoramento do funcionamento dos serviços foi desafiador para a empresa.



Ricardo Ebbers Carneiro Leão 09/09/2020

A Uber ainda está evoluindo sua DOMA (Domain-Oriented Microservices Architecture). O principal insight que eles tiveram nesse processo de migração a partir de um monolito é que uma arquitetura de microserviços é na verdade um único, grande e distribuído programa, e é possível aplicar os mesmos princípios que seriam aplicados na evolução de qualquer pedaço de software, e a DOMA é apenas uma forma de pensar sobre esses princípios na prática.



Ricardo Ebbers Carneiro Leão 09/09/2020

Já no caso do Ebay já se consegue identificar algo consolidado, O Ebay sabe que para se manter competitivo no setor, é necessário oferecer qualidade em um ritmo muito acelerado. Microservices Permitiu ao Ebay responder a desafios do aumento da

RP

Ramom Pereira dos Santos Silva 09/09/2020

[Netflix e Nike] Netflix iniciou sua migração de Monolítico para uma Arquitetura baseada em Microserviços em 2009, que veio a ser concluída, por completo, em 2011. Basicamente, com essa mudança tiveram 3 grandes lições, alta disponibilidade da plataforma (após ficarem fora do ar por várias horas em 2008), escalabilidade (utilizando diversas estratégias de deploy), com utilização de diversos data centers (AWS), fornecendo seus serviços para diversas plataformas simultaneamente.

RP Ramom Pereira dos Santos Silva 09/09/2020

Além disso, velocidade é outro benefício adquirido com os microserviços, como ele possuem centenas de microserviços, foi possível dividir sua equipe de engenharia em diversos times independentes, que são responsáveis por todo o ciclo de vida dos serviços, proporcionando agilidade e produtividade no processo de desenvolvimento.

RP Ramom Pereira dos Santos Silva 09/09/2020

Sobre a Nike, eles utilizam a Stack da própria Netflix (Netflix OSS), e tiveram grandes benefícios na organização dos times de desenvolvimento, com aumento da entrega e produtividade.

JV

Josenildo Vicente 09/09/2020

O eBay entendeu que para se manter relevante e competitivo era necessário entregar novas features num ritmo ainda mais elevado, dividindo todo o seu ecossistema em microserviços para melhorar a experiência tanto do usuário quanto dos seus desenvolvedores.

JV Josenildo Vicente 09/09/2020

A Gilt percebeu que migrando para os microserviços ficaria mais fácil a realização de implantação contínua, diminuindo o tempo da criação até o mercado, várias iniciativas puderam ser criadas em paralelo, ficando mais fácil criar e destruir novos serviços.

JV Josenildo Vicente 09/09/2020

Uma similaridade entre as duas foi a necessidade de entrega de novos serviços em um menor espaço de tempo.

DL

Danilo Lira 09/09/2020

[Netflix vs Orbitz] As duas empresas decidiram por utilizar microserviços por conta da escalabilidade e tiveram lições aprendidas a partir de erros e dificuldades. Um dos aprendizados da Netflix foi definir o escopo dos testes de dependência de seus microserviços, isso aconteceu por conta da dificuldade que teria ao fazer isso para todos eles, então a empresa decidiu por testar ostensivamente apenas seus serviços mais críticos visando aumentar a disponibilidade da aplicação.

DL Danilo Lira 09/09/2020

Já a Orbitz, aparentemente, lida com uma quantidade menor de times e serviços e seu aprendizado foi mais voltado ao processo de deploy. Utilizando diversas tecnologias como Docker, Jenkins e Mesos a empresa conseguiu reduzir o intervalo de 18 dias para, entre, 1 e 4 dias. Foi um longo processo de adequação e aprendizado para criar esse processo otimizado.

IN

Izabella Nascimento 09/09/2020

Com a mudança de arquitetura a Amazon passou a utilizar de algumas lições para se adaptar. A "equipe de duas pizzas" foi uma regra que tinha como objetivo limitar a quantidade de pessoas trabalhando em um único microserviço, onde notou-se que pequenos times tendem a desenvolver de forma mais produtiva do que uma equipe grande, pois as reuniões tendem a serem mais curtas, e a união para desenvolvimento em prazos menores é maior. A ideia de "você constrói, você executa" foi outra regra que teve...

IN Izabella Nascimento 09/09/2020

como objetivo fazer com que o desenvolvedor tenha contato com a parte de desenvolvimento e a parte de operação, tornando o processo cada vez mais ágil e mais fácil de validar o produto. Na Orbitz houve um processo similar, foi percebido que mesmo usando otimização em diferentes lugares, o problema maior estava no processo de pessoas, que era constituído de times muito grande de desenvolvedores em comparação a time de operação, além de trabalharem em diferentes plataformas, assim surgiu a...

IN Izabella Nascimento 09/09/2020

opção de usar microservices, aliado ao desejo de diminuir o número de pessoas envolvidas no desenvolvimento, utilizando Docker application. Também o aprendizado e conselho de fazer uma delimitação o mais rápido possível sobre as configurações a serem utilizadas nas complexidade dos muitos ambientes de trabalho, e não posteriormente, como foi feito pela empresa.

IF

Igor Fernandes 09/09/2020

[Nike + Airbnb] Uma das lições aprendidas da Nike foi a importância que eles perceberam quanto à separação dos serviços, fazendo o chamado Share-nothing, cada serviço tem seu próprio banco de dados e contratos de API. Isso aumentou a produtividade, o que permitiu que se desenvolvessem mais. O Airbnb tinha um certo problema quanto ao entendimento de relatórios gerais de times diferentes, não havia um padrão.

 **Igor Fernandes** 09/09/2020

Os dashboards eram desenvolvidos pelos integrantes dos times, os alertas de testes por exemplo eram personalizados. Ao utilizarem o IDL e Thrift para construir a base de seus serviços, testes e dashboards, permitiu a companhia perceber a importância de se ter um padrão de relatórios e dashboards para todos os serviços. Essa atitude permitia a mudança de integrantes entre as equipes sem problemas de adaptação.

Identificar similaridades e diferenças no que diz respeito a desafios/obstáculos?

 **Roberto Oliveira** 06/09/2020

Na Amazon no começo tudo ia muito bem, porém depois de um tempo eles perceberam que a taxa de progresso e produtividade estavam diminuindo, após uma análise foi descoberto que as equipes de desenvolvimento gastavam cerca de 70% de seu tempo fazendo trabalhos de operação, chegaram a conclusão que seus desenvolvedores estavam resolvendo problemas parecidos continuamente e por conta própria, por que não tinha infraestrutura interna comum que pudessem usar.

 **Lucca** 06/09/2020

A Nike, ao iniciar sua migração para microsserviços, enfrentou alguns problemas, como por exemplo a necessidade de integração de grandes partes legadas do sistema à nova arquitetura, a manutenção, mesmo que parcial, do uso de um sistema de notificações e a burocracia corporativa associada aos seus processos de desenvolvimento, atrapalhando em parte a implementação de um sistema de entrega contínua.

 **Gabriel D'Luca** 06/09/2020

Em ambos os casos, podemos observar burocracias no próprio processo de desenvolvimento, o que demandava um esforço maior por parte dos desenvolvedores. Por outro lado, os desafios encontrados da Amazon ocorreram APÓS a implantação de uma arquitetura baseada em microsserviços (pouca de infraestrutura interna comum), enquanto a Nike enfrentou desafios já DURANTE o próprio processo de implementação.

 **Saulo Guilhermino** 07/09/2020

Ambas empresas se depararam com desafios técnicos e não-técnicos para realizar o processo de migração: o Airbnb precisou adaptar os hábitos de todo o seu time de engenharia, visto que até então estavam acostumados apenas a desenvolver em um sistema monolítico.

 **Lucas Barros** 07/09/2020

Já a Orbitz se deparou com a necessidade de decompor um serviço grande em diversos outros serviços menores mantendo uma compatibilidade com os serviços já existentes e desenvolver um processo de deploy que não necessitasse de trabalho manual, o que corrobora com a necessidade do Airbnb de se desenvolver/encontrar sistemas, ferramentas ou estratégias de se agilizar o deploy.

 **Lucas Cardoso** 07/09/2020

Outra similaridade foi o fato das migrações de sistemas monolíticos para microsserviços de ambos ter sido gradual e cheia de tentativas e erros. Interessante perceber como ambos tiveram soluções e problemas diferentes e específicos, isso acabou refletindo nas respectivas Stacks de tecnologia utilizadas, com Airbnb usando kubernetes e Orbitz usando Apache Mesos + Marathon, por exemplo

 **Gabriel Cavalcanti de Melo** 07/09/2020

[Twitter + eBay] Com base na entrevista com o engenheiro do Twitter em 2017, William Morgan, (<https://dzone.com/articles/experience-at-twitter-improves-runtime-between-mic>), relatou dificuldades no início do desenvolvimento com microsserviços. Por ser um sistema distribuído, é necessário deixar tudo funcionando de forma auto-coordenada, e esse processo foi muito custoso tanto em tempo quanto dinheiro (demoraram cerca de 5 anos).

 **Gabriel Cavalcanti de Melo** 07/09/2020

Com os microsserviços também se faz necessário comunicação em tempo de execução, o que se torna algo complexo mediante necessidades de entender essa comunicação, como monitorar e controlar em tempo de execução. Além de entender bem o funcionamento, devido a forma de comunicação, se algo falhar pode percorrer até derrubar o DataCenter.

 **Gabriel Cavalcanti de Melo** 07/09/2020

Já para eBay o principal desafio também foi relacionado ao monitoramento, como no twitter, e com relação aos módulos compartilhados entre os serviços, pois como o time de desenvolvimento é dividido e cada um tem uma responsabilidade e prioridade de implementação, caso haja uma mudança no módulo compartilhado, acaba ficando disponível no ambiente de produção em momentos diferentes.

 **Rafael Mota Alves** 07/09/2020

Ao migrar suas arquiteturas para microsserviços, um dos principais desafios tanto para a Gilt quanto o Twitter, foi em como dividir sua aplicação monolítica em serviços. Para a Gilt, na primeira tentativa foram criados o que foram chamados de macroserviços, que eram essencialmente aplicações monolíticas distribuídas, o que resolveu vários problemas de escalabilidade, mas introduziu outros problemas como: longos ciclos de integração e carência

de um responsável pelo código.

 **Rafael Mota Alves** 07/09/2020

Esse problema no caso da Gilt, foi resolvido quebrando os macroserviços, em verdadeiros microserviços, o que tornou possível dividir os serviços entre os times e tornar o processo de desenvolvimento menos burocráticos. Já no Twitter, a primeira abordagem de divisão de serviços da plataforma de ADs foi pensada numa divisão por produtos, o que ocasionou uma grande duplicação de código, e uma baixa coesão, pois os serviços acumulavam muitas responsabilidades.

 **Rafael Mota Alves** 07/09/2020

A solução foi separar serviços responsáveis por processos que são usados pelos diferentes produtos, além de separar os produtos. Dando a responsabilidade desses serviços para times específicos, especialistas nos domínios (processos de seleção de ADs por ML...).

 **Rafael Mota Alves** 07/09/2020

Apesar das dificuldades terem sido parecidas, me parece que o Twitter não teve tantas dificuldades em relação a divisão dos serviços entre as equipes, como o Gilt teve, me parece que isso se deve ao Twitter já ter times separados por Produtos antes da migração.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

A Uber passou a ter vários problemas associados com o crescimento de complexidade do sistema de acordo com o crescimento da empresa. Com microserviços se troca um codebase monolítico e singular por várias caixas pretas cujas funcionalidades podem mudar a qualquer momento e podem causar comportamentos inesperados facilmente.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

Entender dependências entre serviços é uma tarefa árdua e chamadas entre serviços podem percorrer por várias camadas. Um spike de latência em uma das dependências pode causar uma casacata de problemas no resto da stack de chamada. Para um engenheiro desenvolver uma feature frequentemente ele precisa trabalhar em múltiplos microserviços, cada um pertencente a indivíduos e times distintos, o que requer bastante colaboração através de meetings, design e code reviews.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

O resultado é uma experiência de desenvolvimento mais lenta, instabilidade para donos de serviços, migrações dolorosas, etc. O Ebay assim com a Uber teve como principal motivo de migração para Microservices crescente complexidade dos algoritmos do sistema, principalmente nos filtros, buscas e recomendações para os usuários.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

Além disso, um dos principais desafios do eBay foi o fluxo de dados: Em um dia normal, os sistemas de TI do eBay tiveram que lidar com um tráfego massivo, como 75 bilhões de chamadas de banco de dados, 4 bilhões de visualizações de página e 250 bilhões de consultas de pesquisa

 **Ramom Pereira dos Santos Silva** 09/09/2020

No início de sua migração, a Netflix utilizava seus próprios datacenters, porém tiveram muita dificuldade com latência de rede, o que ocasionou para a mudança para a AWS. Mesmo na Cloud, tiveram diversos problemas de carga, falha em zonas de disponibilidade (teve o famoso caso em 2011, quando uma az da AWS caiu e muitos serviços ficaram fora do ar no mundo inteiro), desempenho baixo, o que provocou a criação de diversas tecnologias que automatizava a resolução de problemas dessa natureza.

 **Ramom Pereira dos Santos Silva** 09/09/2020

Sobre a Nike, integração dessa nova integração com alguns sistemas legados que possuíam, além de uma baixa na velocidade de desenvolvimento no início do processo, dado que muitos processos eram manuais (deploy de banco de dados, por exemplo), também tiveram problemas burocráticos da empresa.

Gabriel Ramos 09/09/2020

[Netflix vs Orbitz] Falando inicialmente sobre as similaridades, a mais explícita de todas é que ambas as empresas inicialmente utilizavam uma arquitetura monolítica e migraram para uma arquitetura de micro serviço, e um dos gatilhos em comum para a mudança para a Netflix e Orbitz é a massiva quantidade de informação e usuários com suas milhares de instâncias, ainda nesse ponto vale destacar uma diferença de obstáculo entre as companhias, a Orbitz durante sua arquitetura monolítica dependia de +

Gabriel Ramos 09/09/2020

muito processo manual para o "deploy"/"code to production". Um desafio, muito único para a Netflix foi ser uma das empresas pioneiras a migrarem esse tipo de arquitetura - antes mesmo de microserviço ser introduzido -

 **Josenildo Vicente** 09/09/2020

Na Gilt havia o problema da escalabilidade em que foi solucionado após a migração, mas continuaram outros problemas como ciclos de integração longos que dificultavam o deploy do site e novos serviços pareciam mini serviços monolíticos com muita gente trabalhando no mesmo código, outro problema foi uma maior dificuldade em realizar testes funcionais, principalmente de dependências entre os microserviços

 **Josenildo Vicente** 09/09/2020

O eBay começou numa arquitetura monolítica baseada em Perl que depois continuou monolítica baseada em C++ contendo uma única DLL com 3,4 milhões de linhas de código, o que gerava manutenção e escalabilidade muito mais desafiante uma vez que milhões e milhões de requisições eram feitas diariamente todos os dias.

 **Josenildo Vicente** 09/09/2020

A semelhança entre eles foi a dificuldade da escalabilidade antes de implementar os microserviços e a diferença foi da Gilt ter vindo vários desafios novos relacionados a microserviços após a migração.



Izabella Nascimento 09/09/2020

A Amazon com o passar do tempo viu a necessidade de mudar seu sistema para uma arquitetura onde existisse maior flexibilidade devido a limitação que estava encontrando na sua arquitetura monolítica. Com isso existia uma chance onde durante o processo de desenvolvimento de dividir seu sistema em diversos microserviços houvesse uma bagunça que poderia quebrar todo o sistema, então regras como a "equipe de duas pizzas" e "você constrói, você executa" foram definidas para uma maior clareza...

 **Izabella Nascimento** 09/09/2020

durante os processos, mas a produtividade diminuiu devido ao gasto com trabalho operacional, onde diversas atividades repetidas eram feitas continuamente e com isso a AWS ajudou a automatizar essas tarefas e com isso escalar os serviços de acordo com necessidades. Já na Orbitz o desafio foi saber como lidar com falhas, também quando as máquinas virtuais eram movidas e o Docker não tinha conhecimento que podia estar rodando em outra máquina virtual, sendo resolvido com a construção de...

 **Izabella Nascimento** 09/09/2020

scripts Python para realizar o gerenciamento dos deploys simultâneos.



Igor Fernandes 09/09/2020

[Nike + Airbnb] O Airbnb começou a sofrer graves problemas nos deploys do seu monólito a partir do momento que o time cresceu e a adição das novas funcionalidades começaram a gerar uma bagunça. Tal fator aumentou consideravelmente o tempo de desenvolvimento, testes e deploy, ademais com a adição de novas features a única database se tornava cada vez menos consistente.

 **Igor Fernandes** 09/09/2020

A Nike também tinha problemas com o tempo de realização o deploy de novas funcionalidades. Além disso, sofria com os deploys de banco de dados manuais e um grande processo de aprovação para qualquer nova mudança. Havia muitos produtos dentro do monólito (cerca de 14) para os nove times que trabalhavam, e todos esses produtos estavam dentro da mesma base de código, pequenas mudanças causavam muitas consequências indesejáveis.

Identificar similaridades e diferenças no que diz respeito a propósitos/objetivos/motivação?



Roberto Oliveira 06/09/2020

Umas das grandes motivações da Amazon para migrar para microserviços foi a sobrecarga no seu processo de ciclo de vida de desenvolvimento, assim como toda startup elas começaram com um sistema monolito por crescer mais rápido porem com um numero maior de desenvolvedores esse processo de tornou mais demorado, outro fator que contribuiu para esta migração foi a complexidade por trás dessa base de código, adicionar novos recursos ao sistema se tornou-se difícil e arriscado.

 **Gabriel D'Luca** 06/09/2020

No caso da Nike, a principal motivação também foi decorrente de gargalos trazidos por uma arquitetura monolítica: o deploy era demorado (meses para uma feature chegar à produção), e adicionar novas features se tornou algo bem arriscado, pois faltava qualidade no processo (muitos defeitos passavam no deploy). Além disso, pelos riscos trazidos pela falta de qualidade no caminho à produção, era necessário ter uma equipe bem maior de DEVs (gastar mais dinheiro!!!) pra remediar os bugs encontrados.

Lucca 06/09/2020

Ambas as empresas iniciaram a migração de suas aplicações para uma arquitetura de microserviços devido as perdas financeiras relacionadas ao elevado tempo para deploy de novas features, assim como na manutenção de diversos times e desenvolvedores, que em sua maior parte do tempo estavam trabalhando para corrigir problemas ou integrar funcionalidades ao invés de estarem desenvolvendo de fato as novas soluções.



Gabriel Cavalcanti de Melo 07/09/2020

[Twitter + eBay] Ambos apresentavam desafios ligados a deixar a aplicação disponível ao mesmo tempo que precisavam implementar novas funcionalidades e com quantidade de acesso de usuários maior que o esperado, mas muitas vezes a aplicação caía. Por esses motivos, decidiram mudar gradualmente para uma arquitetura de microserviços.

 **Gabriel Cavalcanti de Melo** 07/09/2020

No caso do Twitter, para lidar com o enorme acesso e evitar o aparecimento frequente da tela de erro customizada pela companhia, escalavam a aplicação de forma vertical, ou seja, aumentavam a capacidade dos datacenters em memória, firewall etc, e como o tráfego crescia mais rápido do que era possível fazer uma alteração na arquitetura de um data center, foi necessária a implantação de uma arquitetura mais escalável.

 **Gabriel Cavalcanti de Melo** 07/09/2020

Com relação ao eBay, para se destacar no mercado se deparou com desafios relacionados a melhorar a flexibilidade de implantação e encurtar ciclos de lançamento de novas features, portanto optaram por utilizar a arquitetura de microserviços. Além disso, a camada de dados era compartilhada para todas as funcionalidades do sistema, e isso se

torna um problema quando se tem 800 milhões de produtos cadastrados. Um erro em alguma funcionalidade poderia quebrar todo o sistema.

 **Lucas Cardoso** 07/09/2020

O Airbnb conseguiu resumir suas necessidades em dois tópicos: Melhoria no deploy e Melhoria no Code Ownership, pois um processo de deploy de um sistema monolítico chegava a durar uma média de 15h por semana, além de que, um código "único" torna difícil delimitar as responsabilidades de cada engenheiro.

 **Saulo Guilhermino** 07/09/2020

A Orbitz também enfrentava problemas relacionados a duração do deploy e coordenação entre os engenheiros de cada time, mas também necessitava aprimorar a escalabilidade e manutenibilidade do código, visto que o time de desenvolvimento utilizava ferramentas de deploy diferentes do time de operação e esta diferença de ferramentas fez as pessoas não criarem mais serviços, chegou um ponto que existiam vários monólitos.

 **Lucas Barros** 07/09/2020

O surgimento de novos clientes e serviços oferecidos para a Orbitz mostrou as limitações de sua arquitetura monolítica, que dificultava ainda mais sua escalabilidade.

 **Luan Brito** 07/09/2020

[Gilt - Twitter] Similaridades: redução do acoplamento entre as atividades/times, reduzir a complexidade do sistema inicial, possibilitar uma maior escalabilidade das aplicações.

 **Luan Brito** 07/09/2020

Diferenças: Gilt tinha uma necessidade específica de picos de acesso, então o foco principal era uma necessidade "externa", enquanto no Twitter era mais uma necessidade de desacoplamento para desenvolvimento. Mas evidente que ambos se beneficiaram de outras formas além da qual era seu foco principal .

 **Arthur Frade de Araújo** 09/09/2020

[1 + 9] Motivações comuns à Netflix e à Nike no que se refere a adoção da arquitetura de microserviços são a necessidade de aprimorar: Disponibilidade, Escala e Velocidade de seus produtos digitais. A diferença entre elas se deve ao fato de que uma (Netflix) possui uma demanda digital muito mais robusta que a outra (Nike), o que torna as arquiteturas muito distantes em termos de complexidade. Apesar das motivações serem muito parecidas, o volume de demanda irá exigir soluções mais arrojadas.

 **Arthur Frade de Araújo** 09/09/2020

Para se ter uma ideia, hoje a Netflix conta com uma estrutura de deploy que permite fornecer versões diferentes de determinados serviços por perfil de usuário, ou seja, pode ser que dois amigos brasileiros, assistindo Netflix no mesmo momento em suas casas, estejam consumindo versões diferentes do serviço de vitrine, por exemplo. Essas informações podem ser conferidas no link "Entrevista com Fábio Kung" (a partir do minuto 27:00).

 **Arthur Frade de Araújo** 09/09/2020

Enquanto a Netflix hoje, vincula estratégias de deploy com estratégias de negócio, a Nike vem usufruindo das características mais tradicionais da arquitetura distribuída. Em palestra ao AWS re: Invent, Jason Robey comenta que a equipe da Nike tem evoluído muito sua integração contínua, qualidade de código e disponibilidade. Inclusive, utiliza o Netflix OSS para o desenvolvimento de seus serviços. As empresas portanto seguem uma mesma filosofia, porém com graus de maturidade distintos.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

motivação da Uber para adotar microserviços partiu de um cenário com dois monolitos cheios de problemas operacionais em 2013: * Riscos de disponibilidade; * Deploys arriscados, que consumiam bastante tempo e frequentemente precisava de rollbacks; * Separação de responsabilidades pobre; * Execução ineficiente, frequentemente acoplando times.

 **Ricardo Ebbers Carneiro Leão** 09/09/2020

Já no caso do eBay, a motivação partiu da competição de espaço no setor: * Em 2011 o crescimento do e-commerce era intenso, era necessário ter uma performance nos buscadores * Sistema de Recomendação e propaganda. * Riscos de disponibilidade (Semelhante ao Uber) * Múltiplos Formatos de Dados

EV

Emerson Victor 09/09/2020

A orbitz tinha um sistema baseado na conexão empresa|business layer|back-end. Com o crescimento do negócio e junção de novas empresas e sistemas back-end à plataforma a orbitz percebeu que o sistema não era escalável (ex: empresa A só precisava de acesso ao back-end A, mas, na estrutura antiga, tinham acesso a todos os back-ends). Já no caso da Netflix, eles decidiram mudar devido ao grande crescimento de dados e informações de usuário, tornando difícil armazená-los em seus data centers da época.

JV

Josenildo Vicente 09/09/2020

No eBay o pensamento em torno da migração era que para se manter competitivo é necessário entregar ainda mais features com qualidade e inovações ainda mais rápido. Além de conforme mais mudanças aconteciam no sistema monolítico, maior a complexidade do sistema e mais difícil a manutenção, portanto a migração se fazia necessária para além dos pontos descritos acima, melhorar a produtividade dos desenvolvedores e a experiência do usuário usando o sistema.

Josenildo Vicente 09/09/2020

A Gilt estava sofrendo com a aplicação monolítica, ela não escalava como desejado, a demanda era maior do que o site suportava e o código da aplicação monolítica ficou gigante e complexo. Os microserviços apareceram como uma opção para melhorar a escalabilidade e resolver os problemas que foram originados na tentativa de resolver a escalabilidade com a aplicação monolítica.

Josenildo Vicente 09/09/2020

Uns objetivos semelhantes foram melhorar a produtividade dos desenvolvedores, facilitar a escalabilidade e melhorar a experiência do usuário.

IN

Izabella Nascimento 09/09/2020

Devido ao crescimento da Amazon e o seu problema urgente de escalabilidade, com os gargalos que eles estavam encontrando na sua arquitetura com os problemas de demora de implantação, grandes bancos de dados e perdas financeiras foram motivações para que os arquitetos da Amazon encontrasse uma forma de automatizar esses processos, e que permitisse escalar todos os serviços de acordo com necessidades comerciais, resultando na mudança do sistema monolítico para a de microserviços com uso da nuvem.

Izabella Nascimento 09/09/2020

Com a Orbitz ocorreu de forma semelhante, com a crescente demanda das companhias aéreas querendo utilizar a plataforma da Orbitz, eles perceberam que não seria possível escalar e após 4 anos decidiram mudaram para uma arquitetura de serviços, pois eles possuíam os times dependendo um dos outros, pouca elasticidade, com tudo rodando no mesmo servidor, além de serem muitos times trabalhando em cima de um monolito, haviam commits de diversas origens, causando falhas que só eram descobertas no...

Izabella Nascimento 09/09/2020

fim da produção e encontrar a origem desses erros também se tornava uma tarefa difícil. Foram então buscar outras formas de modelagem mas a maioria não eram escaláveis e nem possuíam um caminho de desenvolvimento amigável, assim escolheram microserviços.

IF

Igor Fernandes 09/09/2020

[Nike + Airbnb] Ambos tinha o propósito de migrar para uma arquitetura SOA para garantir independência de deploy, desenvolvimento e escalabilidade. De tal forma o time de engenheiros não precisariam se preocupar com boa parte das complicações que vinham sendo enfrentadas. A visão era que a aplicação somente entregasse ao cliente o que ele realmente precisava, reduzindo a quantidade de chamadas desnecessárias na aplicação.

Igor Fernandes 09/09/2020

Usando microserviços uma rota de recurso da aplicação poderia ser dividida em serviços distintos. A Nike e o Arbnb se diferenciavam quanto ao sistema legado. Enquanto o Airbnb usou metaprogramação para manter a compatibilidade com os sistemas legados, somente enquanto os serviços ainda estavam sendo desenvolvidos. A Nike criou uma API Gateway para gerenciar as chamadas para o sistema legado, sem necessariamente planejar o desligamento do sistema legado.

Identificar similaridades e diferenças no que diz respeito a métricas e gestão do sucesso?

GD

Gabriel D'Luca 06/09/2020

No caso da Amazon, a mudança para um cenário de microserviços os permitiu escalar o número de servidores para mais ou para menos quando necessário, reduzindo o número e a duração de downtime e economizando dinheiro (\$\$\$). A arquitetura baseada em microserviços também permitiu que eles transacionassem para um estado de implantação contínua (continuous deployment) e agora os engenheiros da Amazon implantam o código em dezenas de segundos.

Lucca 06/09/2020

No caso da Nike, após a implementação da infraestrutura em microserviços, o tempo para o lançamento de novas features caiu de meses para horas, garantindo maior retorno

para o lançamento de novas features caiu de meses para horas, garantindo maior retorno financeiro, junto a um processo de validação de qualidade que se tornou 100% automatizado. As equipes de desenvolvimento também foram reduzidas e, sendo que mesmo assim, a qualidade do deploy foi elevada e o tempo de indisponibilidade pode ser zerado.

RO Roberto Oliveira 06/09/2020

pelos fatos encontrados foi possível perceber que o ganho em ambos os casos foi a diminuição do tempo de lançamento para produção.

RO Roberto Oliveira 06/09/2020

no caso da amazon eles tbm pensaram alem e desenvolveram um novo modelo de infraestrutura "on demand"

SG Saulo Guilhermino 07/09/2020

O Airbnb mediu seu sucesso a partir dos problemas: Uma expressiva redução no tempo semanal de deploy (15h para 115min) e um aumento na taxa de correção de bugs demonstraram que a migração de um sistema monolítico para distribuído foi uma boa escolha. A Orbitz também mediu seu tempo de deploy, mas inclui também todo o processo de desenvolvimento de uma nova feature e a resiliência do sistema.

GC Gabriel Cavalcanti de Melo 08/09/2020

[Twitter + eBay] Em ambos os casos o sucesso do uso de microsserviços afetaram positivamente o processo de desenvolvimento das funcionalidade, e também tornou a aplicação mais escalável. No Twitter, o aparecimento da tela de erro quando o número de acesso era alto se tornou bem menos frequente. Além disso, com a distribuição dos serviços, a equipe possui especialistas para cada parte lógica e funcionalidade da plataforma, isolando portanto as falhas e desenvolvimento.

GC Gabriel Cavalcanti de Melo 08/09/2020

Para o eBay, como disse o ex CTO Steven Fisher, com a utilização de uma arquitetura de microsserviços, ficou mais fácil e rápido adicionar novas funcionalidades e desacoplar dependências desnecessárias.

Gabriel Pessoa 08/09/2020

Tanto o Gilt quanto o Twitter tiveram sucessos bem parecidos, um aumento na velocidade de desenvolvimento pelo fato de não ter dependencia de outros times, e sobre ser mais facil entender como funciona um microserviço isolado do que algo dentro de um monolito. O Gilt também comenta como é mais facil delegar a responsabilidade de monitorar um serviço quando cada equipa é dona de um serviço.

RE Ricardo Ebbes Carneiro Leão 09/09/2020

Embora a DOMA da Uber ainda é recente, eles já conseguem perceber benefícios claros a partir da adoção: Redução geral da complexidade do sistema; Experiência dos desenvolvedores simplificada; Prazos de entregas de algumas features reduzidas de três dias para três horas; Redução de tempo de onboarding de times em 25-50%; Utilização de gateways permite que migrações sejam feitas de forma transparente para os consumidores; Plataformas usando a DOMA são muito mais extensíveis e fáceis de manter.

RE Ricardo Ebbes Carneiro Leão 09/09/2020

Muito Semelhante ao Uber, mas com mais tempo de implementação (2011) o eBay teve como resultado: Redução da complexidade Redução do tempo de entrega e de implantação das features Modularização das soluções

AF Arthur Frade de Araújo 09/09/2020

Tanto a Netflix quanto a Nike obtiveram resultados positivos após a adoção da estratégia de microsserviços. Por ter sido pioneira nessa área, a Netflix hoje é líder técnica e referência mundial em aplicações distribuídas e cloud based. Além disso é uma das principais plataformas de streamming de vídeo do mundo, graças ao modelo dinâmico, resiliente e expansível. A Nike, como sempre fez, segue mantendo-se incremental e inovadora, a adoção da estratégia melhorou a disponibilidade de serviços...

AF Arthur Frade de Araújo 09/09/2020

aumentou o número de deploys no mês e ainda possibilitou a expansão do número de produtos digitais que a empresa fornece hoje. Graças a uma estratégia moderna de presença digital, hoje a Nike além de fornecer produtos voltados à saúde e exercícios, promove conhecimento e acompanhamento da evolução dos atletas através das plataformas NRC, NTC, entre outras.

AF Arthur Frade de Araújo 09/09/2020

grupo [1 + 9]

João Vasconcelos 09/09/2020

A adoção de uma arquitetura de microsserviços trouxe benefícios a ambas e um dos pontos comuns foi a possibilidade de criar times independentes, onde cada um dos mesmos é responsável por um serviço. Notei que, além disso, as vantagens são mais relacionadas ao foco da empresa: enquanto o ebay comentou mais sobre a melhoria da experiência do usuário, tanto em mobile como web, a gilt comentou sobre iniciativas paralelas, diversificação de tecnologias e degradar serviços de uma forma mais segura.

VS

victor sena de lima attar 09/09/2020

A netflix e a orbitz obtiveram grande sucesso na implantação dos seus sistemas, houve melhora nas métricas de velocidade, escalabilidade, interoperabilidade, consistência, velocidade de desenvolvimento, velocidade de implantação de novas mudanças, entre outras métricas. A netflix é focada em larga escala mundial de velocidade e interoperabilidade entre os sistemas internos que estão sempre se conectando entre si.

vs **victor sena de lima attar** 09/09/2020

Já na orbitz a construção do sistema de microserviços foi baseada para suportar a conexão com vários serviços externos, com lógicas diferentes provendo serviços com particularidades diferentes para aplicações diferentes.

IN

Izabella Nascimento 09/09/2020

Na Orbitz a gestão do sucesso com a utilização de microservices foi medida de forma interna, com o sucesso da migração para microserviços com poucas dificuldades, e o ganho de mais parceiros, possuindo várias marcas seja em sites e serviços web, além de serviços de Back end e mais de 500 aplicações rodando, fazendo ocorrer a diminuição drástica do tempo de lançamentos, de 4 por ano em 2010, para de 1 a 4 dias atualmente.

IN **Izabella Nascimento** 09/09/2020

Ocorrendo de forma semelhante com a Amazon com a arquitetura de microservices e com o uso da nuvem fez com que pontos como escalar os serviços de acordo com as necessidades comerciais resultasse na entrega contínua, onde exista uma rapidez com o desenvolvimento e as implantações ocorram com mais flexibilidades.

IF

Igor Fernandes 09/09/2020

[Nike + Airbnb] Segundo o Nike's Journey into Microservices: A cada ano eles analisavam em alto nível o quanto eles estavam entendendo do produto deles (in-house product understanding), o tempo de deploy de novas funcionalidades, perspectiva do produto (caótico ou estável), ciclo de trabalho dos engenheiros (24/7, razoável e recompensador) e qualidade do deploy (validação manual ou automático).

IF **Igor Fernandes** 09/09/2020

O Airbnb assim como a Nike percebeu uma certa agilidade quanto aos deploys de novas funcionalidades, indo de 3 mil deploys/semana para 10 mil deploys/semana. Além disso, percebeu que diferentemente de antes a adição de novos funcionários nos serviços da plataforma não demonstrava afetar a curva de produtividade.

💡 Aplicando Microservices

Tópicos [de aprendizagem]

Os microserviços são bons, mas também podem ser um mal se não forem concebidos adequadamente. Interpretações erradas podem levar a falhas irrecuperáveis!

🔗 Referências

"Microservices," in IEEE Software, vol. 35, no. 3, pp. 96-100, May/June 2018
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8354423&isnumber=8354413>

Why Segment Moved from Microservices to a Monolith
<https://www.programmableweb.com/news/why-segment-moved-microservices-to-monolith/else-where-web-case-study/2019/03/17>

Consumer Driven Contracts - a curated list
<https://gist.github.com/michaellihs/e302a4a8450ca28cbd51f56968157e45>

Robustness Principle Is WRONG? | Code Walks 061
<https://www.youtube.com/watch?v=1B4KjAhQJoQ>

🔗 Links compartilhados

AF **Arthur Frade de Araújo**
 Microservices funny hq
https://www.reddit.com/r/ProgrammerHumor/comments/cu2cga/distributed_computing_is_hard_yall/

AF **Arthur Frade de Araújo**
 Apache Kafka
<https://kafka.apache.org/>

AF **Arthur Frade de Araújo**
 Azure Service Bus
<https://docs.microsoft.com/pt-br/azure/service-bus-messaging/service-bus-messaging-overview>

RM **Rafael Mota Alves**
 Consumer Driven Contracts
<https://martinfowler.com/articles/consumerDrivenContracts.html>

-  **Rafael Mota Alves**
Technology Radar - Consumer Driven Contract
<https://www.thoughtworks.com/pt/radar/techniques/consumer-driven-contract-testing>
-  **Rafael Mota Alves**
gRPC
<https://grpc.io/>
-  **Rafael Mota Alves**
Swagger
<https://swagger.io/>
-  **Rafael Mota Alves**
API Gateway
<https://aws.amazon.com/pt/api-gateway/>
-  **Rafael Mota Alves**
Rabbit MQ
<https://www.rabbitmq.com/>
-  **Emerson Victor**
Mudança paralela
<https://www.thoughtworks.com/pt/insights/blog/mudan%C3%A7a-paralela>
-  **Saulo Guilhermino**
Segurança em Arquiteturas de Microsserviços
<https://blogbrasil.westcon.com/seguranca-em-arquiteturas-de-microsservicos>
-  **Danilo Lira**
Arquitetura de micro-serviços na prática: modelagem e rastreabilidade dos serviços
<https://medium.com/olxbr-tech/arquitetura-de-micro-serviços-na-prática-modelagem-e-rastreabilidade-dos-serviços-21e0822a63fb>
-  **Danilo Lira**
Introduction to Microservices Testing and Consumer Driven Contract Testing with PACT
<https://www.novatec-gmbh.de/en/blog/introduction-microservices-testing-consumer-driven-contract-testing-pact/>
-  **Heitor Carvalho**
Guidelines for API Documentation
<https://www.mulesoft.com/resources/api/guidelines-api-documentation>
-  **Heitor Carvalho**
JSON:API - A Specification for Building APIs in JSON
<https://jsonapi.org/>
-  **Heitor Carvalho**
[VIDEO] CorkDev: Consumer-Driven Contracts: Avoid Microservices Integration Hell!
<https://www.youtube.com/watch?v=rHDyvnp5x3w>
-  **Gabriel Cavalcanti de Melo**
How to test Microservices with Consumer-Driven Contracts?
<https://hackernoon.com/how-to-test-microservices-with-consumer-driven-contracts-9bf5c2c05349>
-  **Talyta Maria Rosas Pacheco**
Introduction to Consumer-Driven Contract Testing
<https://medium.com/kreuzwerker-gmbh/introduction-to-consumer-driven-contract-testing-3a130c8c2ea0>
-  **Lucas Cardoso**
API Documentation Solutions
<https://medium.com/technical-writing-is-easy/api-documentation-solutions-d3719af2780f>
-  **Arthur Frade de Araújo**
Marshalling
<https://pt.wikipedia.org/wiki/Marshalling#:~:text=Marshalling%20%C3%A9%20similar%20%C3%A0%20serializa%C3%A7%C3%A3o,neste%20caso%2C%20>
-  **Claudio Pacheco**
Postel's Law
<https://devopedia.org/postel-s-law>
-  **Marcos Galvão**
Princípio Robustez
https://pt.qwe.wiki/wiki/Robustness_principle
-  **Roberto Oliveira**
10 desafios
<https://vertigo.com.br/microsservicos-10-novos-desafios-em-relacao-a-seguranca/>
-  **Roberto Oliveira**
CDC
<https://medium.com/better-practices/consumer-driven-contract-testing-using-postman-f3580dha5370>

- HM** Hiro Miyakawa
Endpoints and methods (API reference tutorial)
https://idratherbewriting.com/learnapidoc/docapis_resource_endpoints.html
- HM** Hiro Miyakawa
Optimized communications protocol for low earth orbit cubesat
<https://smartech.gatech.edu/handle/1853/60362>

Quais são os desafios técnicos em torno das implementações práticas de microsserviços?

RE Ricardo Ebbers Carneiro Leão 03/09/2020
Microsserviços só fazem sentido se forem escaláveis e resilientes. Para atingir esse requisito o desenvolvedor assume que vendor lock-in não é um problema e usa de soluções criadas pelos cloud providers, ou configura uma solução open source como Kubernetes, que é um mundo de complexidade por si só.

RE Ricardo Ebbers Carneiro Leão 03/09/2020
Outro ponto técnico de alto impacto é que, já que os microsserviços se comunicam via infraestrutura de rede, latência rapidamente se torna um problema. O desenvolvedor que pouco precisava se preocupar com a comunicação entre o monolito e os clientes agora tem um problema de primeira classe a resolver: fazer com que os microsserviços consigam se encontrar facilmente (service discovery) e se comunicar rapidamente (usando mensageria, REST, gRPC, etc)

AF Arthur Frade de Araújo 03/09/2020
Existem inúmeros desafios em manter uma arquitetura de microsserviços: (1) Gerenciamento de múltiplos deploys. Nem toda empresa possui a capacidade técnica para gerenciar múltiplas aplicações. Esse tipo de arquitetura muitas vezes exige outros conhecimentos para possibilitar a orquestração eficiente dos componentes (Docker, Jenkins, Kubernets).

AF Arthur Frade de Araújo 03/09/2020
(2) Health tracking: Se torna bastante complexo monitorar a saúde dos serviços ou encontrar a causa raiz de bugs em uma arquitetura de microsserviços. Hoje as empresas adotam ferramenta de monitoramento, as quais centralizam dados de estado de cada serviço. E isso mais uma vez exige um conhecimento específico e outros profissionais além dos devs/ devops.

AF Arthur Frade de Araújo 03/09/2020
(3) Resiliência. Não adianta fragmentar a arquitetura se os serviços continuam acoplados. É necessário garantir que um serviço possa continuar funcionando independentemente de outros. Para isso se adotam estratégias de mensageria e filas, para garantir que nenhuma comunicação se perca se uma das partes estiver inativa. E isso é mais uma tecnologia que a equipe vai precisar dominar.

RM Rafael Mota Alves 03/09/2020
Os principais desafios técnicos quando se trata de microserviços são: Manter os dados consistentes entre múltiplos serviços que geralmente têm bases de dados diferentes, Gerenciar os múltiplos serviços, dentro de gerenciar temos mais várias coisas como implantar os serviços, monitorar, se recuperar de falhas entre outras coisas, esse processo é complicado pela quantidade de serviços numa arquitetura de microserviços, Comunicação entre os serviços, em que devemos ter formas de os serviços

RM Rafael Mota Alves 03/09/2020
"descobrirem" uns aos outros, além de se comportar corretamente caso algum dos serviços envolvidos na comunicação sofra uma falha.

LB Lucas Barros 05/09/2020
Eu diria que um dos maiores desafios é relacionada a parte operacional, pois pra cada serviço cria-se um novo ambiente onde deve-se configurar as variáveis de inicialização, ajustar o processo de deployment, monitoramento, escalabilidade e desenvolvimento. O gerenciamento de vários serviços acaba por tornar-se mais complicado.

EV Emerson Victor 05/09/2020
Alguns desafios são: identificar e gerenciar dependências de cada um dos serviços, realização de testes end-to-end, automatizar a implantação de diversos serviços e conectividade entre os serviços.

HC Heitor Carvalho 06/09/2020
Uma das grandes vantagens da arquitetura é sua robustez. E assegurar que o sistema é robusto e tolerante a falhas pode ser um grande desafio. Além disso, desenvolvedores devem estar a par da cultura de microsserviços e preparados para os desafios técnicos que surgirão, principalmente em devops. Testes de integração tornam-se mais difíceis, assim como o monitoramento da aplicação como um todo, que agora encontra-se fragmentada.

Além dos desafios citados pelos colegas acima, em uma visão voltada para a segurança de uma arquitetura de microserviços, um desafio também é o monitoramento destes serviços, visto que há uma segmentação maior dos componentes do sistema, o que exige uma arquitetura específica para a coleta de logs, por exemplo.

DL**Danilo Lira** 06/09/2020

Um grande desafio da implementação é o espalhamento das informações de Log. Mesmo que sejam várias aplicações diferentes, temos que lembrar que elas compõem um único serviço e o espalhamento das informações pode dificultar a leitura dos registros. Além disso cada aplicação tem um ciclo de vida diferente, sendo assim, a cultura DevOps se torna essencial e isso exige uma equipe com nível técnico mais elevado.

LC**Lucas Cardoso** 06/09/2020

Acredito que a maior complexidade de microserviços vem da necessidade de orquestrar instâncias e a comunicação dos serviços (coisas como balanço de carga e a comunicação entre serviços). Ferramentas, como kubernetes e linkerd, tem sido desenvolvidas que ajudam a lidar com esses desafios.

RP**Ramom Pereira dos Santos Silva** 06/09/2020

Construir aplicações com Microserviços traz diversas vantagens, porém desafios também, ter uma aplicação executando de forma distribuída, podendo rodar em zonas de distribuição diferentes, ou até data centres diferentes, pode se tornar um grande gargalo, se não forem bem projetados. Outro desafio, seria a comunicação entre esses microserviços, síncrona ou assíncrona, quais ferramentas e tecnologias utilizar para esse fim.

RP Ramom Pereira dos Santos Silva 06/09/2020

Além disso, o monitoramento desses microserviços também deve ser levado em consideração, pois ao invés de lidar com um serviço (no caso dos Monolíticos), agora são diversos serviços, e estratégias de Service Mesh, como Istio e Linkerd, são bem importantes, além de Prometheus e Grafana.

RP Ramom Pereira dos Santos Silva 06/09/2020

Por fim, temos os critérios de definição desses microserviços, qual escopo de negócio que cada um irá lidar, como limitar bem suas responsabilidades, são alguns desafios que precisam ser bem definidos para enviar frustrações futuras.

TM**Talyta Maria Rosas Pacheco** 06/09/2020

Acredito que os desafios estão principalmente ligados ao time de desenvolvimento, pois se um time não tem conhecimento em comunicação e arquitetura de software, ou ainda não segue boas práticas de código, como padrão e testes, dificilmente podem fazer funcionar essa arquitetura. Além disso, por se tratar de uma arquitetura fundamentada em comunicação entre serviços, o processo de monitoramento se torna mais complexo, como na identificação de qual serviço deu erro e ...

TM Talyta Maria Rosas Pacheco 06/09/2020

evitar que outros serviços relacionados também apresentem problemas.

GC**Gabriel Cavalcanti de Melo** 07/09/2020

Alguns desafios ao optar por uma arquitetura de microserviços são: testar um fluxo que apresenta uma maior quantidade de microserviços, pois como esse serviços estão interligados, pode afetar a independência dessas funcionalidades; o monitoramento e logs, pois eles estarão espalhados; e infraestrutura, pois muitas vezes é necessário instalar várias aplicações na máquina do desenvolvedor.

CP**Claudio Pacheco** 08/09/2020

Além dos desafios técnicos citados até agora (complexidade para monitorar; dificuldade maior para testar devido às diversas integrações e dependências; consistência de dados; latência na comunicação; recuperação de falhas, entre outros), também podemos adicionar à lista 1) o suporte e manutenção, visto que cada serviço pode ter linguagem/framework/ciclo de vida diferentes, e 2) os desafios de manter segura a aplicação através das múltiplas interações necessárias.

Gabriel Pessoa 08/09/2020

Um dos maiores desafios é o monitoramento, algo que já é importante e fica ainda mais complexo quando se tem diversos serviços diferentes com métricas diferentes feitas por equipes diferentes. Onde ainda mais um扰动 numa métrica de um serviço pode ser resultado de um problema em outro serviço. Também é importante que os serviços sejam robustos a falha dos outros, para que um problema em um (ou apenas uma mudança de funcionamento) não derrube todos os serviços em cascata.

MG**Marcos Galvão** 08/09/2020

Assim como Lucas citou, creio que um dos grandes pontos de dificuldade está atrelado ao ambiente dos microserviços pois cada um deles tem suas próprias peculiaridades, dependências, tecnologias e eventualmente falhas. Manter tais aplicações, em larga escala, se torna uma atividade desafiadora, cada um deles ainda precisa de atenção individual, tal como já citaram, em seu monitoramento, proteção contra falhas e questões de segurança.

C

RJ**Renato Joaquim de Miranda Ferreira** 08/09/2020

A complexidade do sistema a ser desenvolvido, saber delimitar bem seu escopo e onde de fato o trabalho se torna mais eficiente/viável com o uso de microsserviços para a execução de suas tarefas, os desafios técnicos em torno de sua implementação, como a comunicação de dados/sistemas e como fazer toda essa estrutura funcionar de forma correta.

RO**Roberto Oliveira** 08/09/2020

Além dos desafios citados acima, existe também a chance de mais ataques, devido a sua maior comunicação e quantidade de interações.

Luca 08/09/2020

Além dos pontos levantados pelos meus colegas, temos que alguns problemas "secundários" em relação à microsserviços seriam: o uso e manutenção de padrões e convenções da plataforma / produto pelos diferentes serviços, visto a codebase "fragmentada", ou a dificuldade extra gerado normalmente para equipes de dados que trabalham gerando insights por meio dos logs e operações executadas em microsserviços, dado que esses podem estar espelhando em seus logs um bias da região onde está instanciado.

JS**José Sheldon Brito Fekete** 09/09/2020

O desenvolvimento de um sistema que utilize microsserviços é mais complexo do que sistemas monolíticos, exigindo mais conhecimento da equipe de desenvolvedores, tais como gerenciamento do log, monitorando, segurança, conversão de protocolos, além de necessitar de um maior planejamento no desenvolvendo do projeto, seguindo o API-First.

HM**Hiro Miyakawa** 09/09/2020

Exatamente por serem sistemas "independentes", é mais complexo, e precisa tomar cuidado na comunicação entre eles que os tornam "dependentes".

João Vasconcelos 09/09/2020

1.Existe uma complexidade para manter o banco de dados consistente pois podemos ter vários serviços usando a mesma base 2. Monitoramento se torna cada vez mais complexo com o aumento do número de serviços 3. Testar features que usam mais de um serviço também se torna complicado, demandando uma boa integração de todos os serviços 4 . Seguir e manter uma cultura devops também é retratado como uma dificuldade a ser superada

João Vasconcelos 09/09/2020

<https://dzone.com/articles/challenges-in-implementing-microservices>

Gabriel Ramos 09/09/2020

Quando se coloca em prática existem vários problemas a serem enfrentados, alguns mais comuns, como, gerenciar/organizar múltiplos serviços, por que cada serviço tem/pode ter, por exemplo, seu próprio tempo de deploy, testes automatizados e manuais que será distinto de todos os demais. Manter essa relação é um grande desafio e complexidade, acredito eu

VS**victor sena de lima attar** 09/09/2020

Alguns dos pontos a serem avaliados na arquitetura de microsserviços é: - Balanceamento dos serviços para interoperabilidade; - Distribuição do escopo de operação dos serviços - Distribuição dos databases - Comunicação entre serviços - Geração massiva de testes automatizados para garantir a consistência entre serviços - Gestão de cache de vários serviços - Gestão de orquestração de deploy automatizado unico ou diverso entre serviços Entre outros

IF**Igor Fernandes** 09/09/2020

- Ter um plataforma que consiga gerenciar quantidade de instâncias para cada serviço da plataforma baseado na demanda. - Aumento da complexidade da arquitetura. São vários serviços com base de códigos diferentes para gerenciar.

IN**Izabella Nascimento** 09/09/2020

A arquitetura de microsserviços exige muito mais conhecimento do que projetar uma aplicação monolítica tradicional, um fator importante para o sucesso vai depender da capacidade técnica da equipe e das tecnologias a serem utilizadas. Esse fator humano nos desenvolvimentos muitas vezes é deixado de lado ou não é dado a devida importância.

GD**Gabriel D'Luca** 09/09/2020

Quando a arquitetura escala e atinge um numero maior de microsserviços, fica mto difícil de gerenciar as codebases diferentes. As vezes também temos times diferentes, então pode ser que uma app tenha um time responsável por ela, e esteja consumido um serviço que seja um outro time que dê manutenção e suporte (e precisa ter uma comunicação efetiva entre esses times). Em qualidade, temos q garantir também o funcionamento da integração com esse serviço na pirâmide de testes (testes de integração) +

GD **Gabriel D'Luca** 09/09/2020

e esses testes assíncronos também trazem uma complexidade maior de testabilidade, identificação de cenários de teste, enfim.

 **Rafael Mota Alves** 04/09/2020

A ideia de Contratos Dirigidos ao Consumidor é evitar problemas na evolução de serviços que tem vários consumidores, pois num serviço que tem vários consumidores, mudanças devem ser pensadas de forma que não impactem ou impactem o mínimo de consumidores. Para contornar esse problema com o CDC, o serviço produtor coleta informações de que elementos da sua interface são usados por cada consumidor e então usa essas informações para evoluir sua API de forma a reduzir os custos de cada mudança.

 **Rafael Mota Alves** 04/09/2020

Uma ideia é transformar esses contratos em um suite de testes, de forma que os desenvolvedores do serviço produtor podem executar a suite de testes para verificar se não estão quebrando a integração com um consumidor. Numa abordagem em que existem vários microserviços que comunicam entre si, e que são desenvolvidos por equipes diferentes, o CDC é uma boa forma de escalar e garantir que os vários contratos entre serviços não sejam quebrados.

 **Lucas Barros** 05/09/2020

Contratos dirigidos ao consumidor seguem algumas características para auxiliar na evolução de contratos de comunicação entre serviços. Esse tipo de padrão é fechado e completo em relação às funcionalidades exigidas pelos consumidores, singulares na expressão das funcionalidades (mas essas são derivadas da necessidade dos consumidores) e imutável a partir de um conjunto de consumidores, mas podendo mudar caso os consumidores precisem.

 **Lucas Barros** 05/09/2020

Pensando aqui, um exemplo de tecnologia que pode ser utilizada para descrever esse tipo de abordagem é o gRPC, que utiliza Protocol Buffers.

 **Lucas Barros** 05/09/2020

Fonte das definições: <https://martinfowler.com/articles/consumerDrivenContracts.html>

 **Heitor Carvalho** 06/09/2020

CDC Sugere que a especificação da comunicação entre serviços seja feita através de um contrato que evolui conjuntamente com a aplicação. Esse contrato é decidido pelo consumidor com base nas chamadas a recursos necessários do provedor e a resposta que ele espera de tais chamadas. Com isso é feito um contrato que "dita" o que deve ser enviado, e que será compartilhado com a equipe responsável por desenvolver o serviço provedor. Isso facilita muito os testes, pois há a garantia através do contrato

 **Danilo Lira** 06/09/2020

O CDC é um padrão para a evolução de serviços. Neste padrão os contratos são definições de formatos e funcionalidade que um serviço deve fornecer para um consumidor. Esse padrão ajuda na redução de impacto na manutenção e evolução de serviços, pois é possível testar o funcionamento do serviços através do contrato. No contexto de Microserviços o CDC é utilizado para garantir um padrão na comunicação entre os serviços, e normalmente é o consumidor que compartilha o contrato com o provider.

 **Ramom Pereira dos Santos Silva** 06/09/2020

CDC possibilita a construção de contratos entre fornecedores e consumidores, é uma evolução de Serviços, baseando-se na abordagem de Schemas, ou seja, contratos bem estabelecidos para comunicação entre serviços. No contexto de Microserviços, contribui bastante na manutenção dos serviços, além dos testes, já que podem ser construídos baseados nesses contratos, além disso, especifica como a comunicação deve ser entre fornecedor e consumidor.

 **Gabriel Cavalcanti de Melo** 07/09/2020

O princípio do CDC é evitar a execução de testes de integração entre todos os serviços, definindo uma regra (contrato) para definir se os dados que são passados do servidor para o consumidor estão no formato correto. Isso se encaixa com o estilo de microserviços, pois como garante que cada serviço seja testado e implantado independente dos demais, então segue o princípio da abordagem de microserviços de tornar as partes do sistema independentes.

 **Talyta Maria Rosas Pacheco** 07/09/2020

O CDC é uma solução para testes entre os serviços de uma aplicação, sem a necessidade de testar um fluxo complexo com vários serviços interligados (o que é muito custoso). Relacionam com microserviços pelo fato de assegurar mudanças em um serviço provedor sem prejudicar o funcionamento dos serviços consumidores, facilitando que os serviços consumidores se desenvolvam em relação às alterações.

 **Lucas Cardoso** 07/09/2020

O principal conceito de CDC lida com a evolução dos serviços, em particular em como se dão os contratos dos esquemas entre provedor e cliente. A ideia é que o provedor possa receber informações de como o cliente lida com a API de forma a minimizar mudanças quebrantes e assim poder melhor evoluir o serviço. Em microserviços, essa abordagem ajuda na expansão e manutenção de seus serviços, permitindo que consumidores e produtores evoluam, tomando como base o contrato definido

RE

Ricardo Ebbers Carneiro Leão 07/09/2020

Consumer Driven Contracts trazem dois grandes benefícios para serviços que precisam evoluir (como microsserviços): 1. CDC focam na especificação e entrega de funcionalidade ao redor de valores chave de negócio.. 2. A consequência de expor apenas contratos orientados ao consumidor é que o serviço provedor terá um conjunto de contratos enxutos, que só mudam (ou evoluem) a partir de uma demanda clara do consumidor, por sua vez oriunda de uma demanda de negócio com um claro valor agregado.

CP

Claudio Pacheco 08/09/2020

Contratos Dirigidos ao Consumidor estão relacionados à compatibilidade de comunicação entre dois serviços: o consumidor e o fornecedor. Como sugere o nome, é um acordo/padrão para a troca de dados (qual o formato dos dados, por exemplo). O CDC auxilia na testabilidade, pois permite focar em verificar se o contrato está sendo mantido, em vez de realizar inúmeros testes de integração entre os diversos serviços disponíveis (precisaria subi-los todos, por exemplo). +

Claudio Pacheco 08/09/2020

- + Como microsserviços definem-se por essas múltiplas interações, podemos aplicar os CDCs nesta arquitetura.

MG

Marcos Galvão 08/09/2020

Utilizando CDC toda a comunicação entre consumidor e provedor se dá através de contratos, que basicamente estabelecem como tal comunicação deve ocorrer, isso permite que tanto cliente quanto provedor sejam testados individualmente e torna possível testar o provedor simulando chamadas do cliente somente utilizando o contrato, isso viabiliza, assim como Gabriel e Talyta disseram, que os testes de integração não necessitem rodar fluxos complexos de comunicação.

Marcos Galvão 08/09/2020

Tal ideia encaixa com Microsserviços pois, além da independência para testagem que ele provê, é possível para o provedor saber de suas obrigações (o que o contrato espera dele) e a partir disso rastrear se suas mudanças ainda satisfazem tal pacto, e caso não, quais os clientes que serão afetados. Permitindo que a expansão de tais serviços ocorra da maneira mais condizente possível com o atual funcionamento.

RJ

Renato Joaquim de Miranda Ferreira 08/09/2020

O CDC lida, entre outras coisas, com serviços evolutivos de acordo com o desejo/necessidade do cliente. Esse tipo de contrato se encaixa muito bem com microsserviços uma vez que cada parte do sistema pode ser utilizada ou não de acordo com o cliente, podendo entregar os mais diferentes tipos de solução 'individualmente', dessa forma não é preciso testes no sistema como todo, gerando um custo menor e beneficiando as partes envolvidas.

AF

Arthur Frade de Araújo 08/09/2020

A ideia do Consumer Driven Contracts é garantir que os consumidores de um serviço não sejam impactados por mudanças nos modelos dos produtores. Um repositório de contratos pode ser acessado pelos desenvolvedores através de testes para saber, antes da implantação, se alterações em modelos podem quebrar a expectativa de algum outro serviço. Esse tipo de abordagem pode ser muito útil na gestão diária de uma arquitetura de microsserviços...

Arthur Frade de Araújo 08/09/2020

Apesar de ser relativamente fácil construir e fazer o deploy de um novo serviço, a manutenção e evolução dos mesmos é uma tarefa complexa devido às conexões que os serviços possuem uns com os outros. O CDC ajuda o dev a identificar onde uma mudança pode afetar a arquitetura, antes mesmo de fazer o deploy. Assim pode corrigir ou tomar outra rota para evitar quebra nos fluxos.

Lucca 08/09/2020

Incrementando o que Arthur pontuou: os Consumer Driven Contracts permitem as empresas que fornecem esses serviços possam fazer expansões e melhorias em suas aplicações, sem impedimentos ou altos custos para ajustes decorrentes do uso de contratos anteriores por seus clientes. Ao mesmo tempo essa abordagem permite uma maior especialização de contratos para clientes e casos específicos, dando à empresa um maior poder de customização de seus serviços para os clientes.

RO

Roberto Oliveira 09/09/2020

Ainda, além do que já foi dito, você pode executar seu conjunto de testes sempre que houver uma mudança na especificação de forma proativa e que como consumidor, suas necessidades podem ser conhecidas antes da formação do contrato.

JS

José Sheldon Brito Fekete 09/09/2020

Os testes usados no Consumer Drive Contracts pelo lado do proprietário do serviço muitas vezes podem ser falhos, tendo uma interpretação errada de como o cliente pode usar o serviço. A melhor alternativa seria a criação de um conjunto de testes integrados automatizados, assim o proprietário do serviço recebe os testes de integração de cada cliente e incorpora na sua rotina de teste, assim ficando livre para alterar o código do seu sistema desde que os testes dos clientes sejam aprovados.

C

HM

Hiro Miyakawa 09/09/2020

As melhorias de APIs e serviços podem acabar não atendendo clientes antigos, ou clientes novos, e o fato de não saber corretamente a necessidade dificulta a evolução service owner. O CDC ajuda pega as necessidades dos clientes e coloca dentro dos testes de rotina, e uma vez passando nos testes, o service owner podem continuar implementando. Então, isso mantém algo relevante para o cliente e ao mesmo tempo deixando a possibilidade de melhoria.

HM

Hiro Miyakawa 09/09/2020

"What the parties really need are a set of automated integration tests" segundo o texto "Consumer-Driven Contracts" by Ian Robinson. No contexto de microserviço, isso faz sentido pois eles são sistemas independentes que se comunicam, então enquanto estiver conseguindo comunicar, está tudo certo. A CDC encaixa bem na filosofia de microservices.

VS

victor sena de lima attar 09/09/2020

CDC é um padrão que define como será feita a evolução de serviços. Nele há existência de contratos onde são especificados os esquemas, as interfaces, políticas e conversações entre o consumidor e o provedor do serviço. Essa especificação é importante para que fique explícito para ambas as partes como deve ser consumido esse serviço e os padrões para possíveis evoluções durante o desenvolvimento de novas funcionalidades.

VS

victor sena de lima attar 09/09/2020

Durante uma arquitetura de microserviços, esses contratos são a base para que tudo ocorra de forma menos burocrática durante o desenvolvimento e o consumo.

IN

Izabella Nascimento 09/09/2020

Os contratos orientados ao consumidor (CDC) são uma especialização de testes de simulação, com a especialidade de que o contrato de interface é conduzido pelo consumidor e não pelo provedor. Geram várias vantagens, como obter ciclo de feedback mais rápido, já que é possível executar a qualquer momento, também é possível descobrir operações de uma API não são usadas, para descartar códigos desnecessários que atrapalham a base de código.

GD

Gabriel D'Luca 09/09/2020

Como muitos já comentaram, é justamente isso, um "acordo" definido entre o consumidor e o provedor. Nesse caso, o CDC se trata justamente da orientação desse contrato para aquele que efetivamente o consome. Adicionando, temos também testes de contrato que serem pra verificar se o provedor tá provendo o que o consumidor espera, e se o consumidor continua funcionando com a resposta a ser provida. É útil em casos onde o contrato é modificado, por exemplo, para garantir que nada quebrou, mas (+)

GD Gabriel D'Luca 09/09/2020

em geral, não é um teste tão fácil de implementar e as vezes nem muito efetivo na prática (vida real), dado que exige muita colaboração entre provedores e consumidores.

Geralmente quem mais precisa (o consumidor, por isso o contrato é orientado a ele) fica de mãos atadas pq o teste tende a estar do lado do provedor. Isso é bom pra trazer facilidade ao provedor para perceber que uma resposta quebrou o que o consumidor espera, mas traz um risco (ao consumidor) +

GD Gabriel D'Luca 09/09/2020

visto que o teste não está do lado dele e essa validação pode ser facilmente ignorada.

GD Gabriel D'Luca 09/09/2020

Ah, adicionando a relação com microserviços: esse contrato é justamente o contrato de comunicação definido entre um serviço (provedor) e uma aplicação/outro serviço (consumidor)

IF

Igor Fernandes 09/09/2020

Os princípios do CDC estabelece que uma aplicação deve ter contratos bem definidos quanto a comunicação. Uma das maiores vantagens de se utilizar essa abordagem é a facilidade quanto ao desenvolvimento de testes automatizados. O desenvolvedor pode simular um ambiente de comunicação sem a necessidade de um dos lados, permitindo assim validar todo protocolo definido.

Como a lei de Postel também pode ser relevante nesse cenário?

RM

Rafael Mota Alves 04/09/2020

A lei de Postel diz: Seja conservador com o que você faz, mas seja liberal com o que você aceita de outros. Esse princípio é interessante para comunicação entre microserviços pois se ambos o cliente e o servidor da comunicação seguirem esses princípios, problemas de integração de serviços. Por exemplo, no caso de uma comunicação HTTP, entre dois serviços, se o cliente seguir a lei de Postel, ele vai ser conservador em relação ao que ele envia na requisição e vai ser liberal em relação ao que ele

RM Rafael Mota Alves 04/09/2020

recebe como resposta e se o Servidor seguir, ele vai ser liberal em relação ao que recebe

na requisição do cliente e conservador em relação ao que vai responder, portanto se os dois seguirem a lei a risca será impossível ocorrer erros de integração. No entanto, em situações em que as interfaces entre os serviços mudam, desde que não sejam breaking changes os serviços continuarão funcionando normalmente, dessa forma, podemos por exemplo atualizar serviços de forma independente, sem se preocupar

 **Rafael Mota Alves** 04/09/2020

que por exemplo um campo a mais não quebre a integração.

 EV

Emerson Victor 05/09/2020

A lei de postel, nesse contexto, pode ser explicada como: programas que enviam mensagens devem sempre manter as especificações de quem recebe e programas que recebem devem aceitar entradas não conformantes, desde que sejam claras. Um caso de uso nesse contexto é a evolução de uma API remota, onde pode-se aplicar o padrão ao fazer uma alteração no conteúdo aceito ou retornado pela API em um determinado endpoint, ou introduzir um novo endpoint para distinguir entre as versões nova e antiga.

 HC

Heitor Carvalho 06/09/2020

A lei de Postel diz que deve-se ser liberal com o que é aceito, mas conservador com o que se envia. Aplicando esse princípio em microserviços temos uma visão bem definida sobre como deve ser a intercomunicação nesse tipo de arquitetura: flexível quanto a aceitar qualquer tipo de input, mas extremamente rigorosa em quais inputs devem ser enviados. Isso assegura uma comunicação mais flexível e ao mesmo tempo limpa e concisa.

 DL

Danilo Lira 06/09/2020

O princípio da robustez no contexto da evolução de serviços pode ser bastante útil. Se os serviços forem modificados seguindo esse princípio, então haverá uma preocupação em modificar o mínimo o possível o que é enviado por essas APIs. E se conjuntamente os serviços possuirem flexibilidade no recebimento de requisições, teremos uma maior consistência na comunicação entre APIs e redução na chance de falha. Pois teremos serviços abertos ao recebimento de informações e consistentes em seu retorno.

 RP

Ramom Pereira dos Santos Silva 06/09/2020

A Lei de Postel diz o seguinte: "Ser conservador no que você faz, ser liberal no que você aceitar de outros". Nesse caso, para microserviços que são fornecedores devem seguir estritamente os contratos de comunicação estabelecidos, já os consumidores podem receber mais informações, além das necessárias para sua utilização. No contexto de Microserviços, isso é muito útil em sua evolução, já que podem serem modificados sem perder compatibilidade com seus consumidores.

 RP **Ramom Pereira dos Santos Silva** 06/09/2020

Ao modificar o fornecimento de suas informações, fornecedores podem incluir mais dados, desde que siga os contratos estabelecidos, garantindo a extensibilidade da API. Com essa estratégia, os consumidores serão modificados minimamente, já que os contratos continuam respeitados e foram projetados para serem liberais no recebimento das informações dos fornecedores.

 LC

Lucas Cardoso 07/09/2020

A lei de Postel é interessante para comunicação em microserviços pois ela tende a diminuir a complexidade das comunicações. Se todos os clientes e serviços seguem essa lei, a tendência é que todos sejam flexíveis nas requisições que recebem, permitindo uma maior capacidade de evolução dos serviços.

 RE

Ricardo Ebbers Carneiro Leão 07/09/2020

O princípio de robustez de Postel é muito útil ao criar serviços que são flexíveis ao receber requisições e, quando possível, são capazes de interpretar a intenção correta do consumidor mesmo com um payload diferente do que inicialmente mapeado. Um exemplo prático é expor um endpoint único para duas versões ou casos de uso e decidir internamente qual versão usar com base no alguma feature flag no payload passado. Dessa forma é possível evoluir por meio de reutilização sem quebrar contratos.

 TM

Talyta Maria Rosas Pacheco 07/09/2020

Esse padrão de robustez apresentado pela lei de Postel é relevante pelo fato de manter compatibilidade quando consumidores recebem entradas diferentes do padrão, evitando que eles quebrem.

 GC

Gabriel Cavalcanti de Melo 07/09/2020

A lei de postel se encaixa nesse cenário pois ela diz que que os dados que são passados de um serviço para outro devem estar sempre padronizados, mas deve-se aceitar entradas de dados em diferentes formatos. E por isso, assim como os contratos dirigidos ao consumidor, assegura que os serviços sejam testados e implementados de forma separada, mesmo que haja comunicação entre eles, garantindo a independência e escalabilidade

 LB

Lucas Barros 07/09/2020

A lei de postel é interessante nesse aspecto porque consegue melhorar a interoperabilidade entre serviços, já que os serviços, ao receberem uma chamada à sua interface, tentam corrigir possíveis erros de uso de sua interface e manter a interação válida. De certa visão, pode-se

dizer até que é uma forma consumer-driven de fazer as coisas, já que tenta ajudar o consumidor a realizar a operação de uma forma liberal.

CP

Claudio Pacheco 08/09/2020

A lei está diretamente ligada à robustez na comunicação entre diferentes serviços. Formulada como um guia para redes e protocolo TCP/IP, ela é aplicada em outras áreas, inclusive arquitetura microserviços. Como dito nos comentários anteriores, a lei tem basicamente 2 conceitos: ser conservador no envio (fornecedor deve enviar dados conforme estabelecido

[pelo contrato, por exemplo]), mas liberal no recebimento (não aceitar apenas um formato: "semantics over syntax"). Ex: um campo pode aceitar +

CP **Claudio Pacheco** 08/09/2020

+ "0", "null", "false" ou campo vazio para representar ausência de um dado. O fornecedor deverá enviar como "null", supondo que seja esse o valor especificado. Mas o serviço poderia receber os valores anteriores como forma de manter robustez. Em microserviços, a aplicação dessa lei evita que a comunicação seja muito literal e quebre sempre que o valor não corresponder ao padrão estabelecido, principalmente com atualizações e evolução constantes. Link interessante: Postel's Law

MG

Marcos Galvão 08/09/2020

A Lei de Postel tem muito a ver com a ideia de robustez dos microserviços, a lei basicamente diz para: Enviar dados conforme já está definido no contrato e, ser apto a receber respostas que possuam desvios do que já está estabelecido, desde que a mesma ainda seja clara. Isso permite que os Microserviços sejam mais flexíveis para manipular o que recebem e por conseguinte futuras alterações nas mensagens, caso não sejam abruptas demais, ainda permitem o funcionamento da aplicação.

RJ

Renato Joaquim de Miranda Ferreira 08/09/2020

A lei de postel diz respeito a robustez dos microserviços implementados, deve se enviar o que foi definido previamente e que esteja funcionando corretamente mas também é necessário aceitar críticas/evolutivas para o bom funcionamento do sistema. Dessa forma, é necessário que o sistema seja robusto o suficiente para aguentar a demanda descrita mas também deverá ter evolutivas flexíveis no seu escopo.

AF

Arthur Fraude Araújo 08/09/2020

A lei de Postal se refere a capacidade que uma aplicação possui de tolerar um input que não venha formatado de maneira exata a esperada, mas que possa ser compreendido e processado mesmo assim. O CDC tem tudo a ver com essa lei, pois a ideia é que a evolução dos contratos não interfira no funcionamento das aplicações que os consomem. Por exemplo: A adição de novas informações em um contrato não deveria quebrar um serviço que utiliza apenas uma porção dos atributos do contrato...

RO

Roberto Oliveira 09/09/2020

O principal objetivo é fazer com que diferentes implementações interoperem. é sobre ser conservador no que envia e liberal no que recebe, ou seja, aceitar peculiaridades de inofensivas de outros.

JS

José Sheldon Brito Fekete 09/09/2020

A lei de postel ou o princípio da robustez tem a ideia de enviar dados conforme foi pré estabelecido no contrato e ser mais tolerante com os dados recebidos desde que tenha um entendimento claro dos mesmos. Se assemelhando muito como um sistema de microserviços deve funcionar.

HM

Hiro Miyakawa 09/09/2020

Postels' Principles diz sobre robustez "be liberal in what you receive, and be conservative in what you send". Já que microserviços é algo baseado em comunicação das partes, isso ajuda a manter os serviços rodando, sendo mais maleável com input alheios, contanto que consiga entender, claro (Achei interessante o Do What I Mean e não Do What I Said que Christopher fala no vídeo)

VS

victor sena de lima attar 09/09/2020

A lei de postel intitula que deve ser conservador no que você faz e ser liberal no que se aceita dos outros, ou seja, quando se provê um serviço, deve-se ditar as regras para implementação, no outro lado, quem consome o serviço deve ser flexível quando for consumir o serviço, seguindo as regras estabelecidas no contrato de consumo.

VS **victor sena de lima attar** 09/09/2020

Fazendo a ponte com as regras de atualização de serviços, parte do consumidor usar apenas o que for importante para ele, caso seja adicionado algo, não precisa ser usado, caso precise, possa adicionar, ou seja, sendo liberal no consumo.

IN

Izabella Nascimento 09/09/2020

A lei de Postel fala "Seja conservador no que você enviar, ser liberal no que você aceitar", ou seja, os programas que enviam mensagens para outras máquinas deve obedecer completamente às especificações pedidas, entretanto para os programas que recebem as mensagens devem aceitar a entrada não conforme, desde que haja um entendimento claro da

mensagem passada. Em microserviços se aplica diretamente na forma da comunicação entre os serviços, que deve ser aceito mesmo se não estiver totalmente...

 Izabella Nascimento 09/09/2020

de acordo com as especificações mas puder ser entendido.

 IF

Igor Fernandes 09/09/2020

A Lei de Postel fala sobre flexibilidade, uma comunicação pode ser forgivable para entender o que cliente está requisitando, mas deve estabelecer um "contrato" de como ele vai responder as requisições. A ideia é que um serviço seja fácil de se contactar (entrada) e previsível quanto a saída.

Quais são as principais soluções (padrões, estilos arquitetônicos, ferramentas etc.) a serem implementadas para: (a) Serviços orientados a mensagens; (b) Endpoints REST; (c) Protocolos de comunicação otimizados; (d) Documentação da API.

 AF

Arthur Frade de Araújo 03/09/2020

(a) Existem muitas soluções utilizadas no mercado. Uma das mais famosas é o Apache Kafka, plataforma open source para stream de eventos. É uma espécie de message broker que suporta filas e mensagem no estilo publish/ subscribe. As plataformas de cloud mais famosas também possuem seus próprios brokers como o AWS SNS, Azure Service Bus, entre outros.

 AF Arthur Frade de Araújo 07/09/2020

Endpoints REST (b) Um dos padrões que mais caracterizam uma aplicação RESTful é o não armazenamento de estado. Endpoints RESTful são "stateless" ou seja, eles não armazenam dados de outras requisições já efetuadas. Toda chamada será interpretada individualmente, por exemplo: toda chamada a um endpoints protegido passará pelo processo de autorização (authorization) independentemente de quantas vezes o recurso seja chamado. Além disso existem padrões de nomenclatura geralmente o URI...

 AF Arthur Frade de Araújo 07/09/2020

(universal resource identifier). O padrão REST também aproveita a semântica dos métodos HTTP para a utilização de um mesmo URI. Ou seja, um mesmo recurso pode ser manipulado através de um GET, POST, PUT ou DELETE.

 AF Arthur Frade de Araújo 07/09/2020

Protocolos Otimizados (c) Com o intuito de tornar a comunicação entre serviços mais eficiente e reduzir serializações onerosas ("marshalling" e "unmarshalling") de dados, surgem novas implementações de protocolos de comunicação. Um deles é o gRPC, protocolo desenvolvido pelo Google, que trabalha com o conceito de protocol buffers (protobuf) baseado no RPC (Remote procedure call).

 AF Arthur Frade de Araújo 07/09/2020

Documentação de APIs (d) Hoje em dia existe um padrão famoso denominado OpenAPI, o qual estabelece uma sintaxe de especificação que ao se integrar com ferramentas de documentação que seguem o padrão, promovem funcionalidades interessantes como: criação de clientes (geração de código) e documentos (html).

 RM

Rafael Mota Alves 04/09/2020

Para **(a) Serviços orientados a mensagens,** temos padrões como Publish/Subscribe, Filas, Sagas e Event Sourcing, alguma ferramentas que podem ser usadas para isso são SNS, SQS, rabbitMQ.

 RM

Rafael Mota Alves 04/09/2020

Para (b) Endpoints REST, temos ferramentas como: o API Gateway da AWS que traz ferramentas para implementação dessas APIs, também temos frameworks específicos para REST, como o django rest framework.

 RM

Rafael Mota Alves 04/09/2020

Para (c) Protocolos de comunicação otimizados, temos como exemplo o gRPC, que além de trazer um formato binário, mais performático, permite interoperabilidade entre várias linguagens implementando uma forma de Schema, que pode ser usada para derivar implementações para várias linguagens, além de funcionar como uma ferramenta de RPC

 RM

Rafael Mota Alves 04/09/2020

Para (d) Documentação de API, temos a ferramenta Swagger, que possibilita a documentação de APIs usando YAML, podendo renderizar para uma forma mais amigável para visualização, além de ser possível gerar código que implementa a estrutura básica da API em várias linguagens em frameworks

 LB

Lucas Barros 06/09/2020

(c) Protocolos de comunicação otimizados: um exemplo é GraphQL, uma forma de comunicação em que os dados são requisitados de acordo com a demanda, evitando que os payloads de resposta sejam retornados com valores que não serão utilizados pelo cliente. Esse tipo de protocolo provê apenas um endpoint e possui um protocolo de tipos para que os clientes sempre peçam apenas dados válidos.

LB

Lucas Barros 06/09/2020

(b) Endpoints REST: frameworks web como Ruby on Rails auxiliam na criação de endpoints REST.

HC

Heitor Carvalho 06/09/2020

(a) Para serviços orientados a mensagens, uma solução mais simples seria adotar o servidor faye que oferece uma arquitetura de publish/subscribe (onde existe um canal e tudo que é enviado para aquele canal é repassado aos usuários inscritos), ou em caso de uma abordagem orientada a mensageria para processamento de ações/logs em fila, temos o rabbitMQ ou o kafka.

HC

Heitor Carvalho 06/09/2020

(b) Para endpoints REST a maioria dos frameworks atuais dá suporte para que esse padrão seja seguido, então pode ficar a cargo do desenvolvedor segui-lo utilizando as ferramentas da linguagem. Existem serviços de terceiros para gerenciar API gateways e bibliotecas que auxiliam na documentação de APIs como o Swagger.

RP

Ramom Pereira dos Santos Silva 06/09/2020

(a) Para mensageria, muito útil na comunicação assíncrona entre Microserviços, as ferramentas de grande destaque são Apache Kafka e RabbitMQ, onde oferecem uma arquitetura robusta de pub/sub, basicamente Microserviços podem publicar mensagens em canais, enquanto outros podem ler essas mensagens.

RP

Ramom Pereira dos Santos Silva 06/09/2020

(b) Para endpoints REST, diversas linguagens de programação possuem frameworks que foram construídos para tal propósito, com padrões e documentações bem estabelecidas. A utilização desses frameworks é bem particular de cada empresa, mas alguns de destaque são: Ruby on Rails, Django para Python, Express para Nodejs e Spring para Java.

RP

Ramom Pereira dos Santos Silva 06/09/2020

(c) Alguns protocolos bastante usados atualmente são: gRPC e GraphQL. gRPC é um protocolo, desenvolvido pelo Google, agnóstico de Linguagem de Programação e que utiliza buffers de protocolo, e tem como vantagem o uso reduzido da rede. Já o GraphQL, criado pelo Facebook, visa a diminuição da quantidade de dados nas requisições, onde os clientes só recebem aquilo que irão utilizar.

RP

Ramom Pereira dos Santos Silva 06/09/2020

(d) Swagger é uma ferramenta de grande destaque para documentação de API, principalmente REST, dado sua facilidade em construir arquivos YAML, incluindo também formas automáticas de gerar documentação, além de contribuir em testes automáticos de API.

HC

Heitor Carvalho 06/09/2020

(c) Em caso os protocolos HTTP/REST já não sejam suficientes, existem protocolos otimizados para microserviços como o AMQP, que cuida do enfileiramento de mensagens passadas entre os serviços, ou o gRPC, que trabalha para otimizar as chamadas remotas a métodos utilizando o protocolo HTTP/2 para transporte.

HC

Heitor Carvalho 06/09/2020

(d) Para documentação da API, existem vários padrões que podem ser seguidos, inclusive alguns sendo internos da própria organização. Algumas ferramentas/padronizações existentes são o JSON:API ou o Swagger, por exemplo. Em geral, boas documentações devem conter uma descrição clara do que o método faz, exemplos de chamadas e conteúdo da resposta (Se possível implementadas em vários cenários linguagens - curl, ruby, js, etc...), e se possível, deve oferecer uma experiência interativa para testes.

DL

Danilo Lira 06/09/2020

O que achei de diferente dos colegas foi o OMQ que é uma biblioteca de alta-performance para mensagens assíncronas.

LC

Lucas Cardoso 07/09/2020

(a) além do rabbitMQ e kafka já citados, a maioria dos provedores de cloud possuem algum tipo de serviço desse tipo, como o AWS SNS, IBM MQ, Google Cloud Pub/Sub, Aliware MQ

LC

Lucas Cardoso 07/09/2020

(d) Existem diversas ferramentas para documentação, como Swagger, ClickHelp, DapperDox, Readme.io, API Doc, e GitBook. Alguns desses exemplos podem ser usados para mais tipos de documentações (não só APIs). O importante, como Heitor explicou, é conter exemplos, descrições e ser o mais claro possível, além de, obviamente, estar correta. Exemplos próximos dos casos de uso também são interessantes

RE

Ricardo Ebbers Carneiro Leão 07/09/2020

a) Message Queues e Message Brokers, vários já foram citados pelos colegas; b) Quando preciso criar um server REST rapidamente vou direto no Spring Initializr (<https://start.spring.io/>), onde é possível criar um Web MVC a nível de produção com o mínimo de esforço c) o Remote

 **Ricardo Ebbes Carneiro Leão** 07/09/2020

- d) Gosto muito do padrão do Swagger para documentação de APIs REST (que também permite gerar clients e servers razoavelmente bem), mais recentemente tive contato com o API Blueprint (<https://apiblueprint.org/>) que tem uma sintaxe um pouco mais simples que o Swagger e dá as mesmas capacidades.

 **Renato Joaquim de Miranda Ferreira** 08/09/2020

- a) Conforme citado anteriormente pelos colegas, o RabbitMQ fornece um excelente serviço de mensagem. b) Ruby on Rails atendem muito bem as demandas de Endpoints REST c) GraphQL, no qual a requisição é feita apenas no que é suportado/usado de fato. d) OpenAPI, fornecendo padronizações muito úteis a primeiros entrantes em contato com a tecnologia além de desenvolvedores mais experientes.

 **Gabriel Cavalcanti de Melo** 09/09/2020

Outra linguagem para documentação, modelagem e descrição de APIs que encontrei, é RAML. Ela é baseada em YAML, dá suporte a todo o ciclo de vida da API, divide o sistema e explica o comportamento de cada parte.

 **Talyta Maria Rosas Pacheco** 09/09/2020

- a) Message-Oriented Middleware (MOM) é uma infraestrutura de software ajuda no envio e recebimento de mensagens entre sistemas interligados. Ex de produto é o Amazon SNS b) Google Cloud Endpoints d) ReDoc

 **Claudio Pacheco** 09/09/2020

a) Pelo o que pude entender, um padrão conhecido para serviços orientados a mensagens é o publish-subscribe. O publisher envia mensagens de forma categorizada, mas sem remetê-las diretamente a um receptor. O subscriber, interessado em determinadas categorias, busca essas mensagens sem se preocupar com o remetente em si.

 **Claudio Pacheco** 09/09/2020

- d) Encontrei essa lista recente com cinco ferramentas para documentação de API, apesar de nunca ter tido contato com elas anteriormente: Swagger UI, SwaggerHub, ReDoc, DapperDox, OpenAPI Generator (<https://blog.dreamfactory.com/5-best-api-documentation-tools/>)

 **Roberto Oliveira** 09/09/2020

Além de todos mencionados, existe também o modelo POINT TO POINT, Esse modelo garante que uma mensagem seja entregue a um único destinatário. Portanto, mesmo que a fila tenha mais de um consumidor ativo, apenas um receberá cada mensagem.

 **Hiro Miyakawa** 09/09/2020

- a) Streaming Text Oriented Messaging Protocol que é para trabalhar com um MOM que os colegas citaram. b) Encontrei esse exemplo do instagram. (Endpoints and methods API reference tutorial) c) Optimized communications protocol for low earth orbit cubesat d) swagger io

 **Luan Brito** 09/09/2020

Outra ferramenta muito interessante é o Logstash, muito usado no contexto de visualização de dados em conjunto com a stack ELK. O logstash funciona como uma fila que recebe todos os tipos de dados e envia para um outro ponto, mas com um diferencial que aplica diversas transformações a esses dados.

 **Marcos Galvão** 09/09/2020

- a: Eu entendi algo similar ao Claudio, para tal funcionalidades conheço o Amazon SNS que em sua versão gratuita permite enviar até 1000 email e 1 milhão de push notifications;

 **Marcos Galvão** 09/09/2020

- b: Como já citaram, para criação de endpoints REST praticamente todas as linguagens possuem frameworks, tais como: Ruby On Rails, Django, Flask, etc

 **Marcos Galvão** 09/09/2020

- d: Para tal funcionalidade além das ferramentas já citadas existe o Postman, que permite documentar os endpoints, e de produzir uma mock API para testes.

 **José Sheldon Brito Fekete** 09/09/2020

- a) Message Oriented Middleware (MOM), que dá suporte ao envio e recebimento de mensagens em sistemas distribuídos

 **José Sheldon Brito Fekete** 09/09/2020

- b) PostMan pode ser utilizado para automação de testes das APIs, possui um cliente REST de fácil utilização.

 **José Sheldon Brito Fekete** 09/09/2020

- d) Swagger é uma linguagem de descrição de interface usado para documentar APIs RESTful expressas usando JSON.

**Igor Fernandes** 09/09/2020

a) Amazon SNS, IFTTT b) MockAPI: <https://www.mockapi.io/> c) SOAP o próprio REST d) Swagger e Postman

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

Nenhum comentário realizado

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

🔗 Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

**Gabriel D'Luca** 15/09/2020

Onde devemos colocar os links para os artigos do Estudo de Caso?

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa



🔗 Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Lehman para MBA's

Tópicos [de aprendizagem]

As leis de evolução de software são mais conhecidas por Leis de Lehman e são dividida em oito leis, onde as três primeiras leis foram apresentadas em 1974, a quarta, quinta e sexta lei foram apresentadas em 1980 e as duas últimas leis em 1996

Referências

The Past, Present, and Future of Software Evolution
<https://plg.uwaterloo.ca/~migod/papers/2008/icsm08-fosm.pdf>

On the Evolution of Lehman's Laws
<https://plg.uwaterloo.ca/~migod/papers/2013/lehmanPaper.pdf>

Lehman's laws of software evolution (Wikipedia)
<https://bit.ly/3bkPrB>

As 8 leis de Lehman foram o Manifesto do século XX
<https://bit.ly/2ExyKk0>

Links compartilhados

 Arthur Frade de Araújo

As 8 leis de Lehman foram o Manifesto do século XX
<https://www.baguete.com.br/categorias/jorge-horacio-audy/17/04/2014/as-8-leis-de-lehman-foram-o-manifesto-do-seculo-xx>

A Lei da Modificação Contínua (1974) versa que os softwares devem ser continuamente adaptados para não deixar de atender às necessidades para as quais foram criados. Já a Lei da Complexidade Crescente (1974) diz que à medida que o software sofre alterações, sua estrutura original passa a ser descaracterizada e sua complexidade aumenta, de modo que também aumenta gradativamente os custos de sua manutenção até o momento que a manutenção passe a se tornar inviável. Por fim, a Lei da Auto-Regulação (1974) por sua vez sugere que o software possui uma dinâmica própria definida, o que decide as tendências da manutenção, limitando o número de possíveis mudanças. Como lidar com estas 3 leis em um ciclo de vida de uma Microservice Based Architecture (ou Application)?

 Luan Brito 09/09/2020

1ª Lei - Acredito que seu impacto numa Microservice Based Architecture seja ainda maior do que numa arquitetura monolítica, tendo em vista que a variedade de tecnologias que podem ser usadas por toda a arquitetura é muito maior, fazendo com que a manutenção tenha que se preocupar com diversos contextos diferentes.

 Luan Brito 09/09/2020

2ª Lei - Ao meu ver, essa lei pode ter um impacto reduzido porque atinge principalmente os projetos no quesito complexidade, e numa arquitetura de microserviços tem um grande papel de reduzir as responsabilidades/complexidade das partes. Acho que uma boa forma de lidar é utilizar técnicas de contrato para os provedores de serviço, e se ater a suas responsabilidades. Fazendo com que seja menos difícil manter sua estrutura original.

 Luan Brito 09/09/2020

3ª Lei - Acho que tem duas perspectivas interessantes a serem vistas aqui: uma arquitetura de microserviços possui sua dinâmica muito bem definida, e portanto a lei continua tendo um grande impacto. Por outro lado, se for olhar para as partes em específico, se torna MUITO mais viável fazer diversas mudanças (ou até mesmo substituição de algum serviço), desde que seu papel na arquitetura continue sendo devidamente cumprido

 Rafael Mota Alves 10/09/2020

As 3 leis falam sobre a frequência que softwares devem ser modificados e a consequência dessas alterações. Quando pensamos em evolução de software no contexto de MBAs, o mais importante seria garantir, que para a maioria das necessidades de alteração impliquem na mudança de software na menor quantidade possível de microserviços.



 Rafael Mota Alves 10/09/2020

Para isso devemos usar uma forma de SRP (Single Responsibility Principle), para que código que muda junto esteja contido em um único serviço. Com isso, podemos adaptar o software com maior agilidade, principalmente quando temos times responsáveis por subconjuntos dos microserviços, em relação a regra da Complexidade Crescente, com a uma menor quantidade de lógica em cada microserviço, minimizamos a complexidade de cada serviço individual, e também facilitamos o processo de manutenção

 **Rafael Mota Alves** 10/09/2020

e redução de complexidade, já em relação a Regra Lei da Auto-Regulação, temos essa auto regulação para cada microserviços, o que pode ocasionar que alguns serviços possam ser tornar estáveis antes, e os times podem facilmente ser alocados de acordo com as necessidades de mudanças de cada subsistema.

 **Ramom Pereira dos Santos Silva** 12/09/2020

Partindo do princípio que cada microserviço foi construído seguindo bons padrões de projetos e utilizando o conceito de responsabilidade única, além de cada microserviço ter equipes responsáveis, acredito que as 3 leis teriam os seguintes impactos: 1^a Lei: com a atual dinâmica dos negócios digitais é natural que os serviços sejam modificados no seu ciclo de vida, no caso de microserviços, creio que essa lei teria impacto de acordo com sua concepção, ou seja, caso tenha sido construído com

 **Ramom Pereira dos Santos Silva** 12/09/2020

responsabilidades bem definidas, além de baixo acoplamento com outros microserviços, os impactos seriam minimizados, dado que não atingiria outras partes da aplicação, além do que a mesma equipe seria responsável por essas mudanças. Com microserviços, a evolução é mais tranquila, justamente pela característica de ser independente dos outros microserviços. Porém, vale salientar, que é necessário tomar cuidado com o acoplamento com outros serviços, caso contrário, os impactos seriam muito grandes

 **Ramom Pereira dos Santos Silva** 12/09/2020

2^a Lei: Nesse caso, é necessário que estejam bem definidas as responsabilidades daquele microserviço, com contratos bem estabelecidos com seus consumidores. Como cada microserviço tem uma pequena parte das funções da aplicação seria minimizada também, facilitando a manutenção. Porém, sempre é bom destacar que caso a complexidade de manutenção cresça muito e se torne inviável, talvez seja necessário dividir aquele microserviço em outros, dado que cada um deve ser responsabilidades bem definidas

 **Ramom Pereira dos Santos Silva** 12/09/2020

Com microserviços a complexidade é dividida em várias partes (dividir pra conquistar), não só sem responsabilidades da aplicação, mas também em equipes, onde cada microserviço é de responsabilidade de uma parte da equipe.

 **Ramom Pereira dos Santos Silva** 12/09/2020

3^a Lei: numa arquitetura de microserviços, cada um tem sua própria dinâmica de mudanças, já que possuem atribuições bem específicas e times responsáveis, então essa auto-regulação é dividida por microserviço, o que proporciona divisão de responsabilidades para os desenvolvedores. Mudanças sofrem menos impacto pela característica de separar bem as responsabilidades e desde que aquele papel seja cumprido na aplicação.

 **Ramom Pereira dos Santos Silva** 12/09/2020

E como o time já conhece o que cada microserviço faz, é mais tranquilo realizar alterações dado que o escopo é bem específico.

 **Saulo Guilhermino** 13/09/2020

A arquitetura de microsserviços, de maneira essencial, atende a primeira Lei conforme facilita que as aplicações sejam adaptadas de maneira pontual e simples; A segunda lei deve ser olhada com atenção, pois à medida que novos serviços são adicionados ao sistema, sua complexidade pode se tornar difícil de administrar, mas a característica distribuída facilita a manutenção;

 **Saulo Guilhermino** 13/09/2020

Já a terceira lei eu acredito que tenha se tornado menos forte na arquitetura de microsserviços, visto que a variedade de serviços possíveis de se adicionar ao sistema aumenta bastante a possibilidade de mudanças, não limita.

 **Lucas Barros** 13/09/2020

Na arquitetura de microsserviços a primeira lei se aplica bem à medida que mudanças no sistema (em seus módulos) podem ser realizadas de forma mais fácil (assumindo que a arquitetura seja construída a partir de boas práticas) e portanto facilita as adaptações na arquitetura, já que o desacoplamento das diferentes features permitem uma evolução mais independente.

 **Lucas Barros** 13/09/2020

A arquitetura de microsserviços também ajuda na segunda lei, já que, como cada parte da aplicação é um serviço separado, a complexidade de cada um deles demora para crescer. E caso venha a ficar muito grande, há a possibilidade de separar esse serviço em mais serviços.



 **Lucas Barros** 13/09/2020

Terceira lei: em microsserviços cada serviço que compõe o software possui sua dinâmica

terceira lei: em microserviços cada serviço que compõe o software possui sua dinâmica própria definida, isso meio que expande a ideia dessa lei, onde várias dinâmicas são utilizadas para formar o software

LC**Lucas Cardoso** 14/09/2020

Acredito que a adoção de microserviços se deve principalmente às 2 primeiras leis. Eles surgem como uma arquitetura para se adaptar a realidade atual de aplicações e se propõem a "diminuir" a complexidade das aplicações monolíticas. Ao reduzir a complexidade de parte do sistema, quebrando-o e modularizando-o, MBA, porém, adicionam complexidade à comunicação e gerenciamento.

LC**Lucas Cardoso** 14/09/2020

Sobre a terceira lei, acredito que o maior entrave para mudanças em microserviços é que pode ser necessário mudar diversos serviços, APIs e até se preocupar com a compatibilidade até de pequenas mudanças. Como, geralmente, serviços podem pertencer a times diferentes, orquestrar essas mudanças adiciona uma complexidade única a microserviços comparados com sistemas monolíticos

AF**Arthur Frade de Araújo** 15/09/2020

Como todo produto do mercado, um produto de software possui valor na medida em que proporciona algum tipo de solução relevante para os seus clientes. É de se esperar que conforme o tempo passe, surjam novas necessidades a serem supridas pelo negócio. A Lei da Modificação Contínua, portanto, se adequa bastante a essa característica peculiar a qualquer empresa. Ainda mais nos tempos voláteis em que vivemos.

AF**Arthur Frade de Araújo** 15/09/2020

Diversas estratégias gerenciais (métodos ágeis de gestão) e técnicas (arquitetura de microserviços) vêm sendo implementadas de maneira a tornar a evolução natural dos sistemas algo mais simples.

AF**Arthur Frade de Araújo** 15/09/2020

Dentro desse contexto, a arquitetura distribuída em microserviços se relaciona com a primeira lei (Modificação Contínua) no sentido de ser uma alternativa mais fragmentada de acompanhar as inevitáveis modificações das necessidades dos clientes e do produto digital.

AF**Arthur Frade de Araújo** 15/09/2020

Uma estrutura desacoplada permite que equipes, ou squads, cuidem de porções do produto (serviços) e atendam suas necessidades de modificação maneira mais direcionada. É por isso que MBAs, em geral, são aplicadas negócios de proporções razoáveis à grandes.

AF**Arthur Frade de Araújo** 15/09/2020

Isso ocorre, pois como bem determina a Lei da Complexidade, é natural que a complexidade do produto cresça, na medida em que as necessidades dos clientes tornam-se mais exigentes. Em se tratando de arquiteturas distribuídas é mais fácil gerenciar complexidades de pequenas partes isoladas, do que gerenciar um imenso e único repositório.

AF**Arthur Frade de Araújo** 15/09/2020

Dessa forma, a gestão da complexidade, determinada pela segunda lei, torna-se menos centralizada, permitindo que diversas equipes aumentem a complexidade geral do produto a partir de modificações em suas partes.

AF**Arthur Frade de Araújo** 15/09/2020

Por fim, não apenas a arquitetura de microserviços, mas sua gestão fragmentada irão fazer com que gradativamente, cada serviço ou grupo de serviços, possuam seus próprios stakeholders, e suas necessidades individuais, como determina a Lei da Auto Regulação. Porém isso ocorrerá em cada equipe.

AF**Arthur Frade de Araújo** 15/09/2020

O produto, então torna-se um grande "ecossistema" de equipes que, ao gerenciarem pequenas partes do código, unidas, gerenciam todo o produto.

Gabriel Ramos 15/09/2020

Dividindo entre as 3 leis, na primeira, podemos lidar de forma muito até simples. Com todos os sistemas funcionando de forma independente a adaptação, evolução, correção e manutenção de aspectos são muito mais fáceis de se fazer. Podendo sempre ou acrescentar novas tecnologias ou modificar as existentes para sempre está "Up to date". Já com a segunda lei, me parece algo muito óbvio, mas muito importante de se ter em mente, a complexidade de um sistema que inicialmente, vamos imaginar, era para+

Gabriel Ramos 15/09/2020

suportar uma rede de usuário de em média 5 mil pessoas por semana espalhado por 2 funcionalidades diferentes para esse mesmo sistema que agora tem 6 funcionalidades e 50 mil usuário diários, assim como o projeto desenvolvido vai dar certo e alcançar mais público a complexidade vai estar junto e com a arquitetura de micro serviços eu acho que a facilidade de conseguir expandir essa complexidade, com esforço explícito, é comparado a uma arquitetura monolítica muito mais "fácil"

C

RJ**Renato Joaquim de Miranda Ferreira** 15/09/2020

De maneira geral, os microsserviços atendem bem os requisitos da lei da modificação contínua, uma vez que o sistema estará subdividido em diversos módulos diferentes se comunicando entre si torna a evolução do sistema muito mais simples e menos traumática para seus desenvolvedores. Esse fato também é facilmente observável quando se olha do ponto de vista da 2 lei, pois conforme é mais 'fácil' evoluir também é mais 'fácil' aumentar a complexidade do sistema.

RJ**Renato Joaquim de Miranda Ferreira** 15/09/2020

Portanto é preciso sempre estar ciente do que se está sendo modificado, seus impactos no serviço e, consequentemente, adoção de novas tecnologias para lidar com as dores sofridas nesse processo.

HC**Heitor Carvalho** 16/09/2020

O fato de se escolher utilizar Microsserviços pode ser visto como uma consequência da primeira lei. É o software se adaptando para que continue a cumprir seu papel. Quando falamos de MBA's, podemos pensar em cada microsserviço como sendo um pedaço de software auto contido e independente. Então, A Lei da Complexidade crescente será inevitável, mas ao menos a complexidade adicionada estará fragmentada em diversos "mini programas" de forma que será mais fácil lidar com ela em pequenas doses.

HC Heitor Carvalho 16/09/2020

O que também acaba por se relacionar com a lei da auto regulação. Em um software fragmentado em pequenos serviços, cada um destes tem sua própria dinâmica de evolução, enquanto em um sistema monolítico haveria uma grande dinâmica (possivelmente mais complexa) que contemplaria todo o sistema.

Mateus Nunes 16/09/2020

Acredito que os microsserviços atendem de maneira coesa as 3 leis, a primeira lei se encaixa como uma luva no fundamento para construção do microsserviço com a descentralização da arquitetura a velocidade e a continua evolução é facilitada. Já na segunda lei vemos que se bem estruturado o estilo de arquitetura pode combater a complexidade continuamente constante, ou ao menos evitar que o nível de complexidade prejudique o desenvolvimento a ponto do software se tornar inviável para incrementos.

Mateus Nunes 16/09/2020

E sobre a auto-regulamentação isso ocorre e de maneira fragmentada, dessa maneira a evolução do software em termos de seus atributos e seus processos ocorre auto-regulamentada, mas em cada microsserviço em vez do software como um todo monolítico

HM**Hiro Miyakawa** 16/09/2020

O MBA é uma maneira de estruturar o serviço com dinamismo e adaptação. Logo, a primeira lei faz total sentido. A segunda, também é interessante, pois algumas aplicações preferiram voltar ao monolith pelos cons causados por microsserviços, então é algo a se antenar. A terceira pode ser interpretada para cada parte do microsserviço e para o todo.

IN**Izabella Nascimento** 16/09/2020

Acredito que as Leis da Modificação Contínua e da Complexidade Crescente são motivos de se aplicar microservices, a medida que o software cresce sua complexidade aumenta ao ponto que é necessário haver mudanças para suprir as necessidades. Já em relação a Lei da Auto-Regulação, como ela diz que o crescimento de um sistema segue a curva de distribuição normal, tendo um crescimento mais lento no início e no fim do seu ciclo comparado ao meio, creio que se adequa a microservices, pois ...

IN Izabella Nascimento 16/09/2020

como Ramom Silva falou, a complexidade de cada microsserviços é diminuída, e cada time sabe lidar com o problema específico do seu escopo.

CP**Claudio Pacheco** 16/09/2020

Uma MBA acaba por ser facilitadora do que defende a Lei da Modificação Contínua. Como é manifestado por esta lei, o SW sofrerá adaptações conforme for sendo usado, mas tendo em mente o objetivo original para a sua concepção. Visto que microsserviços acabam por utilizar uma estratégia similar ao 'dividir para conquistar', torna-se mais transparente a necessidade de adaptações. +

CP Claudio Pacheco 16/09/2020

Vejo a Lei da Complexidade Crescente como relacionada à primeira. Adaptações a serem realizadas acabarão por inserir um maior nível de empenho por parte dos times. Na MBA, abordagens como o CDC e Lei de Postel (conservador no envio, liberal no recebimento) acabam por reduzir a probabilidade defendida pela Lei da Complexidade Crescente de que a manutenção será inviável em algum momento. +

CP Claudio Pacheco 16/09/2020

Quanto à Lei da Auto-Regulação, acredito que seus efeitos na MBA estão relacionados não somente ao papel de um microsserviço, mas também a quem consome dele. Isto é, outros microsserviços podem acabar também definindo as tendências de manutenção daqueles dos quais depende. Em resumo, acredito que os efeitos desta lei são maiores em uma

C**NADA**

 Elverson Melo 16/09/2020

A Lei 1 parece estar completamente adaptada ao contexto dos microserviços, já que com esses o tempo da criação até o mercado diminuem, e fica mais fácil a realização da implantação contínua. Por outro lado, a complexidade cresce dado a integração dos diversos serviços, no entanto como cada serviço é construído e mantido independentemente por equipes distintas, os custos de manutenção e alteração não são tão grandes como num software monolítico, ...

 Elverson Melo 16/09/2020

e assim não devem escalar de forma a resultar na inviabilidade descrita na Lei 2. No MBA quando novos requisitos surgem e a complexidade passa a aumentar, um serviço pode ser repartido em mais microserviços, uma ação para redução da complexidade sem tanta dificuldade quanto no modelo tradicional.

 JS José Sheldon Brito Fekete 16/09/2020

O estilo arquitetural de microserviços casa muito bem com a 1º lei, onde a atualizações continuas é algo bastante importante no desenvolvimento e manutenção do sistema e esta arquitetura é algo mais fácil de fazer, tendo em vista que cada processo irá se auto moderar, certificando-se que suas funcionalidades estejam executando satisfatoriamente.

 JS José Sheldon Brito Fekete 16/09/2020

Já na 2º lei, como é trabalhado com processos independentes que possuem suas próprias equipes de desenvolvimento e manutenção, a complexidade irá se muito menor do que trabalhar em um sistema único.

 VS victor sena de lima attar 16/09/2020

Ambas as 3 leis são integradas com MBA pois quando os softwares crescem de complexidade (Lei da complexidade crescente), dividir em serviços auxilia a dividir a complexidade em vários lugares diferentes, a medida que é necessário criar novos serviços mais equipes podem trabalhar individualmente (Lei da auto-regulação) podendo ter sua dinâmica própria e individual para que possa crescer de forma mais fácil (Lei da modificação contínua)

 DL Danilo Lira 16/09/2020

Acredito, assim como Izabella, que essas leis são motivos para utilizar microservices. A primeira é bastante fácil de lidar quando utilizamos essa arquitetura, já que realizar mudanças nos serviços é mais simples e menos custoso. Já em relação à segunda acredito que uma aplicação de microserviços sofre pouco com esse problema, já que por maior que seja a quantidade de serviços a estrutura não é descaracterizada. Vejo que a última é a mais impactante para os serviços, pois afeta cada um deles

 MG Marcos Galvão 16/09/2020

Uma aplicação MBA consegue gerenciar tais leis muito bem, devido as características associadas a modularidade e interoperabilidade dos microservices, é possível controlar as mudanças constantes e o aumento de complexidade em uma perspectiva mais "local", dedicada individualmente aos serviços, sendo assim, o trabalho necessário para controlar tais características podem ser manuseados de maneira menos complexa. Por fim "Self Regulation Law" pode ser gerenciada com a capacidade de times ...

 MG Marcos Galvão 16/09/2020

distintos e especializados trabalharem em cada um dos microservices, de tal forma o ciclo de vida e peculiaridades individuais não influenciam de maneira direta no desempenho/resultado dos demais times ou serviços.

 VG Victor Gabryel 16/09/2020

Levando em consideração que a arquitetura de microserviços tem como característica uma maior flexibilidade faz com que essas leis sejam bem atribuídas ao seu ciclo de vida. Na primeira lei podemos ligar com a questão da modularização dessa arquitetura, onde se tem a facilidade de modificar e adicionar funcionalidades além da facilidade de disponibilização de novos serviços sem impactar o todo. A segunda lei fala sobre a questão de que se cuidados e medidas não sejam tomados, a complexidade +

 VG Victor Gabryel 16/09/2020

pode tornar um problema para futuras manutenções e vendo isso no lado de uma arquitetura de microserviços, tudo falado sobre a modularização faz com que manutenções sejam feitas de formas mais fáceis, onde os serviços pequenos possam ser tratados separadamente, diferente de um sistema monolítico. Acredito que, na terceira lei em um ciclo de vida de um sistema com arquitetura de microserviços, devido ao cada serviço ser trabalhado por equipes diferentes, possíveis tecnologias diferentes, +

 VG Victor Gabryel 16/09/2020

a dinâmica de cada um seja definida de forma singular.

 TM Talyta Maria Rosas Pacheco 16/09/2020

Devido à vantagem da arquitetura de microserviços de facilitar a inclusão de novas features, associa-se muito bem com o que a Lei da Modificação Contínua defende. Com a implantação das mudanças, resultará em maior complexidade como alertado na Lei da Complexidade Crescente, e há necessidade de um esforço maior para lidar com os novos desafios que essas

novas features podem implicar.

 **Talyta Maria Rosas Pacheco** 16/09/2020

Além disso, o impacto da inclusão de novas features deve possuir efeitos imperceptíveis tanto para o time de desenvolvimento quanto para os usuários, como defendido na lei da Auto Regulação.

 **Gabriel Cavalcanti de Melo** 16/09/2020

A utilização de microsserviços, garante ao software uma entrega contínua das funcionalidades. Sendo assim, podemos adaptar os softwares continuamente de maneira mais rápida e menos custosa, assim como diz a primeira lei. Também reduz os custos de manutenção que a segunda lei fala. E por fim, em relação a terceira lei, a arquitetura de microsserviços tem uma dinâmica própria, devido a comunicação e auto coordenação dos serviços.

Como você vê a relação e medição de forças entre o que diz a Lei da Conservação da Estabilidade Organizacional (1980), a Lei da Conservação da Familiaridade (1980) e a Lei do Crescimento Contínuo (1980) no contexto de MBA's?

 **Ramom Pereira dos Santos Silva** 13/09/2020

Acredito que hoje em dia, o software deve atender ao anseios dos clientes da empresa ao qual foi construído, atendendo o que eles precisam. Naturalmente, eles são modificados durante seu ciclo de vida. No contexto de MBA's, cada microsserviço tem uma responsabilidade no problema a ser resolvido pela aplicação, o que torna sua evolução mais tranquila, pois mudanças são feitas quando mais informações são inseridas para resolução daquela parte do problema.

 **Ramom Pereira dos Santos Silva** 13/09/2020

As 3 leis se relacionam bem, pois basicamente, o software evolui de forma constante, devem satisfazer as necessidades dos seus usuários e devem se adaptar ao ambiente externo. No contexto de microsserviços a evolução é feita com inputs vindo do usuário no contexto que ele atende, além disso, com subequipes bem definidas, esse processo se torna mais suave

 **Ramom Pereira dos Santos Silva** 13/09/2020

Em casos de crescimento muito grande dos microsserviços, é natural quebrá-lo em outros, para garantir que permaneçam com responsabilidades bem definidas e independência dos outros microsserviços da aplicação.

 **Lucas Barros** 13/09/2020

Cada uma dessas leis impõe, juntas, que para uma evolução máxima de um software, uma organização deve ter tomadas de decisões de forma invariável no quesito de velocidade, enquanto os profissionais que o evoluem possuam um bom domínio/familiaridade com o código e isso faz com que os usuários se mantenham satisfeitos com o produto.

 **Lucas Barros** 13/09/2020

No contexto de MBA's, isso se torna um pouco mais complexo pois as decisões de mudanças tem que ser tomadas envolvendo vários times, mas em compensação cada time tem uma maior familiaridade com cada um de seus módulos.

 **Lucas Cardoso** 14/09/2020

Acho que as Leis se mantêm com algumas peculiaridades. A Lei da Conservação da Estabilidade Organizacional, olhando para o sistema como um todo (achei engraçado a relação dela com as Sprints das metodologias ágeis). Porém o desenvolvimento individuais de serviços podem ter velocidades diferentes. Ex: digamos que um time tem 2 serviços como responsabilidade, eles podem acabar desenvolvendo apenas 1.

 **Lucas Cardoso** 14/09/2020

As outras duas leis, acredito que, no caso de MBAs, estejam intimamente ligadas ao conhecimento da arquitetura específica dos microsserviços, como eles estão organizados e como se comunicam. Já que essa é uma das maiores complexidades dessa arquitetura

 **Arthur Fraude de Araújo** 15/09/2020

Conforme explicado na introdução da resposta da pergunta 1 é natural que, com o tempo, o negócio tecnológico exija mudanças e mais funcionalidades. Considerando a Lei do Crescimento Contínuo, no contexto de MBAs, com o tempo, novos serviços surgirão na medida em que o negócio se mantém vivo.

 **Arthur Fraude de Araújo** 15/09/2020

Conforme dita a lei da Familiaridade, o crescimento descrito pela sexta lei, será regulado pela familiaridade que uma equipe possui com um produto. Considerando essas duas leis no contexto de microsserviços geridos por múltiplas equipes, é de se esperar que o grau de familiaridade seja maior, já que os desenvolvedores irão gerir porções do produto.

 **Arthur Fraude de Araújo** 15/09/2020

Consequentemente, mesmo se mantendo constante no tempo (Lei da Estabilidade Organizacional), eu acredito que o grau de evolução ocorre em maior velocidade em comparação a modelos mais antigos de gestão de desenvolvimento como o RUP (Rational

Unified Process). Times bem distribuídos e com alta familiaridade irão entregar com mais velocidade.

Gabriel Ramos 15/09/2020

Essas são leis bem intuitivas e de senso comum, acredito eu, principalmente as leis de Crescimento contínuo e Conservação de familiaridade são muito interligadas e fortes. Crescimento contínuo é uma ideia excelente para se aplicar no contexto de MBA, visto que um sistema vai ser escalado conforme o tempo for passando, a escalabilidade, levando em consideração que estamos no contexto ideal de implementação, deve ser algo feito de forma eficiente e aí pode entrar a ideia da Conservação de +

Gabriel Ramos 15/09/2020

familiaridade, se sua equipe tem não só conhecimento técnico mas já trabalha/trabalhou no projeto e tem familiaridade de sua regra de negócio e modelagem da arquitetura, essa equipe deve ser bem entendida do projeto, pela ideia de proporção de familiaridade com o tempo.

RJ

Renato Joaquim de Miranda Ferreira 15/09/2020

Pessoas mudam, culturas se moldam, processo esse que também acontece nos usuários utilizando os sistemas. Adaptabilidade e evolução são peças fundamentais para garantir que seu software não se torne obsoleto e caia no esquecimento. Esse contexto é muito bem versado na lei do crescimento contínuo e é um fato facilmente observável hoje em dia, onde softwares que utilizamos diariamente estão sofrendo mudanças semanais e/ou até mesmo diárias para garantir a melhor experiência possível para o usuário.

RJ **Renato Joaquim de Miranda Ferreira** 15/09/2020

Ainda sobre isso deve se atentar que para que tais fatos sejam possíveis é preciso observar o nível técnico da equipe, o quanto bem é possível entregar as atualizações e manter um trabalho saudável, portanto as 2 leis estão intimamente ligadas nesse processo. Além disso, falando sobre a lei da conservação da estabilidade organizacional, é possível notar que essa lei também está intimamente ligadas às 2 anteriores no sentido em que o usuário deve ter a melhor experiência possível usando o sistema.

RJ **Renato Joaquim de Miranda Ferreira** 15/09/2020

Portanto vejo as 3 leis bem equivalentes porém não constantes no sentido em que em determinados pontos do ciclo de vida do sistema é necessário focar mais um ponto e menos em outros para atingir os objetivos necessários.

Mateus Nunes 16/09/2020

A velocidade de transformação das necessidades dos clientes bem como da própria execução de alterações do software para atender essas necessidades é um fundamento importante para manter o software vivo e eficaz no seu nicho. As três leis fazem parte das 8 leis que Lehman indicou para uma abordagem mais humana na questão. Ao meu ver elas se complementam numa visão para maximização para entrega de valor para o usuário final

Mateus Nunes 16/09/2020

No contexto de MBA as leis se aplicam de maneira interessante: mudança continua vemos que o software deve ser constantemente atualizado, mas que a frequência dessa atualização depende da familiaridade do time com a ferramenta em questão, além do ciclo de vida do projeto que deve ser ampliado ao longo do tempo.

HC

Heitor Carvalho 16/09/2020

No contexto de MBA's a medição de forças dessas leis varia se considerarmos o sistema como um todo, onde, na média, elas provavelmente continuam aplicáveis (e possivelmente em um período maior de estabilidade considerando a atenuação na complexidade exposta na resposta a pergunta anterior). Agora, considerando cada microserviço individual como um sistema isolado, pode parecer em primeiro momento que as leis não se aplicam diretamente. Podem existir microserviços que ficarão um longo tempo (+)

HC **Heitor Carvalho** 16/09/2020

sem atualizações ou features novas, devido à própria natureza daquele microserviço estar cumprindo suas funções na totalidade, enquanto outros recebem novas features frequentemente ou até surgem novos microserviços. Nesse ponto, podemos interpretar que a lei do crescimento contínuo aplica-se não à adição de features, mas de novos microserviços, de uma forma constante no decorrer do tempo, o que manteria o esforço estável como posto nas duas primeiras leis, mas causando aumento de complexidade

HM

Hiro Miyakawa 16/09/2020

As três leis são das 8 leis de Lehman e fala sobre a constante evolução. A 1a fica mais fácil de interpretar quando vê o todo do microserviço conectado, e não apenas de partes isoladas, e a soma do crescimento dela ser algo constante. A 2a, também faz sentido, principalmente quando a gente pensa em partes mais importantes e onde entregamos mais valor no final. Isso pode ir variando conforme os requisitos que vai mudando e logo o valor de entrega que vai mudando. +

HM **Hiro Miyakawa** 16/09/2020

A 3a, acredito que seja a mais óbvia e direta para microserviço, quando a gente pensa na constante adaptação e entrega contínua.

C

IN

Izabella Nascimento 16/09/2020

Estas três leis estão totalmente interligadas, no desenvolvimento de um software é lógico associar que a taxa de evolução de um software está ligada ao grau de familiaridade da equipe, e principalmente no sucesso de aplicar microservices. A Lei do Crescimento Contínuo se adequa a microservices pois um software está em constante mudança, e deve ser implementado funcionalidade por funcionalidade, visando o que dá mais valor para o cliente e expandindo no decorrer do seu ciclo.

IN

Izabella Nascimento 16/09/2020

Por último, é a Lei da Conservação da Estabilidade Organizacional é bem aplicada pois como temos a divisão em vários serviços, é necessário times pequenos de trabalho, provendo uma estabilidade no desenvolvimento.

EM

Elverson Melo 16/09/2020

Ao meu ver a Lei da Conservação da Estabilidade Organizacional (Lei 4) e Lei da Conservação da Familiaridade (Lei 5) tem menos força no contexto de MBA que a Lei do Crescimento Contínuo (6). Como nesse contexto temos um maior número de equipes cada uma focada num microserviço, não ficamos com uma única média de atividade, agora temos várias médias, cada uma de uma equipe em particular. Além disso como cada time tem independência para fazer o deploy de seu serviço e não mais fica refém do ...

EM

Elverson Melo 16/09/2020

lançamento de grandes releases, é possível que não haja uma estabilidade na atividade do software durante o seu ciclo de vida. Outro ponto da divisão em equipes é que não é preciso um membro de uma equipe estar familiarizado com todas as funcionalidades de todo o software. Agora ele pode focar na familiaridade com o microserviço que é ele responsável. Logo se muitas mudanças forem realizadas no software num determinado tempo, não é necessário grande custo de tempo para que todos os ...

EM

Elverson Melo 16/09/2020

desenvolvedores fiquem familiarizados com todas as mudanças. Isso permite que a quantidade de novo conteúdo em cada release sucessivo de um sistema não necessariamente permaneça constante. A lei 6 ainda permanece válida dependendo da granularidade analisada, no contexto do sistema de informação como um todo ela é obedecida já que a quantidade de funcionalidades continua aumentando com o tempo, já no contexto dos microserviços individualmente ela perde força, pois nem todos eles precisam estar...

EM

Elverson Melo 16/09/2020

entregando novas funcionalidades continuamente.

CP

Claudio Pacheco 16/09/2020

As três leis em conjunção acabam por demonstrar que uma MBA reduzem a probabilidade dos efeitos negativos previstos no processo de evolução e manutenção do SW. A Lei da Conservação da Estabilidade Organizacional acaba por ratificar que a "pulverização" dos times e responsabilidades em uma MBA auxilia ainda mais na imperceptibilidade da mudança de recursos ou pessoas na evolução do SW. +

CP

Claudio Pacheco 16/09/2020

Em relação à Lei da Conservação da Familiaridade, visto que a complexidade de um microserviço é menor em comparação a todo um sistema monolítico, nota-se que a obtenção do domínio do conteúdo se torna menos custoso. Portanto, a manutenção e evolução é facilitada. +

CP

Claudio Pacheco 16/09/2020

Quanto à Lei do Crescimento Contínuo, visto que novas alterações/funcionalidades/correções são mais pontuais em microserviços, o aumento do conteúdo funcional do SW passa a ser um problema menos proeminente em comparação à arquitetura monolítica. Novos componentes, por exemplo, podem tornar-se um terceiro microserviço, reduzindo, portanto, os efeitos quando comparados a um SW monolítico.

VS

victor sena de lima attar 16/09/2020

Como os módulos da aplicação são divididos em serviços menores, equipes menores tem maior familiaridade com essa parte da aplicação (Lei da conservação da familiaridade) e com arquiteturas de microserviços, se consegue criar mvp's utilizando api's externas que podem evoluir mais rápidos para sistemas internos em serviços considerando que o software vai ser ampliado, bastando ser adicionado mais serviços (Lei do crescimento contínuo)

vs

victor sena de lima attar 16/09/2020

E com o somatório dessas arquiteturas divididas e o conceito de CI/CD consegue-se manter um software sempre em evolução constante (Lei da Conservação da Estabilidade Organizacional)

VG

Victor Gabryel 16/09/2020

Acredito que essas três leis citadas tem grande relação com os MBA's, pois pode-se analisar que na Lei da Conservação da Estabilidade Organizacional é importante devido ao desenvolvimento dos serviços serem independentes, com isso cada um desses serviços possuem um time para que cada um deles possa estar inteiramente focado na sua evolução. A

Lei da Conservação da Familiaridade tem relação de como cada time que está desenvolvendo cada microserviço, deve manter um grau de intimidade com as +

 Victor Gabryel 16/09/2020

ferramentas e também a arquitetura. E a Lei do Crescimento Contínuo se encaixa justamente pois a arquitetura de microserviços modulariza essas funcionalidades para que, com a demanda dos clientes, possa ser desenvolvido com maior excelência além de buscar criar esses serviços que irão acrescentar no sistema.

 Danilo Lira 16/09/2020

As três leis estão completamente ligadas. A lei de conservação da estabilidade faz bastante sentido no contexto das MBA, pois a troca de recursos poderá afetar apenas alguns serviços e ter pouco impacto no software como um todo, já a lei de conservação de familiaridade faz muito sentido nesse contexto porque os serviços, em geral, são pequenos o suficiente para garantir que os desenvolvedores possuam domínio sobre seu conteúdo. Vejo que a lei de crescimento contínuo é o motivo de existirem as...

 Danilo Lira 16/09/2020

outras duas, pois é porque um software cresce que precisamos nos preocupar com a estabilidade e familiaridade do desenvolvimento, e ao meu ver isso é bastante amenizado no contexto de MBA.

 Marcos Galvão 16/09/2020

As três leis possuem um caráter associado a evolução constante da aplicação, já que tanto a aplicação, bem como os indivíduos associados a ela, devem estar em constante evolução para atingir a "satisfação". Acho extremamente relevante a perspectiva de que as aplicações MBAs devem, idealmente, estar em constante evolução e por tal motivo demandam desenvolvedores que também estejam aptos para tal, levando-se em consideração que o crescimento dos serviços está limitado nas habilidades do time.

 José Sheldon Brito Fekete 16/09/2020

As três leis se adequam muito bem ao desenvolvimento de sistemas MBA's, como equipes são designadas para serviços independentes entre si, cada membro terá um conhecimento mais amplo e especializado da sua funcionalidade, e o MBA prega a maior facilidade de acréscimos de novas funcionalidades, tendo assim a evolução do sistema como um todo.

 Talyta Maria Rosas Pacheco 16/09/2020

As 3 leis estão associadas a responder mudanças incrementais no ciclo de desenvolvimento de um Software. Apesar de o MBA responder bem às modificações, ainda há um esforço para que equipes que compartilham dos mesmos serviços, como no compartilhamento de módulos, estejam a par dos efeitos, e assim sejam afetados com o menor grau possível.

 Gabriel Cavalcanti de Melo 16/09/2020

Utilizar microserviços reduz o tempo de entrega de novas features e manutenção. Isso ajuda a manter o software funcional, entregando funcionalidades que não foram previstas, correções e melhorias, de forma muita mais rápida. Desta forma, as três leis estão bastante ligadas a MBAs

Quais as relações diretas e indiretas entre as consequências da Lei da Qualidade Declinante Modificação Contínua (1996) e da Lei da Realimentação do Sistema (1996) no contexto de MBA's?

 Rafael Mota Alves 11/09/2020

Eu diria que MBA's ajudam a combater as consequências da lei de Modificação Contínua, pois elas permitem trazer funcionalidades e melhorias novas a grande aplicações de forma mais rápida, justamente por causa da natureza de microserviços que permitem que múltiplos times desenvolvam o sistema simultaneamente. Quanto a Lei de Realimentação do Sistema, os MBAs também ajudam no sentido de que permitem que os feedbacks dos usuários cheguem mais rapidamente, visto que MBAs possibilitam técnicas como

 Rafael Mota Alves 11/09/2020

Integração Continua e Implantação Continua.

 Ramom Pereira dos Santos Silva 13/09/2020

Acredito que a dinâmica dos negócios digitais atualmente exigem que os software sejam readequados a todo momento. As necessidades de negócios, e acima de tudo, dos usuários daquele software alteram ao longo do tempo, o que faz com que as empresas modifiquem seus softwares para atender essas expectativas. MBA's se relacionam bem com as 2 leis, pois, normalmente, diversos times trabalham individualmente um um conjunto de microserviços, o que contribui para rápida manutenção e atualização

 Ramom Pereira dos Santos Silva 13/09/2020

dos serviços. No contexto atual de negócios digitais, o feedback é constante, tanto da própria equipe, mas principalmente dos usuários da aplicação, o que está em sintonia com a Lei de Realimentação do Sistema, isso garante que a aplicação, como é MBA's, os microserviços por consequência, sejam modificações constantemente para atender essas expectativas, o que faz com que a Lei da Qualidade Declinante Modificação Contínua seja atendida também

atendendo também.

RP Ramon Pereira dos Santos Silva 13/09/2020

Como a natureza dos microserviços é que tenham responsabilidades bem definidas e normalmente, várias equipes enxutas trabalhem paralelamente em diversos microserviços, faz com que os feedbacks sejam incorporados ao sistema de forma mais rápida. Nos dias atuais, o feedback dos usuários são extremamente importantes.

SG Saulo Guilhermino 13/09/2020

Uma MBA tem como vantagem a segregação de tarefas, que acarreta na facilidade de manutenção e aprimoramento. Como as diversas partes de um sistema funcionam de maneira praticamente independente, adaptações tendem a ocorrer de maneira peculiar em cada tipo de sistema, em decorrência da criticidade de cada serviço, de seu comportamento em situações de downtime, etc.

SG Saulo Guilhermino 13/09/2020

Na minha visão, isto por si só está de acordo da lei da Qualidade Declinante, pois é inerente a qualquer sistema a perda de qualidade com o tempo (que acarreta em sua substituição em alguns casos). Porém, como dito, a facilidade de aprimoramento de uma MBA torna a manutenção muito menos dolorosa para o desenvolvedor e parte constante do processo de desenvolvimento.

SG Saulo Guilhermino 13/09/2020

Assim, entramos na Lei da Realimentação do Sistema, que impacta ainda mais forte nas MBAs, visto que agora as diversas camadas de um sistema estão ainda mais bem definidas em cada microserviço. Sendo assim, o processo de desenvolvimento deve levar muito em conta cada camada para que o propósito do sistema seja atingido com completude, exatidão e eficiência.

AF Arthur Frade de Araújo 15/09/2020

Acredito que de alguma maneira a aplicação do sistema de feedback pode limitar o grau de declínio descrito pela Lei da Qualidade Declinante. A Lei da Realimentação do sistema, ao considerar retornos relacionados a latência e tempo de execução das funcionalidades, por exemplo, pressupõe que a solução desse tipo de feedback exija implementações mais arrojadas consequentemente mais "elegantes".

AF Arthur Frade de Araújo 15/09/2020

Mas claro que essa consequência positiva só será alcançada com a devida priorização da resolução desses comentários, por isso imagino que essa relação feedback e qualidade seja mais indireta. É possível que uma empresa possua uma excelente visibilidade dos retornos dos usuários, mas se ela não parar para avaliá-los e resolvê-los, a qualidade tenderá ao declínio.

AF Arthur Frade de Araújo 15/09/2020

Uma consequência direta que as MBAs trazem para essas leis é facilidade da fragmentação (dividir para conquistar). A distribuição dos retornos de stakeholders e usuários entre múltiplas equipes permite que a gestão de feedbacks seja mais simples. Bem como a criação de sistemas com maior qualidade.

RJ Renato Joaquim de Miranda Ferreira 15/09/2020

Em minha visão essa relação é facilmente observada durante o processo de uso do sistema, no qual usuários geram suas dúvidas, relatórios de uso, feedbacks, expectativas, etc. Portanto o time responsável pelo sistema deve ter bons mecanismos de captura, tratamento e, talvez o mais importante, geração de conhecimento para manter seu software cada vez mais atualizado e que continue agradando seus usuários durante seu uso.

RJ Renato Joaquim de Miranda Ferreira 15/09/2020

Caso essa relação não exista, ou seja mal feita, o software tende a ficar em desuso, gerando experiências desagradáveis tanto para os usuários quanto para o próprio time de desenvolvimento, até o ponto em que pode se tornar obsoleto. É preciso boas análises sobre o que está sendo gerado para que seu produto se torne cada vez melhor.

Mateus Nunes 16/09/2020

Vejo que atualmente os negócios possuem uma velocidade de feedback dos usuários indiretamente ou diretamente de maneira constante, além desse, a própria equipe e o mercado traz uma necessidade de adaptação e evolução constantemente, especialmente quando é possível as alterações e distribuições de atualizações em massa do software. Dessa maneira entendo que a utilização de MBA é importante para atender esses feedbacks de maneira mais ágil, podendo assim distribuir as constantes mudanças que

Mateus Nunes 16/09/2020

são provindas desses feedbacks de maneira coesa. Evitando que o valor entregue pelo software pare de ser proporcional as expectativas que os usuários têm sobre a aplicação

HC Heitor Carvalho 16/09/2020

No Contexto de MBA's ambas as leis falam de como a complexidade crescente do software tende a tornar o desenvolvimento cada vez mais custoso. A arquitetura pode tentar mitigar isso através dos múltiplos serviços, mas a complexidade sempre será inevitável e existirão sempre serviços que serão mais demandados e exigirão mais mudanças pelos clientes, de forma a aumentar a complexidade e o esforço para modificação. Isso pode levar a uma reestruturação

do projeto ou adição de novo serviço.

EM

Elverson Melo 16/09/2020

Ambas as leis estão interligadas, como consequência da lei 8 o feedback dado pelos usuários vai sugerir aos desenvolvedores a adaptação do sistema, com a retirada e a inserção de funcionalidades. Se as mudanças forem implantadas isso impedirá o declínio do software, pois ele vai estar sempre adequado às necessidades atuais. No contexto de MBA ambas as leis

resultarão no aumento do número de microserviços de uma aplicação, ao mesmo tempo que alguns deles serão eliminados.

Elverson Melo 16/09/2020

Como as alterações são mais ágeis no ambiente de MBA e o tempo entre desenvolvimento e o mercado é menor, isso pode reduzir o tempo de vida de alguns microserviços, mas impede o declínio da aplicação, pois rapidamente ela consegue atender a demanda do mercado.

IN

Izabella Nascimento 16/09/2020

Essas duas leis estão relacionadas diretamente, como já foi falado, os softwares estão em constante construção, e devem se adequarem ao surgimento de necessidade de mudanças, e em microservices essa adequação se torna mais fácil pelo fato do sistema estar dividido, e um processo de sistema de feedback atualmente se vê como uma ferramenta essencial para permitir melhorias nos softwares, sendo até impensável não ser realizada.

HM

Hiro Miyakawa 16/09/2020

Acredito que tem uma relação de "sintoma" e "exame". A qualidade, ou seja, a aplicação tem que sempre entregar valor ao usuário e se manter útil. Os feedbacks e capturas ajudam a entender esse contexto, e o MBA com a sua estrutura de adaptação, facilita esse entendimento e modificação.

MG

Marcos Galvão 16/09/2020

Exergo ambas as leis associadas a um ciclo bem padrão na vida de um software e creio que MBA permite que elas sejam melhor administradas ao longo do tempo, por exemplo, para a "Declining Quality Law", a capacidade dos microservices evoluírem, trocarem suas tecnologias/técnicas e recompor-se para evoluírem fornecendo um bom meio garantir a constante evolução, ademais, por seu escopo de atuação ser muito mais limitado que uma aplicação por completo, o processo de manutenção pode ser garantido...

Marcos Galvão 16/09/2020

mais facilmente. A "Feedback System Law" está intrinsecamente relacionada com a lei anterior, pois a constante iteração entre os serviços, desenvolvedores e usuários pode prover meios de identificar onde e como melhor a aplicação já existente, e devido às características dos microservices tais melhorias podem ser pontualmente implementadas de maneira mais direta.

CP

Claudio Pacheco 16/09/2020

Julgo que, apesar de ser em menor escala em comparação com uma arquitetura monolítica, as consequências da Lei da Qualidade Declinante Modificação Contínua ainda podem se fazer presentes nas MBAs, especialmente se as mudanças forem frequentes. Supondo que há um serviço do qual dependem muitos outros serviços, alterações realizadas nele durante a evolução natural do SW acabarão por exigir mais recursos. Por diversos motivos, falhas no processo de qualidade do serviço alterado e nos seus +

Claudio Pacheco 16/09/2020

dependentes acabam por reduzir o nível de qualidade do SW como um todo. Já em relação à Lei da Realimentação do Sistema, como há um número de comunicações maiores e maior dependência numa MBA, o feedback para um serviço A que parece não ter relação direta com um serviço B pode ter grande carga de influência. Assim sendo, acredito que o crescimento de um SW baseado em microserviços acaba por ser definido por muitos fatores devido à sinergia esperada na integração entre os serviços.

VG

Victor Gabryel 16/09/2020

Devido à alta flexibilidade na arquitetura de microserviços, acredito que devido à necessidade de crescimento contínuo, a segunda lei citada exemplifica que a captação de feedback's faz com que os serviços estejam em constante checagem para que existam mudanças para que se adaptar às necessidades atuais se necessário. Com isso, a primeira lei tem um relacionamento com ela, devido ao fato dessas mudanças vistas como necessárias, façam com que o software possua uma constante relevância +

Victor Gabryel 16/09/2020

solucionando os problemas que ele se propõem solver.

VS

victor sena de lima attar 16/09/2020

Com o mundo real sempre crescendo e mudando, o software não pode ficar estagnado pois perde de entregar valor ao usuário, e com MBA se consegue entregar mais rápido e mais fácil com CI/CD e testes automatizados



Dudu Lira

Dudu Lira 16/09/2020



DANIELLO LIMA 10/09/2020

Um software vai perdendo qualidade com o passar do tempo e por isso precisa estar em constante manutenção e evolução. As consequências disso são várias modificações no código da aplicação que no contexto dos microsserviços são facilitadas em comparação aos monólitos. Então acredito que as aplicações nesse contexto tem o potencial de sofrer menos com a perda da qualidade



JS José Sheldon Brito Fekete 16/09/2020

As 3º leis citadas fazem parte do ciclo natural de um sistema, sendo ele monólito ou MBA, assim como qualquer software, os MBA's não escapam de ter uma vida útil, necessitando de mudanças conforme o passar dos tempos com a ampliação de necessidades dos clientes e feedback dos mesmos, porém as mudanças em sistemas MBA's tende a ser mais simples, sofrendo menos com a perda da qualidade conforme o tempo.



TM Talyta Maria Rosas Pacheco 16/09/2020

Como já dito, o processo de evolução de software requer esforços tanto no contexto de desenvolvimento de um monolítico quanto de microsserviços. A qualidade do software está diretamente ligada a como o time saberá lidar com as mudanças, seja com testes, comunicação entre os times etc



GC Gabriel Cavalcanti de Melo 16/09/2020

Acredito que ao utilizar microsserviços a perda de qualidade diminui bastante, pois essa arquitetura permite uma enorme capacidade de se adaptar, assim como a lei da modificação contínua diz.



Com base no entendimento das 8 leis de Lehman, quero saber se a sua percepção sobre o que representa um caso de sucesso e de falha no desenvolvimento de MBA's mudou? O que representa uma falha épica, na sua opinião (justifique)?



LB Luan Brito 10/09/2020

Sinto que sabendo das leis tenho um maior embasamento para dizer o que é um sucesso ou falha em potencial, mas não só no contexto de MBA's mas de softwares como um todo. Na minha opinião uma falha épica num contexto de MBA é uma arquitetura mal elaborada, com papéis mal distribuídos, ou pior: pouco claros.

LB Luan Brito 10/09/2020

A justificativa é que vai ser difícil construir, manter e adaptar um projeto assim... a complexidade vai crescer cada vez mais e em pouco tempo sua manutenção vai ser inviável



RM Rafael Mota Alves 11/09/2020

Eu diria que a leis de Lehman reforçaram a minha percepção sobre a importância da capacidade de sistemas de software evoluirem ao longo do tempo, e acredito um dos pontos fortes das MBA's é justamente possibilitar a evolução de sistemas complexo, portanto sendo sucesso e falha seria definido pela capacidade de evoluir o sistema. Então, acredito que uma falha épica no contexto de MBAs seria um sistema baseado em microsserviços que seja difícil de modificar, pois nesse caso teríamos todas as

RM Rafael Mota Alves 11/09/2020

desvantagens dos microsserviços, mas sem uma das características mais importantes.



RP Ramom Pereira dos Santos Silva 13/09/2020

As 8 leis de Lehman contribuem para um entendimento mais geral, sobre manutenção, em qualquer tipo de arquitetura. Acredito que casos de falha em Microsserviço seja quando eles são mal elaborados, sem um critério bem definido, e são apenas utilizados por estarem no hype, e não pela necessidade de negócio da empresa, microsserviços com diversas responsabilidades, diversas pessoas trabalhando no mesmo microsserviço, pode tornar o processo de funcionamento um grande dor de cabeça.

RP Ramom Pereira dos Santos Silva 13/09/2020

Em contrapartida, quando um critério bem definido, seguindo necessidades de negócio (e.g. DDD), contextos e responsabilidades bem definidas, e equipes enxutas trabalhando em alguns microsserviços, acredito que toda a empresa terá grandes benefícios com essa abordagem, acredito que isso caracteriza um sucesso com MBA's. Acredito que uma falha épica nesse contexto é quando os microsserviços estão fortemente acoplados aos outros, o que torna o entendimento e manutenção muito complicado para os deve.

RP Ramom Pereira dos Santos Silva 13/09/2020

Dito isso, acredito que minha percepção sobre o tema foi amplificada com as Leis de Lehman, dado que já tinha um conhecimento inicial sobre o tema.



LB Lucas Barros 13/09/2020

MBA's de sucesso estão bastante atreladas as leis apresentadas. Uma MBA ideal, isto é, que (ao meu ver) segue os princípios de responsabilidade única, possui baixo acoplamento, os times que cuidam de cada módulo possuem uma boa comunicação entre si, etc... tende a ser bastante favorável às leis de Lehman.

LB Lucas Barros 13/09/2020

Uma falha épica se dá em um sistema com o inverso do que citei acima, porque que nessa situação o sistema torna-se desfavorável a todas as 8 leis, que possuem pontos cruciais para a evolução e manutenção de um software.

LB Lucas Barros 13/09/2020

Então sim, as leis influenciaram bastante na minha visão sobre casos de sucesso e falhas de MBAs.

LC Lucas Cardoso 14/09/2020

Olhando para as leis de Lehman, acredito que um caso de sucesso seria um sistema bem modelado, dividido entre times com responsabilidades bem definidas, com relativa flexibilidade para expansão (de novas funcionalidades), porém que mantém a complexidade sob controle. Um fracasso, como Rafael falou, seria um sistema mal modelado que traria apenas as desvantagens da arquitetura, tornando mudanças muito difíceis e a complexidade altíssima.

AF Arthur Frade de Araújo 15/09/2020

Considerando as 8 leis de Lehman eu entendo que arquitetura de software sozinha, não é suficiente para fazer um produto evoluir de maneira saudável. Produtos desenvolvidos seguindo os padrões de MBA serão bem sucedidos na medida em que a sua gestão também é descentralizada. Só assim será possível mitigar complexidade em âmbito geral. E isso me leva ao que eu entendo por falha épica:

AF Arthur Frade de Araújo 15/09/2020

Implementar arquitetura baseada em microsserviços sem que a empresa possua estrutura organizacional para gerir essa estratégia. A arquitetura por si só é bastante interessante, porém, desvincilhada de uma gestão capaz de supri-la pode ser um empecilho ainda maior para a evolução, principalmente, de pequenos negócios.

RJ Renato Joaquim de Miranda Ferreira 15/09/2020

Não diria que mudou minha percepção mas sim organizou e formalizou melhor conceitos que aprendi durante a experiência que tive tanto profissional quanto acadêmica. Os conceitos melhor mostrados aqui me ajudaram a perceber como deve ser tratado todo o ciclo de desenvolvimento/manutenção de um software, qualquer que seja. Em minha opinião uma falha épica seria o que aconteceu com o Orkut tempos atrás, que era uma febre de uso e caiu no esquecimento até ser descontinuado.

Mateus Nunes 16/09/2020

Acredito que a introdução das 8 leis de Lehman ajudou a percepção e reflexão nos pontos importantes para a flexibilização, construção e crescimento de softwares. Dessa maneira a forma como se arquiteta um software como MBA por si só não traz garantias de um boa continua evolução, dessa maneira acredito que os conceitos apresentados fundamental a percepção de sucesso, mostrando que é uma combinação entre uma arquitetura bem construída e uma gestão otimizada para o crescimento

Mateus Nunes 16/09/2020

Falhas épicas pra mim seriam num quesito em que a arquitetura não corrobora pra a constante evolução e que o mercado e usuários se transformam e o software não, mesmo que o desejo de transformação ainda exista. Porém acredito que mesmo nesses casos existem justificativas plausíveis que podem não significar o fim do software, por exemplo a necessidade de agilidade na construção de MVP por startups, que guiam a tomada de decisão de maneira tão veloz que equipes sem experiência podem experimentar

Mateus Nunes 16/09/2020

verdadeiros fracassos de arquitetura e crescimento, mas que mesmo assim conseguiram atingir o seu objetivo de validação de um software

HC Heitor Carvalho 16/09/2020

Minha percepção mudou no sentido da forma como olhamos para as 8 leis dentro do contexto de como o software é organizado. Em um sistema monolítico com apenas uma equipe de desenvolvimento a relação era simples e direta. Numa arquitetura MBA, é possível utilizar várias lentes de granularidade (sistema como entidade analisada / serviço como entidade analisada) para interpretar a aplicabilidade das leis, o que pode ajudar no planejamento do desenvolvimento, uma vez que elas ainda se mantém verdade.

IN Izabella Nascimento 16/09/2020

Essas leis são geralmente seguidas atualmente sem sabermos que a estamos usando, ter o conhecimento formal delas me fez acreditar em mais casos de sucesso sendo aplicadas em microservices. Uma falha épica para mim depende muito do tipo de aplicação e de onde está no desenvolvimento, como foi falado por Renato Ferreira a falha do Orkut não levou tanto em consideração o sistema de feedback e acabou ficando no esquecimento e sendo substituída, entretanto creio ser uma falha épica não ter uma boa...

IN Izabella Nascimento 16/09/2020

estrutura de gestão e planejamento tornando difícil fazer mudanças para se adequar a novas ferramentas.

HM Hiro Miyakawa 16/09/2020

Acredito que as 8 leis ainda se mantém coerente com o mundo que vivemos, e a MBA é uma estrutura que consegue se manter coerente às leis no mundo onde os serviços precisam de um rápido crescimento e adaptação. Isso vem de constante monitoramento dos feedbacks e capturas com analytics para entender o comportamento e necessidade dos usuários, principalmente se for B2C que exige um input muito maior de usuários. As redes sociais obsoletas que os colegas citaram é um bom exemplo disso. +

HM Hiro Miyakawa 16/09/2020

+uma outra falha épica seria serviço como Uber não conseguir se adaptar rapidamente e conseguir entregar valor aos clientes (pagantes) e usuários (motoristas), conforme aparecem novas leis, usuários, ambientes diferentes (como a de barco que é usado em alguns países asiáticos). A não percepção e adaptação também dificultaria a criação de serviço como Uber Eats, aproveitando os assets existentes.

JS José Sheldon Brito Fekete 16/09/2020

Não mudou, mas formalizou um pensamento que eu já vinha maturando com os aprendizados em âmbito acadêmico e principalmente nesta cadeira. A falha épica no desenvolvimento de MBA's acontecem quando ocorre o mal planejamento do sistema, onde não é pensado no futuro e sim fazer o mais rápido possível, sem ir em busca do conhecimento técnico necessário.

CP Claudio Pacheco 16/09/2020

As 8 leis de Lehman ajudaram a ter mais clareza sobre princípios que são essenciais para arquitetar uma boa MBA. Como essas leis dizem respeito à evolução e manutenção de um SW ao passar do tempo, foi útil refletir sobre problemas futuros que, muitas vezes, não são levados em conta adequadamente na concepção de um sistema, independentemente do estilo arquitetural definido. +

CP Claudio Pacheco 16/09/2020

Na minha opinião, uma falha no desenvolvimento de MBAs é o mau dimensionamento dos limites de cada serviço. Ou seja, estabelecer serviços que possuem papéis demais ou de menos podem elevar o nível de complexidade do sistema como um todo no decorrer do seu ciclo de vida.

EM Elverson Melo 16/09/2020

O conhecimento das leis não mudou, mas fortaleceu a base teórica sobre desenvolvimento e ciclo de vida de sistemas. Acredito que uma falha épica seja tentar implementar uma arquitetura baseada em microserviços, mas resultar em múltiplos "mini serviços monolíticos" na prática. Onde apenas alguns problemas relacionados a escalabilidade seriam resolvidos, mas os problemas de desenvolvimentos continuariam existindo, como ciclos de integração longos que dificultam o deploy do sistema. ...

EM Elverson Melo 16/09/2020

Neste caso, passariam a existir todas as dificuldades dos microserviços, porém benefícios como a redução do time to market (necessário para o rápido atendimento dos feedbacks obtidos) ou a superação das restrições da lei 4 e lei 5 estariam ausentes.

VS victor sena de lima attar 16/09/2020

Na minha concepção, para se ter sucesso na construção de um MBA deve-se medir pelo valor ao usuário, e o usuário sempre muda e evolui, pois está inserido em um ambiente evolutivo, então as leis são para postular essa idéia de sempre estar em constante mudança e melhoria para sempre continuar entregando mais valor aos usuários.

DL Danilo Lira 16/09/2020

Acredito que minha percepção não mudou. Uma falha épica nesse contexto, para mim, é desenvolver uma aplicação sem pensar no seu crescimento. Todas as aplicações precisam crescer e se adaptar para se manterem úteis, sendo assim, desenvolver um sistema muito acoplado e mal estruturado pode ser considerado uma falha épica. Idealmente devemos pensar em como tornar a aplicação mais flexível para aceitar novas mudanças e a arquitetura tem potencial para fornecer essa flexibilidade

VG Victor Gabryel 16/09/2020

Acredito que essas leis me ajudaram a fixar e compreender melhor alguns pontos fundamentais para o ciclo de vida de um software, para assim poder tê-lo em contínuo crescimento se mantendo satisfazível para os usuários mesmo com o passar do tempo. Uma falha épica na minha opinião seria na questão organizacional, onde uma empresa que adote essa arquitetura e tenha um time que não tem maturidade para fazer o gerenciamento dos serviços e capacidade de fazer entregas contínuas pode afetar +

VG Victor Gabryel 16/09/2020

totalmente no desenvolvimento do software e nas suas manutenções.

MG Marcos Galvão 16/09/2020

Assim como Sheldon a minha visão também não mudou, porém as leis proveram uma definição um pouco mais formal do porque as características dos microservices, se bem aproveitadas, permitem uma longevidade bastante satisfatória para o sistema. Enxergo uma

rainha épica em MBAs quando os microservices são mal projetados e adquirirem uma dependência grande entre si. A longo prazo tudo indica que tal sistema pode virar um emaranhado de serviços que podem sofrer dos mesmos problemas que algum Monólitos

 **Marcos Galvão** 16/09/2020

tornando-se difícil de modificar os serviços já existentes, prover-lhes a devida manutenção, otimizar suas comunicações e ademais acabar sofrendo de problemas como delay de conexão ou queda de serviços essenciais, o que causaria um efeito cascata nos demais.



Talyta Maria Rosas Pacheco 16/09/2020

Minha percepção não mudou. O gerenciamento de falha e sucesso em uma arquitetura de microserviços deve levar em consideração pontos tanto quanto em monolíticos. Questões sobre planejamento, comunicação, e saber arquitetar bem os microserviços também precisam ser considerados para que não acarrete em uma falha grave.



Gabriel Cavalcanti de Melo 16/09/2020

Essas leis não mudaram minha percepção, mas afirmaram que a arquitetura de microserviços tem uma grande importância na construção de um software à medida que ele ficar maior e mais complexo. Eu acredito que uma falha épica seria construir um software no estilo arquitetural de microserviços sem pensar em como vamos gerenciar a comunicação entre esses serviços.

Escalabilidade & Elasticidade

Sistemas [de tecnologia]

Escalabilidade é uma característica de uma organização, sistema, modelo ou função que descreve sua capacidade de lidar e ter um bom desempenho sob uma carga de trabalho ou escopo maior ou em expansão. No contexto de MBAs, para escalar com sucesso, cada microserviço individual precisa escalar individualmente e como parte de um sistema maior.

🔗 Referências

Wiki | Escalabilidade

<https://pt.wikipedia.org/wiki/Escalabilidade>

ScienceDirect | Scalability

<https://www.sciencedirect.com/topics/computer-science/scalability>

How to break a Monolith into Microservices

<https://martinfowler.com/articles/break-monolith-into-microservices.html>

The Hardest Part About Microservices: Your Data

<https://blog.christianposta.com/microservices/the-hardest-part-about-microservices-data/>

The challenges of scaling microservices

<https://techbeacon.com/app-dev-testing/challenges-scaling-microservices>

🔗 Links compartilhados

 **Emerson Victor**

API Gateway

<https://microservices.io/patterns/apigateway.html>

 **Ramom Pereira dos Santos Silva**

AWS API Gateway

<https://aws.amazon.com/api-gateway/>

 **Ramom Pereira dos Santos Silva**

Elasticity and Resilience

<https://fabric8.io/guide/develop/elasticity.html>

 **Talyta Maria Rosas Pacheco**

Scalability Versus Elasticity: What's the Difference, and Why Does It Matter?

<https://www.parkplacetechologies.com/pt-br/knowledge-center/blog/data-center/scalability-versus-elasticity-whats-difference-matter/>

 **Arthur Frade de Araújo**

Elastic Load Balancing AWS

https://aws.amazon.com/elasticloadbalancing/?sc_channel=PS&sc_campaign=acquisition_BR&sc_publisher=google&sc_medium=english_load_balancing_b&sc_content=aws_load_balancer_e&sc_data=dkW41EG8EGkaAlrCEALw_wcB:G:s&s_kwcid=AL!4422!3!159751609434!e!!g!!load%20balancing%20aws

 **victor sena de lima attar**

DIFFERENCES BETWEEN ELASTICITY & SCALABILITY IN CLOUD COMPUTING

<https://theappsolutions.com/blog/cloud/cloud-computing-elasticity-vs-scalability/>

 **Roberto Oliveira**

DESEMPENHO E ESCALABILIDADE

<https://repositorio.ufu.br/bitstream/123456789/12488/1/ivens.pdf>

Qual a relação (diferença, similaridade) entre escalabilidade e desempenho?

EV Emerson Victor 17/09/2020

Um sistema escalável é aquele que tem a capacidade de lidar com o aumento de carga sem impacto no desempenho, fazendo com que os recursos disponíveis possam ser aumentados rapidamente

RM Rafael Mota Alves 18/09/2020

O desempenho fala sobre a capacidade dos sistemas de software de oferecer seus recursos de forma rápida, e a escalabilidade é essa capacidade desse sistema de "aguentar" um aumento de uso do mesmo, mantendo tanto a confiabilidade quanto o desempenho

VG Vinicius Garcia 23/09/2020

qual a definição de elasticidade, na física?

RE Ricardo Ebbers Carneiro Leão 19/09/2020

Um sistema escalável é capaz de aumentar sua própria capacidade em receber requisições de acordo com a demanda, mas não necessariamente um sistema escalável tem alto desempenho. No caso de um sistema de renderização 3D distribuído, por exemplo, onde a principal funcionalidade demanda muitos recursos de processador, ser capaz de atender 10 ou 1000 usuários não reduz o tempo de processamento individual. Escalabilidade reduz filas de espera, desempenho reduz o tempo de processamento.

AF Arthur Fraude de Araújo 19/09/2020

Apesar de ambas as características impactarem positivamente o funcionamento de uma aplicação, escalabilidade e desempenho diferem em conceito. Escalabilidade se refere à capacidade de um sistema conseguir manter-se disponível e funcional independentemente do nível de demanda (quantidade de clientes consumindo a aplicação). Já o desempenho diz respeito a quão performático o sistema é ao executar as tarefas a que se propõe (tempo de resposta, complexidade de algoritmos e etc).

AF Arthur Fraude de Araújo 19/09/2020

Porém, mesmo diferentes, ambas as características se relacionam de alguma maneira. Sistemas de alto desempenho geralmente são naturalmente mais escaláveis do que sistemas ineficientes.

RP Ramom Pereira dos Santos Silva 19/09/2020

Ambos os conceitos se relacionam quando falamos de aplicações de alta performance, porém são diferentes conceitualmente. Desempenho diz respeito a performance do sistema em executar suas funcionalidades, como tempo de resposta e tempo de execução de alguma rotina. Já escalabilidade é a capacidade de um sistema suportar aumento da carga de trabalho de forma uniforme, ou seja, mantém-se disponível e em execução, sendo transparente o aumento da capacidade computacional alocada.

LB Luan Brito 20/09/2020

Como dito pelos colegas, a ideia da escalabilidade é a capacidade de manter um desempenho constante apesar de um aumento de carga ou escopo do sistema.

SG Saulo Guilhermino 20/09/2020

Escalabilidade relaciona-se com a capacidade de um sistema de flexibilizar recursos de maneira quantitativa, à medida que sua demanda varia para mais ou para menos. Desempenho também se relaciona com a oferta de recursos, mas com o olhar mais qualitativo, ou seja, medidas como tempo de resposta e de execução são levados em conta para avaliar o quão boa a oferta de recurso está sendo.

HC Heitor Carvalho 20/09/2020

Desempenho pode ser entendido como métricas relevantes para medir se o sistema é eficiente no que se propõe a fazer. Por exemplo, um sistema de processamento de arquivos tem um bom desempenho se for capaz de processar grandes quantidades de arquivos eficientemente. Já a escalabilidade é uma característica do sistema que diz respeito a capacidade do sistema de aumentar seu desempenho quando são acrescidas a ele mais recursos: espaço, capacidade de processamento, memória, etc.

LB Lucas Barros 21/09/2020

Desempenho é a capacidade de um sistema de prover uma resposta em determinado intervalo de tempo. Escalabilidade auxilia isso, já que faz com que altas cargas de trabalho não deixem o sistema fora do ar ou demore para executar sua ação, mantendo o desempenho da aplicação sem influências desses outros fatores.

LC Lucas Cardoso 21/09/2020

Desempenho em termos de software, geralmente se define em quão bem um software se adequa ao seu requisito temporal. Escalabilidade, se refere ao quanto o sistema pode alocar novos recursos, para uma crescente carga de trabalho mantendo o desempenho em níveis aceitáveis

 **Lucas Cardoso** 21/09/2020

Então desempenho pode ser interpretada como a qualidade percebida de um sistema ou serviço e escalabilidade é uma propriedade que pode ajudar a atingir isso

 **Talyta Maria Rosas Pacheco** 22/09/2020

Desempenho se enquadra como um padrão de qualidade de um software, muito relacionado ao tempo para realizar uma ou mais atividades, tempo de resposta, vazão etc. Já escalabilidade está relacionado com a capacidade de manter desempenho quando sujeito à grandes cargas de requisições.

 **Gabriel Cavalcanti de Melo** 22/09/2020

Escalabilidade é a capacidade de aumentar a carga de trabalho sem diminuir o desempenho, que por sua vez, é a relação entre a capacidade de funcionamento com o tempo e a utilização de recursos.

 **Marcos Galvão** 23/09/2020

O Desempenho é algo associado, principalmente, a como um sistema responde a uma dada circunstância, podendo ser o tempo da resposta, o quanto de recurso, a capacidade de usuários que ele oferece suporte simultâneo, etc. A escalabilidade é uma característica que permite ao sistema se adaptar a maiores demandas, permitindo-o ser flexível e aumentar sua capacidade, sendo assim pode-se enxergar a escalabilidade como algo que permite o desempenho escalar mesmo sob demandas maiores.

 **Danilo Lira** 23/09/2020

A escalabilidade está bastante atrelada ao desempenho, pois como Gabriel citou, ela é a capacidade de aumentar carga sem reduzir o desempenho. A primeira característica visa manter a qualidade de um serviço o tornando mais potente de acordo com a necessidade, já a segunda é o quão bem um serviço pode trabalhar, sendo assim vemos que eles estão bastante ligados, pois um possibilita o outro, mas são conceitos diferentes.

Mateus Nunes 23/09/2020

-Escalabilidade mostra o potencial de crescimento do software sem necessitar e expansão da equipe e de complexidade do software -Desempenho é associado a qualidade do software, está ligado ao tempo de resposta das ações que são requisitadas. Os dois podem ter choques, se criar-se um software com ótimo desempenho sem pensar na escala pode ser um risco para o software em geral. A escalabilidade tende a ser um agravador de desempenho com software mal projetados

 **victor sena de lima attar** 23/09/2020

Desempenho é associado às métricas de qualidade do software, como tempo de resposta, escalabilidade, e utilização de processamento. Escalabilidade é uma métrica que define se o sistema pode crescer de forma sustentável devido a demanda enfrentada.

 **José Sheldon Brito Fekete** 23/09/2020

A escalabilidade e o desempenho estão atrelados, enquanto o desempenho é um indicador de qualidade do software, tais como tempo de resposta e disponibilidade do sistema. Já a escalabilidade é a capacidade de se manipular uma quantidade crescente de trabalho sem perder o desempenho ou afeta-lo minimamente.

 **Izabella Nascimento** 23/09/2020

O desempenho se baseia na capacidade de resposta de um sistema para a sua execução, já a escalabilidade é a capacidade de um sistema lidar com o crescimento sem ter impacto no desempenho.

 **Roberto Oliveira** 23/09/2020

A escalabilidade se tornou uma importante propriedade em seu projeto e arquitetura, fazendo com que tenham que lidar com cargas de trabalho, de dados e de acessos cada vez maiores enquanto é exigida um desempenho satifatório, ou seja os dois estão bastante relacionados. O desempenho é a quantidade de trabalho realizado por um sistema comparado ao tempo e recursos utilizados.

O que é escalabilidade no contexto de MBAs e como eu faço para aplicá-la com sucesso? Cite 3 boas práticas e 3 riscos/obstáculos mais representativos na sua opinião.

 **Emerson Victor** 17/09/2020

Algumas boas práticas são deploys automatizados, criação de um gateway e monitoramento através de logs. Já riscos/obstáculos que podem ser encontrados são integração de dados, necessidade de identificar quais elementos individuais precisam ser escalonados para atender a demanda e o deploy em diferentes locais da nuvem.

 **Emerson Victor** 17/09/2020

Algumas boas práticas são deploys automatizados, criação de gateway e monitoramento através de logs. Já riscos/obstáculos que podem ser encontrados são integração de

dados, necessidade de identificar quais elementos individuais precisam ser escalonados para atender a demanda e o deploy em diferentes locais da nuvem

 VG Vinicius Garcia 23/09/2020

deploy automatizados + elasticidade... não consegui seguir o raciocínio

 RE Ricardo Ebbers Carneiro Leão 19/09/2020

IaaS, OSS e Infraestrutura como Código ajudam bastante na criação de produtos naturalmente escaláveis. A forma mais simples é ter um load balancer e serviços que consigam se registrar nele automaticamente. Quando a taxa de requisições estiver uma certa % da capacidade máxima (80%, por exemplo), basta levantar mais um container do mesmo serviço e deixar que ele se registre e o load balancer passe a enviar requisições para ele.

 RE Ricardo Ebbers Carneiro Leão 19/09/2020

Formas mais sofisticadas de criar infraestrutura escalável envolve orquestração de containers com o Docker Swarm, Kubernetes, Terraform, etc. Uma excelente prática para escalabilidade é também inserir o conceito de elasticidade - reduzir a quantidade de instâncias quando a alta demanda acaba. Também é importante garantir que os múltiplos serviços não estão disputando os mesmos recursos de IO, memória ou CPU e que os serviços sejam idempotentes.

 RE Ricardo Ebbers Carneiro Leão 19/09/2020

O risco mais comum e difícil de detectar cedo é o processamento duplicado de um mesmo evento por serviços em paralelo. Outro risco é ter escalabilidade mas não elasticidade, deixando instâncias ociosas após as demandas acabarem. Por fim, um serviço escalável pode gerar pressão em serviços que não são, causando indisponibilidade.

 RP Ramom Pereira dos Santos Silva 19/09/2020

Escalabilidade no contexto de MBA's é mais complexa, dado que cada microserviço pode escalar de forma independente, conforme a demanda, além de poderem executar em zonas de disponibilidade diferentes ou até mesmo regiões (diferentes continentes). Boas práticas de implementação de escalabilidade seria o uso de Monitoramento, ferramentas de orquestração de contamines (Kubernetes, por exemplo) e Infraestrutura como código (Terraform e CloudFormation, por exemplo), muito útil no uso de Shards,

 RP Ramom Pereira dos Santos Silva 19/09/2020

para replicação completa de uma infraestrutura específica. Porém alguns riscos são o custo de cloud conforme a aplicação vai escalando, além disso, entender quando escalar também é muito importante, ou seja, quais métricas utilizar para definir o momento de escala, e também manter elasticidade, dado que existe a possibilidade de recursos computacionais não serem usados, o que impacta nos custos também.

 AF Arthur Frade de Araújo 19/09/2020

No contexto das MBAs a definição de escalabilidade se mantém, o que muda é a forma como ela será implementada. E sobre isso, existem algumas práticas já validadas por muitos negócios, que permitem que esse desafio seja cumprido com sucesso: 1 - Load-Balancing. O fluxo de requisições ao serviço será direcionado a um componente que irá distribuir o tráfego por diferentes servidores.

 AF Arthur Frade de Araújo 19/09/2020

2 - Containers. Tecnologias que permitem a criação e o gerenciamento de máquinas virtuais autocontidas. Cada container, como são chamados, possui seu próprio sistema operacional e suas próprias aplicações. Isso permite gerenciar os serviços de maneira mais desacoplada dentro de um mesmo servidor. Geralmente, plataformas como o Docker, permitem replicar, desligar e criar novos containers com muita simplicidade. Comandos prontos ajudam a promover escalabilidade de maneira simples.

 AF Arthur Frade de Araújo 22/09/2020

3 - Orquestração de containers. Os containers por si só não são automaticamente escalados e gerenciados. É necessário implementar um fluxo de automatizações para fazer um uso mais eficiente dessa tecnologia. Através desses processos é possível replicar e reduzir containers conforme a demanda, bem como integrá-los a ferramentas de CI/CD como pipelines de implantação (github actions, git lab pipeline e outros). Além disso, ferramentas de orquestração permitem manipular vários containers de uma vez

 AF Arthur Frade de Araújo 22/09/2020

Obstáculos: Entendimento de ferramentas complexas e capacitação de equipes. Implementar esse tipo de solução do zero exige MUITO esforço. Nesses casos é bom ter conhecimento de plataformas de Cloud Computing, onde esse tipo de prática é mais facilmente implementada. Outro desafio é o gerenciamento e manutenção dessa infraestrutura complexa e automatizada

 LB Luan Brito 20/09/2020

No contexto de MBA, é importante perceber quais serviços do sistema tem (ou podem vir a ter) uma carga grande, e quais não. 3 boas práticas -> 1) gerenciar carga de serviços. monitorando cargas e computando métricas de desempenho. 2) se surgir sobrecarga, procurar entender bem sua origem, para atacar o problema de forma correta. 3) Utilizar técnicas de replicação de instâncias de serviços com muita demanda e load balance



 LB Luan Brito 20/09/2020

riscos: 1) má implementação de protocolos de comunicação entre serviços, podendo levar à bottlenecks. 2) aumento da preocupação com as dependências, já que diferentes serviços podem ter dependências, aumentando a complexidade para garantir escalabilidade/eficiência. 3) dificuldade de manter consistência de informação, devido a separação entre as peças e replicação de instâncias.

 Heitor Carvalho 20/09/2020

A escalabilidade no contexto de MBA pode ser vista como a capacidade de adaptação de serviços específicos de alta demanda para continuarem a fornecerem suas funcionalidades sem penalidades na performance, como por exemplo, lentidão ou travamento. Para aplicá-la com sucesso é necessário entender quais serviços necessitam de serem escalados e aplicar técnicas como balanceamento de carga, API gateways e replicação automática de instâncias (docker swarm por exemplo) para garantir disponibilidade.

 Heitor Carvalho 20/09/2020

Entre os riscos/obstáculos mais representativos estariam o de garantir a integridade dos dados (gerenciar a informação compartilhada entre as múltiplas instâncias replicadas de um serviço, quem lê/escreve primeiro no banco, etc), lidar com os custos envolvidos com escalabilidade de aplicações (uma aplicação escalável mal planejada pode "escalar demais" e gerar grandes custos na manutenção dos servidores) e a automatização da pipeline deste processo, que pode ser complexa no início.

 Lucas Barros 21/09/2020

Escalabilidade no contexto de MBA é interessante pois se torna mais fácil de escalar certas partes do sistema que são mais afetadas por grandes cargas de trabalho, ao invés do sistema todo. Entretanto, torna-se mais complexo de gerenciar, já que é preciso dar atenção a diversos serviços quanto a esse ponto. ... Boas práticas/riscos nos comentários

 Lucas Barros 21/09/2020

Boas práticas seria 1) Utilizar orquestradores de containers (Kubernetes, Docker Swarm) pois eles fornecem funcionalidades de escala horizontal e vertical, sendo possível configurar de acordo com o serviço. 2) Manter um bom monitoramento dos serviços, para que seja fácil identificar onde há aumento de carga e agir de acordo 3) Utilizar-se de ferramentas para comunicação assíncrona entre serviços (Kafka, RabbitMQ) para que minimize o número de altas cargas de trabalho em certos serviços.

 Lucas Barros 21/09/2020

Riscos/Obstáculos seriam 1) Lidar com o custo (\$) associado às escalas horizontais e verticais. 2) Estudo e configurações das ferramentas corretas para automatização desse processo 3) Reunir informações e entender qual serviço escalar, em que modo (horizontal, vertical), e em que fator (mínimo, máximo de escala).

 Rafael Mota Alves 21/09/2020

Quanto aos obstáculos, temos a dificuldade de gerenciar a quantidade maior de aplicações inerente a MBAs, outra dificuldade é não tornar os sistemas que guardam estado (como bancos de dados, caches e sistemas de mensageria) um gargalo para aplicação, e também gerenciar a comunicação entre os serviços nesse contexto, evitando que microserviços sobrecarreguem uns aos outros e também evitando sobreregar o sistemas de comunicação, podendo causar uma maior latência na comunicação entre os

 Rafael Mota Alves 21/09/2020

serviços nesse contexto, evitando que microserviços sobrecarreguem uns aos outros e também evitando sobreregar o sistemas de comunicação, podendo causar uma maior latência na comunicação entre os serviços. Quanto as boas práticas para evitar evitar/contornar esses obstáculos seriam, respectivamente: usar ferramentas de gerenciamento de serviços como: k8s, service mesh, ecs..., evitar fazer manter estado nas aplicações, e quando necessário utilizar ferramentas facilmente escaláveis como:

 Rafael Mota Alves 21/09/2020

redis, mongodb, dynamodb... e tentar usar ao máximo comunicação assíncrona entre os serviços usando ferramentas como: kafka, rabbitmq e SQS.

 Danilo Lira 23/09/2020

A escalabilidade no contexto de microserviços tem o mesmo objetivo que a abordagem tradicional que é adaptar a capacidade da aplicação de acordo com a necessidade, porém nesse contexto apenas os serviços da aplicação que estão sendo mais requisitados precisam ser adaptados, sendo essa uma forma de aplicá-la com sucesso. Pensando nas boas práticas temos o uso do balanceamento de carga, telemetria para checar a saúde dos serviços e utilização de contêineres para cada serviço.

 Danilo Lira 23/09/2020

Vejo que as maiores dificuldades são relacionadas a quais serviços escalar e quando. Entender quando há a necessidade de escalar um serviço e fornecer todo o desempenho que ele necessita pode se tornar bastante custoso.

 Marcos Galvão 23/09/2020

No contexto de MBA escalabilidade tem como principal conceito garantir que a capacidade do sistema atender suas demandas cresça conforme a API sofre de mais requisições, não permitindo que ela deixe de oferecer seus serviços por falta de recursos.

 **Marcos Galvão** 23/09/2020

Três boas práticas são: 1- Rastrear com exatidão quais são os serviços que precisam ser escalonados para, caso necessário, não gastar recursos em lugares desnecessários. 2- Utilizar um load-balance para distribuir as requisições através das várias instâncias dos serviços que estão sendo escalonados. 3- Utilizar containers para permitir de maneira facilitada a replicação de novas instâncias dos serviços.

 **Marcos Galvão** 23/09/2020

Três riscos: 1- Caso a análise de quais serviços precisam ser escalonados seja má feita recursos serão desperdiçados nos serviços, gerando gastos desnecessários. 2- Caso não seja utilizada uma boa política de balanceamento, pode ocorrer de mesmo ampliando as instâncias dos serviços a distribuição das requisições entre elas ainda causará gargalo. 3- Conseguir gerir os gastos adicionais providos pelo escalonamento, pois é muito mais difícil prever quanto será gasto em recursos.

Mateus Nunes 23/09/2020

Escalabilidade na arquitetura MBA se torna um pouco mais complicada para implementação, porém extremamente eficiente. Como cada microserviço opera de maneira independente, estes devem escalar e ser elásticos de maneira independente conforme as demandas do usuário aumentam em determinadas partes do software atendendo também demandas em diferentes localizações do mundo. Boas práticas seriam Estruturação de Containers, Load Balance, Monitoramento.

Mateus Nunes 23/09/2020

O maior problema é o custo financeiro e de desenvolvimento da arquitetura, como é algo extremamente sofisticado pode não ser a melhor opção para um teste sujo e validação inicial, mas indispensável para construção de soluções sólidas mas a empresa tem que ter em mente que ferramentas podem ser custosas como kubernetes nesta forma de implementação

VS

victor sena de lima attar 23/09/2020

Para MBA se usa a escalabilidade horizontal, criando novos serviços para diminuir o peso de apenas uma máquina rodando todos os serviços, utilizando técnicas de docker e kubernetes como orquestrador desses containers, que vigia os nós do container para verificar se algum erro aconteceu para que possa reiniciar o nó ou verifica se necessita de mais nós, criando mais nós para aguentar a carga de uso.

VS

victor sena de lima attar 23/09/2020

Como pontos negativos, temos a complexidade para tratar esse pontos necessitando de gente especializada, sendo difícil para aplicação de MVP's, mas sendo essencial para grandes aplicações com vários sistemas complexos.

JS

José Sheldon Brito Fekete 23/09/2020

A escalabilidade em MBA tem como objetivo garantir a capacidade do sistema de suportar uma maior quantidade de trabalho sem diminuir o desempenho, e no cenário de MBA. Como em MBA se trabalha com serviços independentes se torna possível aprimorar e expandir um componente sem modificar todo o sistema, acelerando assim atualizações no sistema e poupança de tempo e trabalho.

JS

José Sheldon Brito Fekete 23/09/2020

Acredito que algumas boas práticas seriam: Ter um bom planejamento de quais os serviços que mais necessitam de escalabilidade e constante monitoramento de requisição dos mesmos, utilizar平衡adores de carga, tão como um API gateway por exemplo e o uso de containers, como exemplo o Docker.

Na maioria das implementações de microsserviços, os endpoints internos não são expostos externamente, eles são mantidos como serviços [privados]. Para expor um conjunto de serviços públicos, uma solução é utilizar um padrão chamado API Gateway. No que consiste este padrão e quando eu devo utilizá-lo? Dê exemplo de um cenário.

EV

Emerson Victor 17/09/2020

API gateway é uma ferramenta de gerenciar APIs. Ele intercepta as requisições, agrupa todos os serviços necessários para responder e, em seguida, retorna o resultado. Algumas funções comuns incluem autenticação, roteamento, limitação de taxa, faturamento, monitoramento, análise, políticas, alertas e segurança. Essa ferramenta lida com solicitações de duas maneiras, algumas são simplesmente encaminhadas para o serviço apropriado, outras agregam informações de diversos serviços.

EV

Emerson Victor 17/09/2020

Um exemplo de uso é a Netflix, que em vez de fornecer um única API com todas suas informações, utiliza um gateway para fornecer para cada cliente uma API que atenda apenas as suas necessidades.

RE

Ricardo Ebbers Carneiro Leão 19/09/2020

Uma ótima forma de usar um api gateway é agregar cross-cutting concerns num só lugar - como autenticação e autorização, segurança contra ataques, pode ser uma forma de prover

uma "falha rápida" sem causar efeitos colaterais a partir de circuit breaking, pode permitir escalabilidade como um load balancer. Um caso de uso prático é injetar um header de identificação do usuário logado no header de forma que os microserviços internos que dependem dessa info não precisem consultar um serviço a parte.

RP **Ramom Pereira dos Santos Silva** 19/09/2020

API Gateway é um padrão utilizado para gerenciar API's, unificando diversas em um único ponto de entrada para a aplicação, muito útil em aplicações MBA's, visto que proporciona, aos serviços consumidores da aplicação, transparência da API. Como benefícios, fácil monitoramento do uso, autenticação e roteamento de requisições, escalabilidade das requisições e flexibilidade ao adicionar ou remover novas API's (microserviços) e isso ser transparente para os clients.

RP **Ramom Pereira dos Santos Silva** 19/09/2020

Como exemplo prático temos a Netflix, que construiu sua própria ferramenta (pertencente a Stack Netflix OSS), construindo diversas camadas de API Gateway, possibilitando também versões diferentes de sua aplicação para clientes distintos. Atualmente as plataformas de Cloud (AWS, GCP, Azure, etc) construíram suas próprias ferramentas.

LB **Luan Brito** 20/09/2020

API Gateway deve ser usada para abstração de recursos e redução da complexidade de comunicação e utilização de serviços, centralizando num único canal de comunicação genérico.

SG **Saulo Guilhermino** 20/09/2020

API Gateway trata-se de um único ponto de acesso aos recursos de diversas APIs, atua no redirecionamento do tráfego e também filtragem. Pode ser visto como uma API para acessar APIs. Como citado anteriormente, um ótimo exemplo é o da Netflix, que fornece para cada cliente a API que melhor atende suas necessidades através de um API Gateway

HC **Heitor Carvalho** 20/09/2020

Um API GATEWAY pode ser visto como um "agregador" de APIs, que mantém registro de todos os serviços ao qual ele pode fornecer acesso, e, através de uma única requisição a este gateway, o cliente pode obter de forma transparente os dados que precisa sem se preocupar em qual API deve consultar. A resposta de vários serviços pode ser aglutinada em uma única forma a não só reduzir o numero de requisições mas também a balancear a carga entre os multiplos serviços gerenciados pelo gateway.

HC **Heitor Carvalho** 20/09/2020

Um possível cenário onde o uso de um API GATEWAY se encaixaria seria o seguinte: Uma aplicação de gerenciamento de locação de carros para uma locadora, onde cada loja dessa locadora (espalhada em várias cidades do país) possui seu próprio microsserviço e banco de dados correspondente. Um API Gateway faria com que a consulta aos automóveis disponíveis pudesse ser feita em uma única requisição, mas internamente, consultando e agregando todas as APIs dos vários microsserviços.

LB **Lucas Barros** 21/09/2020

Deve ser utilizado quando clientes externos querem acessar várias APIs que são providas por diferentes serviços. O API Gateway serve como uma "cola" para essas APIs: todas elas podem ser acessadas por um domínio único, além de ter a possibilidade de haver um pipeline associado (autenticação, monitoramento, gerenciamento de tráfico, transformação de requests e response, etc). Um exemplo de uso seria um website que, para renderizar uma página, precisa requisitar 2 ou mais APIs (por utilizar MBAs).

RM **Rafael Mota Alves** 21/09/2020

O API Gateway, é um padrão em que uma aplicação fica na "frente", dos demais serviços da sua aplicação, gerenciando o acesso aos recursos dele. Um API Gateway tem várias aplicações, mas em geral, usamos ele quando queremos centralizar e um único lugar o controle de acesso aos recursos da aplicação. Um cenário que API Gateway poderia ser usado, seria pra agrregar as rotas de recursos de uma aplicação, controlando que tem acesso a elas (autenticação e autorização) e fazendo também o controle de

RM **Rafael Mota Alves** 21/09/2020

carga entre instâncias dos mesmos serviços.

LC **Lucas Cardoso** 21/09/2020

Como explicado pelos meus colegas, o API Gateway serve como um hub que disponibiliza as APIs de um sistema, que podem estar divididas em diversos serviços. Um exemplo de uso, seria para fornecer as APIs públicas de um sistema composto de diversos microsserviços

AF **Arthur Frade de Araújo** 22/09/2020

Uma API Gateway é a aplicação responsável por receber as requisições dos clientes e roteá-las para os microsserviços. Esse tipo de arquitetura permite que os clientes acessem qualquer serviço do ecossistema através de um único ponto central. Essa estratégia geralmente é útil na execução de tarefas comuns a vários endpoints, como autorização e segurança. Uma aplicação bancária por exemplo, que possua serviços para execução de transações, geração de relatórios e cadastro de clientes...

 Arthur Frade de Araújo 22/09/2020

Pode utilizar uma API gateway para que um aplicativo (banco digital) consiga acessar os serviços através de um mesmo domínio (dns ou ip) e porta.

 Danilo Lira 23/09/2020

O API Gateway é um padrão onde há um serviço exposto que é utilizado como mediador entre chamadas externas e os serviços de uma aplicação. Neste padrão o usuário faz chamada para o API Gateway e ele por sua vez chama o serviço requisitado. Esse tipo de padrão deve ser utilizado com microserviços para protegê-los. Uma aplicação pode utilizar esse padrão para rastrear o tráfego dos usuários e criar registros sobre os serviços mais utilizados

 Marcos Galvão 23/09/2020

O API gateway funciona como um centralizador para as requisições, oferecendo para os clientes um único portão, que será chamado, e que dará conta de rotear a requisição para os demais serviços da aplicação. O gateway pode prover uma série de funcionalidades, por exemplo: validação, balanceamento de chamadas para instâncias de seus serviços, uma maior proteção pois seus serviços não são diretamente acessíveis, etc.

 Marcos Galvão 23/09/2020

um exemplo de uso seria um MBA que contem diversos microserviços compondo a aplicação, assim sendo, quando um cliente acessar o gateway, internamente os devidos/necessários serviços serão acessados e a informação requisitada será retornada pelo gateway, fazendo assim o cliente não enxergar ou preocupar-se com a distribuição interna dos microserviços na aplicação.

Mateus Nunes 23/09/2020

API Gateway é uma aplicação que agrupa os vários serviços necessários para realizá-las e retorna o resultado apropriado. Isso é importante nas aplicações MBA pois da um ponto central para acesso ao ecossistema de serviços construídos. Os benefícios são autorização, escalabilidade e flexibilidade.

Mateus Nunes 23/09/2020

Um exemplo seria uma empresa com diversos centros de estoque e que cada centro (os chamados CD) teriam seu banco de dados e serviço rodando, o sistema de central de estoque que se conectaría com estes e com uma única requisição poderia fazer as demais requisições privadas necessárias para conseguir o dado do estoque.

 victor sena de lima attar 23/09/2020

Api gateway é uma aplicação que cria uma barreira para os serviços da organização, pois os acessos externos chegam primeiro no api gateway, que poderá fazer validações na chamada antes de ser repassada para o serviço. Aplicações que usam MBA tem serviços que se comunicam internamente e outros que são chamados pela aplicação externamente, o api gateway vai barrar a comunicação externa com esses serviços que apenas deveriam se comunicar entre si.

 José Sheldon Brito Fekete 23/09/2020

A utilização do API Gateway nessa situação é utilizado como centralizador de serviços de uma aplicação MBA, recebendo solicitações de clientes e encaminhando para os devidos serviços do sistema, servindo também como um encapsulador para o cliente que está utilizando a API.

 José Sheldon Brito Fekete 23/09/2020

Um exemplo seria uma empresa que utiliza o estilo arquitetural de microserviços, possuindo diversos processos executando independentemente, nessa situação ele utiliza o API Gateway para fazer a centralização de chamadas de usuários que estão acessando seu sistema e encaminha para os devidos processos, assim devolvendo a resposta, ele serve como encapsulador pois o usuário não tem a menor ideia de quais ou quantos processos passou a sua requisição.

Qual a relação entre escalabilidade e elasticidade e como podemos caracterizar a elasticidade no contexto de MBAs?

 Ricardo Ebbers Carneiro Leão 19/09/2020

Escalabilidade é a capacidade de um serviço de aumentar sua própria capacidade dinamicamente frente a aumentos de demanda. Elasticidade é, além de conseguir reservar mais recursos quando a demanda cresce, ter a capacidade de liberar os recursos reservados para ele quando não são mais necessários.

 Ramom Pereira dos Santos Silva 19/09/2020

Conforme dito anteriormente, escalabilidade é a capacidade de uma aplicação crescer os recursos alocados dinamicamente quando necessário. Elasticidade é a capacidade de alocar os recursos quando necessário, mas também liberar esses recursos quando não estiverem sendo utilizados, ou seja, quando os recursos são aumentados e diminuídos, conforme a demanda.

  Ramom Pereira dos Santos Silva 19/09/2020

No contexto de MBA's, como lidamos com diversos serviços independentes, é necessário olhar individualmente para cada um, pois podem escalar de forma independente. Quando falamos de aplicações com alto desempenho, sempre é necessário levar em consideração o custo atrelado a isso. Falando sobre elasticidade, é necessário analisar se o microserviço é stateless ou não.

 **Ramom Pereira dos Santos Silva** 19/09/2020

Stateless: como não guarda estado, é tranquilo de escalar, e diminuir a capacidade alocada, posteriormente. Tecnologias Serveless, como AWS lambda (funções orientadas a eventos) e Cloud Run (containers sem estado) podem ser bem aproveitadas nesses contextos. Stateful: Nesse caso, já é mais complicado, dado que pode ser necessário replicar dados e informações entre os novos nós quando escalar. Kubernetes pode muito bem ser utilizados nesses casos, utilizando Replica Sets, que possibilita

 **Ramom Pereira dos Santos Silva** 19/09/2020

realizar o autoscaling, setando algum target (CPU ou Memória RAM) de utilização, o que possibilita o K8S realizar, de forma automática, escalabilidade e elasticidade, além de Service Discovery.

 **Luan Brito** 20/09/2020

Elasticidade é o passo seguinte depois da escalabilidade, uma aplicação deve não só crescer deliberadamente, mas poupar recursos quando podem ser necessários para outras aplicações.

 **Saulo Guilhermino** 20/09/2020

Uma aplicação escalável é capaz de oferecer mais recursos a medida que o projeto necessite atender mais demandas de forma planejada. Uma aplicação elástica é capaz de, dinamicamente, aumentar ou diminuir a oferta de recursos a medida que a demanda aumente no decorrer das horas de um dia, por exemplo.

 **Saulo Guilhermino** 20/09/2020

No contexto de MBA, podemos caracterizar a elasticidade como a capacidade de um serviço ser "reativo", ou seja, se auto-replicar para que consiga fornecer recursos suficientes para as demandas à medida que novas demandas chegam.

 **Heitor Carvalho** 20/09/2020

A elasticidade de uma aplicação está ligada a sua escalabilidade no sentido de que ela deve ser adaptável, não somente crescendo/alcancando recursos e replicando suas instâncias quando necessário, mas também contraíndo-se, eliminando suas cópias e poupando recursos em momentos onde eles não estão sendo constantemente utilizados.

 **Lucas Barros** 21/09/2020

Após a escala, é necessário dar atenção para quando voltar o sistema para o estado original para assim poupar recursos computacionais. É aí que a elasticidade entra: é a capacidade de um sistema de adaptar-se -> escalando quando há aumento de carga e ir diminuindo ao longo que vai ficando mais tranquilo. No contexto de MBA essa característica se torna crucial, já que existem vários serviços a serem escalados e que recebem diversas cargas de trabalho diferentes durante o dia.

 **Rafael Mota Alves** 21/09/2020

A elasticidade é a capacidade de um sistema de aumentar a sua capacidade de receber carga (escalar) e também de diminuir sua capacidade, de acordo com a necessidade, portanto a elasticidade é um dos fatores necessários para um sistema ser "escalável". No contexto de MBAs, a elasticidade é facilitada, pois nesse contexto as partes da aplicação podem ser escaladas independentemente.

 **Lucas Cardoso** 22/09/2020

Como explicado, a elasticidade se dá na capacidade do sistema de gerenciar seus recursos se adaptando a carga demandada, tanto aumentando quanto diminuindo os recursos. No contexto de MBA que geralmente utilizam de funcionalidades de cloud, é de extrema importância, pois está diretamente ligado ao custo de uso dos serviços.

 **Lucas Cardoso** 22/09/2020

Além disso, ela geralmente se dá na capacidade que um sistema tem de criar e também matar instâncias novas, para maximizando o uso de recurso e mantendo a qualidade

 **Talyta Maria Rosas Pacheco** 22/09/2020

A principal diferença entre esses dois atributos de um data center tem a ver com o planejamento. Escalabilidade é a capacidade do sistema atender a diversos tipos de demandas/requisições de forma planejada. O aumento de recursos na infraestrutura é planejado de acordo com previsões de uso do sistema.

 **Talyta Maria Rosas Pacheco** 22/09/2020

Já Elasticidade é a capacidade do sistema atender a um pico de requisições de forma mais "live". Ele próprio se ajusta (aumenta/diminui) mediante as necessidades, como acontece na Black Friday, já que não tem uma previsão sobre a quantidade de requisições. No contexto de microserviços, elasticidade caracteriza-se por os serviços se tornarem elásticos individualmente



GC

Gabriel Cavalcanti de Melo 22/09/2020

A escalabilidade é aumentar capacidade de forma projetada. Já a elasticidade é a capacidade de aumentar ou diminuir a infraestrutura do sistemas de forma automática e dinâmica de acordo com a quantidade de acessos. A Elasticidade é importante para a economia, justamente pelo fato de que ela também é capaz de desacoplar recursos desnecessários, quando os acessos estão baixos.

Gabriel Cavalcanti de Melo 22/09/2020

Em MBAs, elasticidade tem a mesma característica. Seria a capacidade de resposta e o dimensionamento automático dos serviços individualmente afetados.

AF

Arthur Frade de Araújo 22/09/2020

Na minha opinião elasticidade é um tipo mais rebuscado de escalabilidade. Um sistema pode ser extremamente escalável, por exemplo: Imagine que uma empresa desenvolve um sistema que através de um load balancer tem seu tráfego distribuído em 10 instâncias da aplicação. Isso seria bastante escalável, já que 10 instâncias conseguiriam suportar um grande número de requisições. Porém, imagine que em um dado momento a demanda do sistema não é alta o suficiente para necessitar dessa estrutura...

Arthur Frade de Araújo 22/09/2020

A empresa nesse caso possuiria um sistema bastante escalável, mas que irá cobrar bastante da empresa (muitas instâncias rodando) sem que ela necessite de todas a todo o momento. É aí que entra o diferencial da elasticidade. Elasticidade é um tipo de escalabilidade auto regulada...

Arthur Frade de Araújo 22/09/2020

Quando a demanda pelos recursos aumenta, uma infraestrutura elástica automaticamente iria replicar instâncias da aplicação em picos de utilização (spikes) e reduzir o número de instâncias em momentos de baixa demanda.

Arthur Frade de Araújo 22/09/2020

Esse tipo de tecnologia permite a otimização da utilização de recursos de hardware e claro, dos custos relacionados a hospedagem de software.

Arthur Frade de Araújo 22/09/2020

Um exemplo é o serviço ELB da Amazon Web Services.

MG

Marcos Galvão 23/09/2020

A escalabilidade permite ao sistema atender a maiores demandas, sendo assim aumentar sua capacidade. Elasticidade está intrinsecamente ligado a escalabilidade, pois a elasticidade garante a capacidade da aplicação aumentar ou diminuir sua capacidade/recursos de acordo com o nível da demanda.

Marcos Galvão 23/09/2020

No contexto de MBA, assim como já citaram, elasticidade é fundamental para, primeiramente, permitir a aplicação a escalar e atender as maiores demandas vinda dos consumidores e, segundo, liberar recursos que estão ociosos e não são necessários para um baixo nível de demanda, gerando assim uma economia considerável no custo de operação da aplicação.

DL

Danilo Lira 23/09/2020

Como citado anteriormente a escalabilidade é aumentar a capacidade do serviço de acordo com a demanda, já a elasticidade é a forma como a aplicação lida com a escalabilidade. Essa característica define o quanto bem uma aplicação consegue escalar seus serviços para obtenção de um resultado otimizado. No contexto de microserviços, elasticidade pode ser vista em como a aplicação lida com a escalabilidade de cada um de seus serviços e tem grande foco em economia de recursos e aumentar disponibilidade

Mateus Nunes 23/09/2020

A Elasticidade proporcionada pela computação em nuvem dá às empresas capacidade de reduzir e ampliar dinamicamente as capacidades computacionais dos serviços contratados baseado na demanda atual da organização. A elasticidade é uma estrutura que dá às empresas mais capacidade de ser tornar escaláveis, para atender maiores demandas sem prejudicar o desempenho do software

VS

victor sena de lima attar 23/09/2020

Escalabilidade e elasticidade são dinâmicas complementarem de balanceamento de carga. A escalabilidade faz parte do desenho arquitetural e define como os serviços devem ser aumentados e diminuídos para que toda a aplicação se mantenha em pé. Elasticidade faz parte de eventos programados e pontuais, onde já se espera o aumento (ou diminuição) de demandas, podendo crescer ou diminuir os recursos computacionais.

JS

José Sheldon Brito Fekete 23/09/2020

A escalabilidade é a capacidade de atender a demandas maiores de trabalho sem afetar o desempenho, enquanto a elasticidade funciona dinamicamente reduzindo ou aumentando a capacidade de um sistema conforme a demanda do cenário atual, assim diminuindo o custo computacional podendo transferir recurso computacional de um processo ocioso para um mais requisitado

IN

Izabella Nascimento 23/09/2020

Escalabilidade é a capacidade de um negócio em conseguir suprir uma alta quantidade de demanda em relação àquilo ao que se propõe a fazer, e a elasticidade é definida na capacidade de se adaptar à carga de trabalho, tendo a capacidade de escalar de forma rápida e fácil quando quiser variar os recursos.

Hackenge

Métodos [de trabalho]

Hackathon + Challenge = Método [de trabalho] cujo foco é criar um desafio prático para os participantes

Referências

Wiki | Hackathon

<https://pt.wikipedia.org/wiki/Hackathon>

Links compartilhados

VG

Vinicius Garcia

Netflix Zuul

<https://github.com/Netflix/zuul>

LB

Lucas Barros

Gloo

<https://docs.solo.io/gloo/latest>

RE

Ricardo Ebbers Carneiro Leão

[hscm, recl, wmo] Implementação de api-gateway com Spring Cloud Gateway

<https://github.com/ricardoebbers/api-gateway-example>

RM

Rafael Mota Alves

[rma7,lab7,gap,hscs] Configurando um API Gateway com Eureka, Zuul e Docker

<https://medium.com/@rma7/configurando-um-api-gateway-com-eureka-zuul-e-docker-8e02928b6cef>

LB

Lucas Barros

[lbam, sgfl, lccao] Microsserviços com Gloo API Gateway no Kubernetes

<https://medium.com/@lucasbalmeida9/microsservi%C3%A7os-com-gloo-api-gateway-no-kubernetes-e8331d21dffcc>

JV

Josenildo Vicente

[rjmf, jvsn, jva] Configurando uma API Gateway com Zuul

<https://medium.com/@JosenildoVicente/configurando-uma-api-gateway-com-zuul-d919732a00ce>

RP

Ramom Pereira dos Santos Silva

Microsserviços com API Gateway Zuul e Docker

<https://medium.com/@rpss/microsservi%C3%A7os-com-api-gateway-zuul-e-docker-9242a85f65cb?sk=6a3769fb1692baf310f86cca4e030464>

VS

victor sena de lima attar

Youtube indiano do zuul

https://www.youtube.com/watch?v=-I-9gK8NWXY&list=PLq3uEqRnr_2GleAdJYmlBkB_RfbjMGdoH&ab_channel=GreenLearner

VS

victor sena de lima attar

Apigateway em nodejs

<https://www.express-gateway.io/>

DL

Danilo Lira

[drla, evfl, grro, vsla] Começando com API Gateway e Zuul

<https://medium.com/@danilo.lira01/como-come%C3%A7ando-com-api-gateway-e-zuul-5703422e6068>

VG

Victor Gabryel

[vgss, ipcn, djs] Como configurar o Eureka e o Zuul como API Gateway em uma aplicação em NodeJs e usando Docker.

https://medium.com/@vgss_40348/como-configurar-servi%C3%A7os-em-nodejs-usando-o-eureka-e-o-zuul-como-api-gateway-com-docker-51e1439652f

IF

Igor Fernandes

[ifc, jsbf, matg] Hands-On – API Gateway: Zuul vs Nginx

https://medium.com/@matg_31194/hands-on-api-gateway-zuul-vs-nginx-45711c2665b6

Qual o desafio proposto?

 **Vinicius Garcia** 16/09/2020
Como usar o proxy Zuul como API gateway e como configurar o recurso de alta disponibilidade no Zuul?

 **Vinicius Garcia** 16/09/2020
Escrever um post no Medium ou LinkedIn com o "hands on" da implementação desse caso didático.

Existem premissas, restrições ou condições?

 **Vinicius Garcia** 16/09/2020
Implementar um cenário de aplicação de resolução do problema de escalabilidade utilizando o Zulu (ou algum equivalente a sua escolha). A aplicação deve conter características que a classificam como uma MBA.

 **Vinicius Garcia** 16/09/2020
Zulu = Zuul

 **Vinicius Garcia** 16/09/2020
Sugestão: Lets Chat

Observações adicionais?

 **Vinicius Garcia** 16/09/2020
O proxy Zuul usa internamente o servidor Eureka para descoberta de serviço e Ribbon para平衡amento de carga entre instâncias de serviço. Ele também é capaz de rotear, monitorar, gerenciar resiliência, segurança e assim por diante. Em termos simples, podemos considerar o Zuul um serviço de proxy reverso. Com o Zuul, podemos até mesmo mudar o comportamento dos serviços subjacentes, substituindo-os na camada API.

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes
<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

 **Igor Fernandes** 21/09/2020
Balanceadores de carga (software) geralmente estão ouvindo todo o tráfego da conexão ou somente redirecionam a conexão? Com ouvir todo o tráfego eu quero dizer que toda a informação que chega nesse programa, o programa envia esse pacote para a máquina do serviço desejado (usando alguma política de balanceamento).

 **Igor Fernandes** 21/09/2020
Com redirecionar a conexão eu quero dizer que o balanceador é como um servidor DNS, ele diz somente para o cliente o IP da máquina que ele deverá se conectar, e todo o tráfego da conexão é entre o cliente e a máquina do serviço. O custo computacional/memória de guardar num buffer o pacote recebido e enviar o pacote para máquina do serviço, na primeira situação que eu comentei, é um custo que chega a fazer uma diferença quando comparado com o segundo tipo citado acima?

 **Vinicius Garcia** 23/09/2020
A função básica de um BC é servir de porta de entrada para TODAS as conexões que chegam e cuidar de direcioná-las, segundo alguma política pré-estabelecida, para o cluster que está "atrás" dele. De certa forma, ele está plenamente CAPACITADO para "ouvir" o que chega

 **Vinicius Garcia** 23/09/2020

Se fosse apenas um redirecionamento, seria uma outra coisa, como um Hub ou roteador (na verdade, roteador é mais inteligente que o hub).

 **Vinicius Garcia** 23/09/2020

Ele não diz para o cliente qual o IP da máquina, como um DNS. Ele assegura justamente o contrário! O único IP conhecido é o do BC. Temos uma arquitetura Shared-Nothing para garantir isso.

 **Vinicius Garcia** 23/09/2020

Quanto ao custo, não existe bem uma persistência né, fica tudo na memória volátil... a não ser que o BC exista com outras finalidades

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Escalonamento automático

Sistemas [de tecnologia]

Em aplicações nativas de nuvem, o escalonamento automático é uma técnica de habilitação central para se adaptar às mudanças de carga de trabalho por meio do escalonamento horizontal. No entanto, torna-se um problema desafiador em um sistema de microserviços, uma vez que tal sistema geralmente compreende um grande número de instâncias diferentes com interações complexas.

Referências

Why Microservices Should Be Event Driven: Autonomy vs Authority

<http://blog.christianposta.com/microservices/why-microservices-should-be-event-driven-autonomy-vs-authority/>

3 Easy Things to Do to Make Your Microservices More Resilient

<http://blog.christianposta.com/microservices/3-easy-things-to-do-to-make-your-microservices-more-resilient/>

Lightning Talk - Micro-Services Lifecycle Management at Twitter by Micheal Benedict, Twitter
<https://youtu.be/Q1CEgPw7CG8>

Microsoft Azure | Dimensionamento automático

<https://docs.microsoft.com/pt-br/azure/architecture/best-practices/auto-scaling>

Links compartilhados

 **Rafael Mota Alves**

Top 5 AWS Auto Scaling Strategies

<https://www.missioncloud.com/blog/top-five-aws-auto-scaling-strategies>

 **Rafael Mota Alves**

Autoscaling: Its Purpose and Strategies

<https://rollout.io/blog/autoscaling-purpose-strategies/>

 **Gabriel D'Luca**

AWS Auto Scaling: Escalabilidade de aplicativos para otimizar desempenho e custos

<https://aws.amazon.com/pt/autoscaling/>

 **Danilo Lira**

Dimensionamento sob demanda - VMware

<https://www.vmware.com/br/cloud-solutions/hybrid-cloud/scalability.html>

Quais são os indicadores que remetem a escolha de implementação de escalonamento automático em uma solução?

 **Rafael Mota Alves** 24/09/2020

Eu diria que temos 3 principais indicadores para uma que uma solução use escalonamento automático: O caso mais conhecido, que seria o caso de solução que tem padrões de uso muito irregulares, tendo momentos de pico e vales de carga, um exemplo é uma aplicação de e-commerce, que em dia normais tem determinada carga usual, mas dias/horários especiais, como black friday, pode ter aumentos de carga em questão de segundos.

 **Rafael Mota Alves** 24/09/2020

Outro caso seria os sistemas críticos, que apesar de não terem o caso anterior muito comum, não podem correr o risco de ter downtime ou problemas de performance, caso em algum momento essa carga aumente

 **Rafael Mota Alves** 24/09/2020

e um último caso seria soluções que tem usos muito esparsos, e portanto não precisam ter sua infraestrutura provisionada a todo momento, nesse caso é interessante usar o auto scaling para provisionar os recursos da aplicação apenas quando ela for usada como uma forma de reduzir gastos.

 **Rafael Mota Alves** 24/09/2020

Portanto, os indicadores seriam: picos de carga, quanto vales de carga e necessidade de alta disponibilidade e tolerância a falhas.

 **Luan Brito** 24/09/2020

Acredito que os casos de mudança brusca de necessidade de recursos é um bom indicador, pois a solução adquire uma capacidade de lidar com as adversidades. Serviços que tem essa necessidade de estabilidade tendem a precisar dessa implementação. De modo genérico, uma atividade deve ser automatizada quando os custos com ela são grandes: comum quando ela é requisitada com certa frequência, ou quando a necessidade dela seja imprevisível, mas que existam gatilhos que "ativem" seu uso

 **Ramom Pereira dos Santos Silva** 26/09/2020

Creio que um fator importante para usar auto-scaling seja para aplicações que possuem picos de utilização e uso de recursos computacionais, por exemplo, ecommerce em Black Friday ou Natal, naturalmente, terão picos de utilização em comparação com outras épocas do ano, então, aplicações com sazonalidade em seu uso é um indicador para utilização de auto-scaling.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Acredito também que empresas que estão em forte expansão de sua base de clientes (ex. Startups), ou seja, empresas que estão escalando seu negócio, pode utilizar da estratégia de escalonamento automático de recursos computacionais, já que com o forte crescimento da quantidade de clientes, o uso e quantidade transações realizadas crescerão também, potencialmente. Logo, aplicações que estão crescendo rapidamente, auto-scaling também ajudará nisso, evitando esforços manuais nesse quesito.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Outro indicador é a quantidade de comunicações entre serviços em uma aplicação, quando isso cresce de forma desproporcional, ou seja, um desses serviços utilizam muito o canal de comunicação, o "receptor" dessa comunicação precisa acompanhar essa demanda, como também, se a quantidade de serviços, no canal, aumentar, também é necessário que o "receptor" esteja apto a suportar a forte alta de demanda, nesse caso, auto-scaling contribuiria bastante.

 **Lucas Barros** 26/09/2020

Os indicadores são picos de uso e estratégia de negócio. O primeiro remete à existência de uma janela de horário específica em que o serviço tem que lidar com uma utilização maior e o segundo (estratégia de negócio) remete à importância de fazer com que os usuários da aplicação não presenciem lentidão (devido a picos de uso) em nenhum momento ao conversar com a aplicação (se isso não é tolerável, escalonamento automático deve ser considerado)

 **Saulo Guilhermino** 27/09/2020

A irregularidade da demanda de recursos da aplicação certamente é um fator decisivo para a implementação de escalonamento, visto que a economia de recursos técnicos e monetários quando a demanda está baixa vale muito mais a pena do que construir uma infraestrutura gigantesca que passará por bons momentos de ocio.

 **Lucas Cardoso** 27/09/2020

Variações de uso de CPU, memória e rede, de modo que: 1. prejudiquem a disponibilidade e performance do serviço, ou 2. levam a um super dimensionamento do serviço em momentos de baixa demanda, ocasionando uma maior custo operacional. A implementação do dimensionamento automático também deve levar esses fatores em conta, de forma a manter a qualidade percebida do serviço e manter custos baixos

 **Lucas Cardoso** 27/09/2020

Observar a periodicidade dos picos de carga também é um fator que deve ser observado,

pois pode ajudar na escolha e projeto do dimensionamento automático

IF Igor Fernandes 27/09/2020

Quando se pensa em adicionar o dimensionamento automático na plataforma, se deve ter em mente como consequência, um aumento da complexidade e de custos. Em casos onde o sistema é muito estável e a carga de trabalho não cresce rapidamente e nem com muita frequência, pode ser melhor usar um dimensionamento manual para evitar essas desvantagens citadas anteriormente.

IF Igor Fernandes 27/09/2020

Portanto o dimensionamento automático é interessante para casos onde a carga de trabalho varia bastante no tempo ou de forma alguma o sistema pode apresentar atrasos ou erros de conectividade (garantir desempenho constante).

GD Gabriel D'Luca 28/09/2020

Concordo com Rafael e Ramon ao mencionarem que os picos e vales de utilização são coisas que devem ser levadas em conta. Respondendo a pergunta em si, acho que a sazonalidade é um grande indicador e isso muitas vezes está atrelado ao objetivo/estratégia de negócio, que muitas vezes se intensificam em datas festivas (como Black Friday, Natal e Páscoa), e ter uma infra elástica o suficiente é muito importante para lidar com o atendimento desse público atípico e pontual.

DL Danilo Lira 28/09/2020

Concordo com Ramom, acredito que a maior necessidade de escalonamento automático esteja relacionada aos picos de utilização. No caso da resposta de Ramom, ele cita eventos sazonais, mas vejo que o escalonamento pode ser considerado quando há variação de acesso durante as horas de um dia. Sites de e-commerce, por exemplo, possuem muito mais acesso em horário comercial se comparado com a madrugada. Essa variação de acesso já pode ser uma boa métrica para utilização ou não desta solução.

RJ Renato Joaquim de Miranda Ferreira 29/09/2020

Aplicações que possuem algum nível de criticidade nos seus serviços, seja por questão de saúde, processo delicado, movimentação financeira e entre outros devem ter essa escolha logo a nível de projeto, uma vez que por um aumento de carga/processamento o serviço ficar indisponível pode ter um sério risco no meio em que está inserido.

HC Heitor Carvalho 29/09/2020

Quando a aplicação está exigindo alta disponibilidade e apresentando picos regulares de aumento ou diminuição de demanda, de forma que seja possível prever o quanto e quando se deve realizar uma alteração no dimensionamento. O contrário também pode acontecer, isto é, a aplicação apresenta momentos de baixa utilização onde recursos são desperdiçados desnecessariamente, possivelmente elevando o custo. Dessa forma o dimensionamento seria "para baixo"

JS José Sheldon Brito Fekete 30/09/2020

Aplicativos que tenham uma grande variação de requisições dos seus serviços durante o dia ou até mesmo em épocas específicas, como é o caso dos e-commerces, também é o caso de aplicações críticas, que não podem dar ao luxo de ficar indisponível ou ter queda no seu desempenho.

VS victor sena de lima attar 30/09/2020

Como fatores para implementação de escalonamento, o principal é a quantidade de acesso, pois ela define bem quando deve ser aumentado a quantidade de disponibilidade, mas existem outros como criticidade. Grandes demandas são vistas em datas comemorativas e blackFriday para o comércio, sites do governo em eleições e provas nacionais.

MG Marcos Galvão 30/09/2020

Eu vejo a variância no número de acessos como um dos fatores principais, no caso uma aplicação que tenha alta demanda em momentos bem específicos durante o dia, isso pode ser tomado como base para utilização do escalonamento automático a fim de evitar desperdício de recursos.

Quais as abordagens mais indicadas para considerar na implementação de um escalonamento automático, de acordo com os indicadores levantados anteriormente?

RM Rafael Mota Alves 24/09/2020

Essencialmente, temos 3 técnicas principais para implementar os escalonamento automático, e a escolha delas depende bastante do caso de uso, e também do padrão de carga do sistema. Essas abordagens são: escalonamento agendado, escalonamento dinâmico e escalonamento preditivo

RM Rafael Mota Alves 24/09/2020

No escalonamento agendado, os ajustes na infraestrutura são feitos em momentos definidos manualmente pelo desenvolvedor, essa técnica pode ser útil pra sistemas que tem um padrão de uso consistente e previsível, além de ser útil para datas especiais como em grandes promoções.

 **Rafael Mota Alves** 24/09/2020

No escalonamento dinâmico, nesse tipo, a decisão sobre o escalonamento ou não da infraestrutura é feito a todo momento, geralmente observando métricas definidas pelo desenvolvedor como: uso de memória, uso de CPU, quantidade de requisições, esse tipo de escalonamento é útil para sistemas com cargas imprevisíveis, no entanto esse técnica pode causar problemas enquanto a carga aumenta, visto que o escalonamento irá acontecer em tempo real.

 **Rafael Mota Alves** 24/09/2020

No escalonamento preditivo, são usados os dados de uso do sistema do passado (geralmente limitado a uma janela de tempo mais recente) para determinar quando o sistema deve ser escalado, essa técnica pode ser usada em sistemas que tem cargas um pouco previsíveis, mas que podem mudar com o tempo. Essas técnicas podem ser usadas em conjunto.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Novamente, vai depender muito da natureza do negócio. Porém, acredito que algumas estratégias são interessantes para escalonamento automático. O primeiro deles seria o escalonamento dinâmico, onde a quantidade instâncias cresce e diminui automaticamente, sempre atrelados a algum target, normalmente CPU ou RAM. Nesse caso, acredito que aplicações com picos de uso sazonais (ecommerce na Black Friday), se beneficia bastante disso.

 **Ramom Pereira dos Santos Silva** 26/09/2020

No caso de empresas em forte crescimento e comunicação entre serviços, acredito que o escalonamento preditivo seja uma boa, já que permite configurar limites máximos e mínimos para o auto-scaling, atrelados a alguma métrica. Nesses casos, uma abordagem interessante seja os shards, onde serviços ou até infraestruturas sejam replicadas automaticamente para suportar o aumento da carga de trabalho da aplicação.

 **Igor Fernandes** 27/09/2020

Depende. Se a organização conhece como é o uso da aplicação no tempo (quanto é usado de processamento e memória em um determinado período de tempo), é mais adequado um dimensionamento agendado, pois a empresa evita os atrasos de implantação automática.

 **Igor Fernandes** 27/09/2020

Caso seja uma informação desconhecida ou muito volátil, pode-se usar o dimensionamento automático baseado em métricas. O sistema orquestrador acompanha o uso de memória, processamento e tamanho da fila de requisições para determinar se precisar aumentar ou diminuir a quantidade instâncias de um serviço.

 **Gabriel D'Luca** 28/09/2020

A abordagem mais indicada é algo bem dependente da situação. Concordo com os pontos trazido por Rafael e Igor de que, caso a organização já possua vivência para identificar um pico de uso sazonal, isso pode ser resolvido por um dimensionamento agendado. No entanto, se a frequência desses picos e vales de utilização for maior (flutuação semanal ou diária), talvez seja mais interessante recorrer a um escalonamento dinâmico ou preditivo (caso hajam métricas de uso da aplicação em PROD).

 **Heitor Carvalho** 29/09/2020

Escalonamento automático agendado, onde o escalonamento é feito em quantidade e horário previamente definidos por quem configurou o sistema, Escalonamento preditivo, onde o escalonamento é feito com base em dados e métricas coletados previamente do sistema, e escalonamento dinâmico, onde o escalonamento é feito em tempo real conforme a demanda aumenta ou diminui no sistema.

 **José Sheldon Brito Fekete** 30/09/2020

Sobre a questão de algumas aplicações como e-commerce, que possui algumas datas específicas que irá ocorrer uma maior procura pelos seus serviços, como em datas festivas, é recomendado utilizar o escalonamento agendado, onde é configurado baseado em um estudo prévio dos picos de requisições.

 **José Sheldon Brito Fekete** 30/09/2020

Mas quando os picos de requisições não são previamente conhecidos e não possui um padrão estabelecido é recomendável a utilização de estratégias como escalar juntos com a demanda, sendo capaz de responder a picos de tráfego que você não pode prever.

 **Marcos Galvão** 30/09/2020

Não existe solução única para todos os casos, em uma circunstância como a que citei na questão anterior, pode ser utilizado um Dimensionamento Agendamento, visto que o horário influência diretamente na demanda, outrossim numa aplicação mais imprevisível uma abordagem de Dimensionamento baseado em Metricas pode ser mais recomendado, visto que ela acompanharia o consumo de recursos, atendendo as demandas necessárias durante a execução.

Como você pode implementar o dimensionamento dinâmico nativo de nuvem? Traga exemplo.

 **Rafael Mota Alves** 24/09/2020

Existem várias formas de implementar o dimensionamento dinâmico em um ambiente de nuvem, podendo utilizar serviços "gerenciados", que abstraem a complexidade do dimensionamento, podendo até dar ao desenvolvedor se definir as métricas que vão coordenar o dimensionamento, mas podendo também abstrair isso. Exemplos desses tipos serviço são: Heroku (Paas), AWS Fargate (para Containers), AWS Auto Scaling (Para máquinas virtuais) e AWS Lambda (Serverless). Mas podendo também utilizar plataformas como

 **Rafael Mota Alves** 24/09/2020

Kubernetes e Docker Swarm, que podem executar em máquinas virtuais dos provedores de Cloud e se integrar com as APIs do serviços de Cloud instanciar os recursos necessários para o dimensionamento, dando o controle do processo de dimensionamento dinâmico (quase) total ao desenvolvedor.

 **Luan Brito** 24/09/2020

<https://aws.amazon.com/pt/autoscaling/faqs/> O serviço da amazon parece interessante, eles fornecem "com apenas alguns cliques" todo o plano de escalabilidade. Definindo como cada recurso da aplicação será escalonado, eles possuem até algumas métricas disponíveis que podem priorizar disponibilidade, otimização de custos ou ambos.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Antes de tudo, é necessário analisar a natureza da aplicação a ser provisionada em ambientes de nuvem, levando em consideração arquitetura de infraestrutura e custo (importante!!!). Basicamente, existem duas formas que acredito serem bem interessantes: Orquestração de Containers e Serviços Serveless.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Para aplicações executando em serviços de Orquestrações de Containers, e o Kubernetes é uma solução ótima para esses casos, eu utilizaria os mecanismos de escalonamento automático horizontal de Pods, atrelado a targets de CPU e Memória RAM (dependendo da aplicação), na maioria dos casos, mas depende da necessidade da aplicação. Além disso, utilizaria recursos de métricas para analisar o desempenho da aplicação, Service Mesh (Istio, Linkerd, etc) e Grafana, são alguns exemplos para isso.

 **Ramom Pereira dos Santos Silva** 26/09/2020

Para aplicações que encaixam em arquitetura Serveless, acredito que existem diversas soluções no mercado. Funções como serviço (ex: AWS Lambda) podem ser utilizadas para sistemas orientados a eventos, assim como Containers Serveless (ex: Cloud Run, AWS Fargate) também podem oferecer diversos benefícios, assim como, o Azure Static Web App, muito útil para front-end. Esses serviços são interessantes quando analisamos o custo, já que não é necessário provisionar infraestrutura, só paga pelo que usa

 **Lucas Barros** 26/09/2020

Utilizando Kubernetes, podemos facilmente implementar utilizando uma de suas funcionalidades: HPA (Horizontal Pod AutoScaling) <https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/> para escala horizontal Configurando os recursos dos containers <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#:~:text=Each%20Container%20has%20a%20limit%20of%200.5%20cpu%20and%20128MiB%20of%20memory>. para o caso de escala vertical

 **Lucas Cardoso** 27/09/2020

Usando as ferramentas dos serviços de infraestrutura em nuvem, pode-se facilmente (alguns cliques como dito por Luan) implementar o dimensionamento dinâmico das aplicações. Para que esse dimensionamento seja feito, é necessário criar uma política de escalonamento. Nela pode-se definir metas de uso de CPU, memória e/ou consumo de rede que os serviços observam e escalam o sistema de acordo.

 **Lucas Cardoso** 27/09/2020

Ex: No portal da Azure é possível definir isso (<https://docs.microsoft.com/en-us/azure/azure-monitor/platform/autoscale-get-started>)

 **Saulo Guilhermino** 27/09/2020

Como a Nuvem é um conjunto transparente de máquinas que compartilham o processamento de alguma atividade, pode-se implementar o dimensionamento a partir da programação de alocação automática de mais máquinas para o processamento da atividade requerente.

 **Igor Fernandes** 27/09/2020

Grandes companhias como Microsoft, Amazon e Google possuem serviços em Nuvem especializados para dimensionamento automático. A Google por exemplo exige que certas configurações sejam feitas para ativar o recurso, como por exemplo:

 **Igor Fernandes** 27/09/2020

Política de Escalamento (com base na utilização de CPU, no serviço do balanceamento de carga ou nas métricas da Cloud Monitoring), período de espera (tempo para que o escalonador automático comece a coletar métricas de um instância que está sendo inicializada) e período de estabilização (tempo de observação da demanda de trabalho).

 **Gabriel D'Luca** 28/09/2020

Como já trouxeram, o AWS AutoScaling é uma opção prática de usar, e pode ser associada a algumas outras ferramentas do ecossistema da Amazon, como o AWS CloudWatch, que serve pra definir as métricas capazes de afetar a performance da aplicação. Além disso, essa ferramenta também permite combinar escalonamento preditivo com políticas de escalonamento dinâmico, de forma que a garantir que os picos de utilização também sejam gerenciados de uma forma mais suave.



Heitor Carvalho 29/09/2020

Google, AWS, Kubernetes (no caso de uma arquitetura baseada em containers). Todas essas alternativas permitem que seja configurado com facilidade na aplicação serviços de escalonamento automático com regras definidas pelo usuário. A amazon tem apostado fortemente nesse setor ultimamente com a "amazon elastic container service" que promete servidores virtuais que aumentam e diminuem seus recursos automaticamente conforme a demanda.



Danilo Lira 30/09/2020

Para aplicar o dimensionamento dinâmico em nuvem, é possível utilizar diversos serviços que já foram citados pelo colegas. Esses serviços oferecem diversas ferramentas para personalizar o dimensionamento de acordo com métricas definidas pelo administrador do sistema. Encontrei um serviço chamado VMware, ele apresenta benefícios bastante semelhante aos de outras soluções como aumento da disponibilidade do serviço e ajuste da capacidade sob demanda, o serviço utiliza o AWS como base da sua infra.



José Sheldon Brito Fekete 30/09/2020

Existem diversas empresas que fornecem esse serviço, tais como Microsoft Azure, Google cloud e o AWS Auto Scaling. Todos esses serviços citados trazem um dimensionamento dinâmico nativo em nuvem com facilidade e praticidade para a sua configuração.



victor sena de lima attar 30/09/2020

Para implementar o dimensionamento de nuvem, deve-se escolher o sistema de cloud a ser utilizado, como AWS, Azure ou google Cloud. Dentro de cada um existem ferramentas de monitoramento baseado em métricas e de escalabilidade, para configurar o dimensionamento no seu serviço.



Marcos Galvão 30/09/2020

O AWS Auto Scaling oferecido pela amazon permite configurar a estratégia de escalonamento, podendo escolher entre dinâmico ou preditivo, das máquinas virtuais, um bom ponto do serviço é sua integração com os demais produtos oferecidos pela amazon, permitindo de tal forma o escalonamento de maneira facilitada.

◆ Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

Nenhum comentário realizado

◆ Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

🔗 Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Nenhum comentário realizado

Hackenge

Métodos [de trabalho]

Hackathon + Challenge = Método [de trabalho] cujo foco é criar um desafio prático para os participantes

Referências

Wiki | Hackathon

<https://pt.wikipedia.org/wiki/Hackathon>

Links compartilhados

 **Rafael Mota Alves**

Artigo rma7, hscs, lab7 e gap

<https://medium.com/@rma7/dimensionamento-e-o-gerenciamento-do-ciclo-de-vida-de-microservi%C3%A7os-um-estudo-de-caso-um-tanto-347ffbe950ad>
 **Lucas Cardoso**

[lbam, lccao, sgf] Escalonamento de microserviços

https://medium.com/@lccao_3274/escalonamento-de-microservi%C3%A7os-a7c7387bdb3?sk=3ed93dda00367d934e500a3947f0e81a
 **Ramom Pereira dos Santos Silva**

Artigo [afa4, mnngs e rpss]

<https://medium.com/@rpss/dimensionamento-autom%C3%A1tico-de-microservi%C3%A7os-9aee38022180?sk=81eb92c3d99a977cd1be4b2fe6103cee>
 **Igor Fernandes**

Kubernetes by Example

<https://kubernetesbyexample.com/>
 **Josenildo Vicente**

[rjmf, jvsn, jva] Um olhar sob os Dimensionamentos

<https://medium.com/@JosenildoVicente/um-olhar-sob-os-dimensionamentos-d48410d03868>
 **Igor Fernandes**

[ifc, jsbf, matg, vgss] Dimensionamento Automático: Let's Chat + Nginx

https://medium.com/@ifc_7168/dimensionamento-autom%C3%A1tico-lets-chat-nginx-f847d1f26b24
 **Danilo Lira**

[drla, evfl, grro, vsla] Dimensionamento automático

<https://medium.com/@danilo.lira01/dimensionamento-autom%C3%A1tico-em-microservi%C3%A7os-auto-scaling-56d1a3fe31ac>

Qual o desafio proposto?

 **Vinicius Garcia** 23/09/2020

Escrever um post no Medium ou LinkedIn tomando como caso de estudo o "hands on" da implementação do caso didático anterior.

Existem premissas, restrições ou condições?

 **Vinicius Garcia** 23/09/2020

O seu artigo deverá cobrir, mas não limitado, aos seguintes tópicos: - O conceito básico de dimensionamento automático e abordagens diferentes para dimensionamento automático

 **Vinicius Garcia** 23/09/2020

- O que é dimensionamento dinâmico e como se diferencia do automático. - Por que o dimensionamento dinâmico é importante em um contexto de microserviços.

 **Vinicius Garcia** 23/09/2020

- Como você pode implementar o dimensionamento dinâmico nativo de nuvem no seu exemplo. Justifique. - A importância e os recursos de um gerente de ciclo de vida no contexto de microserviços no seu exemplo. Justifique.

Observações adicionais?

Nenhum comentário realizado

Auto-organização de microsserviços

Sistemas [de tecnologia]

Ao adotar o auto dimensionamento, um fator importante que vem sido abordado por provedores de nuvem é não só adotar a replicação como meio de escalar os serviços, mas utilizar de uma abordagem para organizar as réplicas de serviços de maneira inteligente. Por exemplo, ao definir que novas réplicas de serviços sejam alocadas em diferentes regiões, a infraestrutura colabora para que a aplicação se torna mais resiliente mesmo que venham a ocorrer falhas em determinada(s) região(ões) onde determinada(s) réplica(s) esteja(m) situada(s).

Referências

Manual scaling for Amazon EC2 Auto Scaling
<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-manual-scaling.html>

Visão geral do dimensionamento automático no Microsoft Azure
<https://docs.microsoft.com/pt-br/azure/azure-monitor/platform/autoscale-overview>

New – Predictive Scaling for EC2, Powered by Machine Learning
<https://aws.amazon.com/pt/blogs/aws/new-predictive-scaling-for-ec2-powered-by-machine-learning/>

Example: Distributing instances across Availability Zones
<https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-benefits.html#arch-AutoScalingMultiAZ>

Links compartilhados

-  **Rafael Mota Alves**
PRINCIPLES OF CHAOS ENGINEERING
<https://principlesofchaos.org/>
-  **Rafael Mota Alves**
Chaos Engineering Upgraded
<https://netflixtechblog.com/chaos-engineering-upgraded-878d341f15fa>
-  **Saulo Guilhermino**
O que é autoscaling? Por que você deve considerar essa solução?
<https://www.somasagility.com.br/o-que-e-autoscaling-por-que-voce-deve-considerar-essa-solucao/>
-  **Ramom Pereira dos Santos Silva**
MELIcast #5 Fury – A PaaS do Mercado Livre
<https://open.spotify.com/episode/004aGMxzfOJWiat8K6lQZG>
-  **Gabriel D'Luca**
[gdsv/rosf3] Artigo - AWS e a auto-organização de microsserviços
<https://medium.com/@gdsv/aws-e-a-auto-organiza%C3%A7%C3%A3o-de-microsservi%C3%A7os-cf2de956a5f4>
-  **Marcos Galvão**
Escalonamento automático de Cluster - Google
<https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-autoscaler>
-  **Marcos Galvão**
AWS - Scaling throughout zones
<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-add-availability-zone.html>
-  **José Sheldon Brito Fekete**
Escalonamento automático de instâncias do Google Cloud
<https://cloud.google.com/compute/docs/autoscaler?hl=pt-br>
-  **Marcos Galvão**
Machine learning-based auto-scaling for containerized applications
<https://link.springer.com/article/10.1007/s00521-019-04507-z>

Uma instância prática dessa abordagem pode ser visualizada em um súbita interrupção nos serviços do AWS ocorrido setembro em 2015. Na época, Netflix, umas das empresas a fazer uso do AWS, pouco sofreu com essa interrupção pois as réplicas de seus serviços estavam situadas em diferentes

regiões do AWS. Quais políticas ou diretrizes você destacaria como importantes no momento de definir uma abordagem de auto-organização aliada ao auto-dimensionamento?



Nenhum comentário realizado

Qual a estratégia da auto-organização praticada no AWS junto a sua ferramenta de auto-escalonamento

Nenhum comentário realizado

Além do AWS, quais abordagens, técnicas, metodologias, mecanismos existem?

Nenhum comentário realizado

O mais que vem acontecendo nesse campo de estudo?

Nenhum comentário realizado

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

Nenhum comentário realizado

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Hackenge

Métodos [de trabalho]

Hackathon + Challenge = Método [de trabalho] cujo foco é criar um desafio prático para os participantes

Referências

Wiki | Hackathon

<https://pt.wikipedia.org/wiki/Hackathon>

Links compartilhados



 Vinicius Garcia

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Avaliação parcial da disciplina IF107 (2020.3)
<https://forms.gle/SnSSYGGun5fVMsE78>

Qual o desafio proposto?

Nenhum comentário realizado

Existem premissas, restrições ou condições?

Nenhum comentário realizado

Observações adicionais?

Nenhum comentário realizado

Logs e Monitoramento

Tópicos [de aprendizagem]

Um dos maiores desafios, devido à natureza distribuída da implantação de MBAs em larga escala, é o registro e monitoramento de cada um dos microsserviços individualmente. É uma tarefa muito difícil rastrear transações de ponta a ponta correlacionando os registros emitidos por diferentes microsserviços.

🔗 Referências

Monitoring containerized microservices with a centralized logging architecture. A case study of Project Horus
<https://hackernoon.com/monitoring-containerized-microservices-with-a-centralized-logging-architecture-ba6771c1971a>

Who monitors the monitoring systems?
<https://medium.com/@adrianco/who-monitors-the-monitoring-systems-715a333f97fc>

Cinco coisas que todo desenvolvedor de software deve saber sobre Arquitetura de Software
<https://www.infoq.com.br/articles/architecture-five-things>

Logging and Monitoring Microservices
<https://github.com/IF1007/if1007/blob/master/lectures/if1007-microservices-08.pdf>

Crosscutting Concerns: Monitoring (Chapter 7 from Len Bass' book)
<https://github.com/IF1004/if1004/blob/master/lectures/if1004-devops-09.pdf>

🔗 Links compartilhados

 Danilo Lira
Zipkin
<https://zipkin.io>

 Danilo Lira
Fluentd
<https://www.fluentd.org/architecture>

 Arthur Frade de Araújo
Slack API
<https://api.slack.com/>

 Arthur Frade de Araújo
Telegram API
<https://core.telegram.org/>

Qual o estado atual do cenário do Log & Monitoramento para o desenvolvimento/migração de soluções para arquitetura nativa de nuvem (Microservice Based Applications - MBA)?

Nenhum comentário realizado

✍ Quais as principais abordagens para gestão de logs e monitoramento?

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Nenhum comentário realizado

Quais as principais ferramentas?

Nenhum comentário realizado

Quais os desafios e dores para o cenário futuro de logs e monitoramento?

Nenhum comentário realizado

Se fôssemos começar a levantar features (dores, anseios, expectativas, oportunidades) para uma solução de apoio a tomada de decisão para implantação de MBAs (por meio de logs e monitoramento), o que iria nos guiar nessa arquitetura?

Nenhum comentário realizado

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

Nenhum comentário realizado

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

🔗 Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Dúvidas

Métodos [de trabalho]

Método [de trabalho] cujo foco é compartilhar dúvidas com o time

🔗 Referências

Dúvidas frequentes

<https://pt.m.wikipedia.org/wiki/FAQ>

Insira aqui suas dúvidas para que todos possam ajudar

Nenhum comentário realizado

Dona Deda

Métodos [de trabalho]

Método [de trabalho] cujo foco é conversar livremente sobre qualquer coisa

Referências

Papo furado

https://pt.wikipedia.org/wiki/Papo_furado

Fale sobre o q achar relevante para a disciplina e não está relacionado nas questões essenciais de outras caixas de ferramentas

Nenhum comentário realizado

Observabilidade como função

Tópicos [de aprendizagem]

Zach Jory, da Aspen Mesh, afirma que, embora o monitoramento, que visa dar uma ideia da saúde geral de um sistema, tenha existido por muitos anos, o conceito de OBSERVABILIDADE, que visa fornecer dados sobre o comportamento dos sistemas, tornou-se cada vez mais importante para sistemas distribuídos na nuvem, como os comumente encontrados com os microsserviços.

Referências

Observabilidade e microservices: por que precisamos de tracing e métricas eficazes
<https://www.infoq.com/br/news/2018/07/observability-microservices/>

Observability, or Knowing What Your Microservices Are Doing
<https://dzone.com/articles/observability-or-quothowing-what-your-microservic>

Microservices Observability
<https://bit.ly/3iX9LLO>

Links compartilhados

 **Danilo Lira**
Redhat - Service Mesh
<https://www.redhat.com/pt-br/topics/microservices/what-is-a-service-mesh>

 **Rafael Mota Alves**
Monitoring and Observability
<https://copyconstruct.medium.com/monitoring-and-observability-8417d1952e1c>

 **Igor Fernandes**
Data Storytelling Canvas: conte a história dos seus dados
<https://stefanocarnevalli.medium.com/data-storytelling-canvas-conte-a-hist%C3%B3ria-do-seus-dados-9313d786bed9>

 **Roberto Oliveira**
Observabilidade
<https://www.opservices.com.br/observabilidade/>

 **Lucas Cardoso**
The Three Pillars of Observability
<https://www.oreilly.com/library/view/distributed-systems-observability/9781492033431/ch04.html>

Jory acredita que o surgimento de tecnologias de [service mesh], como o Istio, tornou a observabilidade o requisito número um para quem quer usar ou desenvolver com elas. Na sua opinião qual a relação de observabilidade vs microsserviços vs service mesh?

Nenhum comentário realizado



Ainda segundo Jory, os dois principais recursos de observabilidade que seriam fornecidos por um service mesh são rastreabilidade e métricas. Você concorda ou não? Justifique sua resposta.

Nenhum comentário realizado

Qual o principal obstáculo a implementação de observabilidade como função?

Nenhum comentário realizado

Quais são os benefícios que a observabilidade pode trazer para meu ecossistema?

Nenhum comentário realizado

Hackenge

Métodos [de trabalho]

Hackathon + Challenge = Método [de trabalho] cujo foco é criar um desafio prático para os participantes

Referências

Wiki | Hackathon

<https://pt.wikipedia.org/wiki/Hackathon>

Links compartilhados

 Vinicius Garcia

Docker Fundamentals Exercises book

<https://github.com/IF1007/if1007/blob/master/hw/docker-fundamentals-exercises.pdf>

 Vinicius Garcia

Play with Docker: A simple, interactive and fun playground to learn Docker

<https://labs.play-with-docker.com/>

Qual o desafio proposto?

Nenhum comentário realizado

Existem premissas, restrições ou condições?

Nenhum comentário realizado

Observações adicionais?

Nenhum comentário realizado

Equipes

Sistemas [de tecnologia]

Constituição e demais informações das equipes

Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.



Nenhum comentário realizado

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

Nenhum comentário realizado

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

Hackenge

Métodos [de trabalho]

Hackathon + Challenge = Método [de trabalho] cujo foco é criar um desafio prático para os participantes

Referências

Wiki | Hackathon

<https://pt.wikipedia.org/wiki/Hackathon>

Links compartilhados

 Vinicius Garcia

Course on Docker and Kubernetes

<https://www.freecodecamp.org/news/course-on-docker-and-kubernetes/>

Qual o desafio proposto?

Nenhum comentário realizado

Existem premissas, restrições ou condições?

Nenhum comentário realizado

Observações adicionais?

Nenhum comentário realizado

Projeto 2020.3

API sistema arquitetura integração implementar informações
Mota Cluster documentação serviços fazer Definir Iniciar load
logs kafka comunicação plataforma transações cada
Finalizar - configuração | Stack projeto infraestrutura
funcionamento desenvolvimento sobre gerar balancer serviço
endpoints uso possível todos microserviços serão riscos
relevantes métricas monitoramento produto usuário dados
aplicação repositório configurar microsserviços

C

O projeto conta com **41 pessoas**, tendo a seguinte composição: **1 administrador** e **40 membros**.

No período de **20/08/2020** até **30/11/2020** foram aplicados nos mapas modulares um total de **71 ferramentas** (hexágonos). Ao todo, foram registradas **456 falas**. Uma análise de estatística descritiva indica que o projeto possui uma média de **6.42 falas por ferramenta** e de **11.12 falas por membro**.

Na plataforma strateegia.digital existem dois tipos de falas: as respostas diretas às questões e os comentários a essas respostas. Do total de **456 falas** houve **441 respostas (96.71% do total de falas)** e **15 comentários às respostas (3.29% do total de falas)**. Entre as 441 respostas, 12 delas foram provocativas o suficiente para gerar comentários respostas.

Equipe 01

Sistemas [de tecnologia]

Constituição e demais informações da equipe

Referências

tds.company
<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 RM Rafael Mota Alves 22/10/2020
Rafael Mota, Luan Antunes e Heitor Sammuel

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 VG Vinicius Garcia 28/10/2020
Cadê o repositório?

 RM Rafael Mota Alves 03/11/2020
<https://github.com/rafaelmotaalves/microservices-communication>

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

Equipe 02

Sistemas [de tecnologia]

Constituição e demais informações da equipe

Referências

tds.company
<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 JS José Sheldon Brito Fekete 26/10/2020
Igor Fernandes Marcos Galvão Sheldon Fekete Victor Gabryel

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 IF Igor Fernandes 26/10/2020
<https://github.com/Igorxp5/stackprior>

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

Equipe 03

Sistemas [de tecnologia]

Constituição e demais informações da equipe

🔗 Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 **Emerson Victor** 21/10/2020

Danilo Lira (drla) Emerson Victor (evfl) Gabriel Ramos (grro) Victor Sena (vsla)

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 **Vinicius Garcia** 28/10/2020

Cadê o repositório?

 **Emerson Victor** 11/11/2020

<https://github.com/emersonvictor/microservices-20203>

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

Equipe 04

Sistemas [de tecnologia]

Constituição e demais informações da equipe

🔗 Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 **Saulo Guilhermino** 21/10/2020

Saulo Guilhermino, Lucas Cardoso e Lucas Barros

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 **Vinicius Garcia** 28/10/2020

Cadê o repositório?

Tem alguma dúvida ou observação a fazer?

 **Vinicius Garcia** 28/10/2020

Vale a pena dar uma olhada nesse projeto do semestre 2020.1*, de repente vocês podem fazer

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Equipe 05

Sistemas [de tecnologia]

Constituição e demais informações da equipe

🔗 Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 **Ramom Pereira dos Santos Silva** 26/10/2020
Ramom Pereira (rpss) e Arthur Frade (afa4)

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 **Vinicius Garcia** 28/10/2020
Cadê o repositório?

 **Ramom Pereira dos Santos Silva** 08/11/2020
<https://github.com/rpss2/microservices-logs-management>

 **Ramom Pereira dos Santos Silva** 30/11/2020
Link para o screencast do projeto:
<https://www.loom.com/share/3bc7cf642e2147d0baa2788d3568736b>

Tem alguma dúvida ou observação a fazer?

 **Vinicius Garcia** 28/10/2020
Vale a pena dar uma olhada nesse projeto do semestre 2020.1*, de repente vocês podem fazer o de vocês a partir desse: <https://github.com/microobs>

Equipe 06

Sistemas [de tecnologia]

Constituição e demais informações da equipe

🔗 Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

 **Ricardo Ebbers Carneiro Leão** 28/10/2020
Ricardo Ebbers Carneiro Leão (recl) Wellington Oliveira (wmof)

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

 **Vinicius Garcia** 28/10/2020
Cadê o repositório?

  **Ricardo Ebbers Carneiro Leão** 28/10/2020
<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

Equipe 07

Sistemas [de tecnologia]

Constituição e demais informações da equipe

🔗 Referências

tds.company

<https://tds.company/>

Quais são os membros da sua equipe? Basta um dos membros informar a composição da mesma.

RJ Renato Joaquim de Miranda Ferreira 30/11/2020

Equipe final: Josenildo Vicente (jva) Renato Joaquim (rjmf)

Já criou um repositório no github para a equipe? Informe aqui o link pra ele.

VG Vinicius Garcia 28/10/2020

Cadê o repositório?

JV Josenildo Vicente 30/11/2020

<https://github.com/renatojmf/MicroServices-Guide>

RJ Renato Joaquim de Miranda Ferreira 30/11/2020

Link para nossa apresentação: <https://docs.google.com/presentation/d/1uQn-3ve9KfmddQOP5X2OWzL0la2PWXISFSzFXGdYq6w/edit?usp=sharing>

Tem alguma dúvida ou observação a fazer?

Nenhum comentário realizado

💡 desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

🔗 Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

RJ Renato Joaquim de Miranda Ferreira 28/10/2020

Logs e disponibilidade

Quais perguntas podem guiar o time a entender melhor o tema definido?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Como definir bem a comunicação entre os micro serviços?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Como realizar nossos processos sem impactar o usuário final?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Como definir a prioridade dos processos?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Como realizar nossas operações?

Como condensar as informações recolhidas em uma questão essencial?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Como garantir que nossos processos operem corretamente sem impactar o usuário final?

A partir da questão essencial, que desafio será escolhido pelo time?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Busca e recolhimento de dados sob operações (a definir).

 Renato Joaquim de Miranda Ferreira 29/11/2020

Um guia de microsserviços utilizando o Docker

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model

bit.ly/35NfdKH

O que é o Business Model Canvas

bit.ly/2Ylutc3

Proposta de valor

bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example

bit.ly/3be7kAG

Startup: qual sua proposta de valor?

bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor

bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

 Renato Joaquim de Miranda Ferreira 28/10/2020

Ajudar clientes na tomada de decisão; busca de dados facilitada; Métricas de monitoramento.

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento



Renato Joaquim de Miranda Ferreira 28/10/2020

Praticidade na tomada de decisão; Diminuir estresse com a busca de itens.

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

Renato Joaquim de Miranda Ferreira 28/10/2020

Segurança na aplicação, conforto na busca pelos itens, disponibilidade para a busca dos mesmos.

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

Renato Joaquim de Miranda Ferreira 28/10/2020

Não, em um primeiro momento será uma aplicação única para os mais diversos tipos de usuário.

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

Renato Joaquim de Miranda Ferreira 28/10/2020

Facilidade na utilização do sistema e entrega dos resultados esperados pelos usuários.

desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Links compartilhados

Ricardo Ebbers Carneiro Leão

Saga distributed transactions pattern

<https://docs.microsoft.com/pt-br/azure/architecture/reference-architectures/saga/saga> **Ricardo Ebbers Carneiro Leão**

Serverless implementation of Saga

<https://github.com/Azure-Samples/saga-orchestration-serverless> **Ricardo Ebbers Carneiro Leão**

Transações distribuídas em micro-serviços

<https://medium.com/senior/transa%C3%A7%C3%B5es-distribu%C3%ADdas-em-micro-servi%C3%A7os-70568b378d77> **Ricardo Ebbers Carneiro Leão**

Pattern: Saga

<https://microservices.io/patterns/data/saga.html>

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

Ricardo Ebbers Carneiro Leão 28/10/2020

Transações distribuídas no contexto de aplicações baseadas em microserviços

Quais perguntas podem guiar o time a entender melhor o tema definido?

Ricardo Ebbers Carneiro Leão 28/10/2020

O que é uma transação distribuída?

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Como podemos garantir transações ACID (Atomicity, Consistency, Isolation, Durability) que envolvem múltiplos microserviços?

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Como evito estados inconsistentes das minhas entidades do domínio após atualizações que envolvem múltiplos microserviços?

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Como gerencio acessos concorrentes a um recurso que está sendo modificado por uma transação distribuída?

Como condensar as informações recolhidas em uma questão essencial?

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Como garantir ACIDez nas minhas transações dado uma arquitetura orientada a microserviços?

A partir da questão essencial, que desafio será escolhido pelo time?

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Implementar um microserviço genérico que seja capaz de orquestrar ou coreografar transações complexas e possa ser plugado numa aplicação baseada em microserviços

💡 proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

🔗 Referências

what is a business model
bit.ly/35NfdKH

O que é o Business Model Canvas
bit.ly/2Ylutc3

Proposta de valor
bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example
bit.ly/3be7kAG

Startup: qual sua proposta de valor?
bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor
bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Estados inconsistentes causados por transações não ACIDas num contexto de microserviços

RE Ricardo Ebbers Carneiro Leão 28/10/2020
Transações duplicadas, referências inválidas são os principais exemplos de problemas causados por falta de ACIDez

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento

 Ricardo Ebbers Carneiro Leão 28/10/2020

Atomicidade de movimentações

 Ricardo Ebbers Carneiro Leão 28/10/2020

Observabilidade das transações

 Ricardo Ebbers Carneiro Leão 28/10/2020

Transações performáticas e seguras

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

 Ricardo Ebbers Carneiro Leão 28/10/2020

Confiabilidade no sistema, segurança em movimentações financeiras, praticidade para o desenvolvedor

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

Nenhum comentário realizado

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

Nenhum comentário realizado

desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

 Ramom Pereira dos Santos Silva 26/10/2020

Gerenciamento de Logs

Quais perguntas podem guiar o time a entender melhor o tema definido?

 Ramom Pereira dos Santos Silva 26/10/2020

Quais ferramentas existem para captura de logs de uma aplicação MBA?

 Ramom Pereira dos Santos Silva 26/10/2020

Quais dados são importantes para captura?

 Ramom Pereira dos Santos Silva 26/10/2020

É possível utilizar os logs e extrair informações relevantes?

 Ramom Pereira dos Santos Silva 26/10/2020

Conseguimos utilizar essas informações para gerar transparência aos clientes?



Como condensar as informações recolhidas em uma questão essencial?

RP Ramom Pereira dos Santos Silva 26/10/2020

Quais funcionalidades estão em maior uso em nossa aplicação?

A partir da questão essencial, que desafio será escolhido pelo time?

RP Ramom Pereira dos Santos Silva 26/10/2020

Desenvolver mecanismos de captura de logs e fornecer informações relevantes do uso da aplicação para os clientes.

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model
bit.ly/35NfdKH

O que é o Business Model Canvas
bit.ly/2Ylutc3

Proposta de valor
bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example
bit.ly/3be7kAG

Startup: qual sua proposta de valor?
bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor
bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

RP Ramom Pereira dos Santos Silva 26/10/2020

A falta de informações sobre o funcionamento e estado de funcionalidades. A baixa capacidade de recuperação de erros e transparência sobre o desempenho da aplicação para o cliente.

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento

RP Ramom Pereira dos Santos Silva 26/10/2020

Rastreabilidade e Transparência.

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

RP Ramom Pereira dos Santos Silva 26/10/2020

Transparência de utilização da aplicação.

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

RP Ramom Pereira dos Santos Silva 26/10/2020

Não

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

Ramom Pereira dos Santos Silva 26/10/2020

Através de ferramentas de visualização de dados.

desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

Saulo Guilhermino 25/10/2020

Observabilidade em API Gateways

Quais perguntas podem guiar o time a entender melhor o tema definido?

Saulo Guilhermino 25/10/2020

Quais ferramentas existem atualmente e funcionam como API Gateways? É possível se extrair métricas destas ferramentas? Como se pode aumentar a observabilidade na análise das métricas?

Como condensar as informações recolhidas em uma questão essencial?

Saulo Guilhermino 25/10/2020

Como obter métricas de requests direcionadas a um API Gateway?

A partir da questão essencial, que desafio será escolhido pelo time?

Saulo Guilhermino 25/10/2020

Fornecer uma ferramenta/estratégia de API Gateway com alta observabilidade

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model

bit.ly/35NfdKH

O que é o Business Model Canvas

bit.ly/2Ylutc3

Proposta de valor

bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example

bit.ly/3be7kAG

Startup: qual sua proposta de valor?

bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor

bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

**Saulo Guilhermino** 25/10/2020

Falta de métricas e insights para compreender e melhorar o tráfego que passa por um API Gateway

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento

**Saulo Guilhermino** 25/10/2020

Observabilidade

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

**Saulo Guilhermino** 25/10/2020

Confiabilidade

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

**Saulo Guilhermino** 25/10/2020

Não

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

**Saulo Guilhermino** 25/10/2020

Através de uma demo da solução

hexagonal desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

link Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

**Emerson Victor** 26/10/2020

Logs e monitoramento no contexto de e-commerces



Quais perguntas podem guiar o time a entender melhor o tema definido?

 Emerson Victor 26/10/2020

Quais os problemas relacionados a monitoramento em e-commerce?

 Emerson Victor 26/10/2020

Quais os principais dados gerados por um e-commerce?

 Emerson Victor 26/10/2020

Quais os serviços críticos de um e-commerce?

 Emerson Victor 26/10/2020

Quais as melhores formas de centralizar logs?

 Emerson Victor 26/10/2020

Que técnicas de monitoramento podem ser utilizadas nesse contexto?

 Emerson Victor 26/10/2020

Como gerenciar métricas, usando logs como base, para criar insights do negócio?

Como condensar as informações recolhidas em uma questão essencial?

 Emerson Victor 26/10/2020

Como utilizar monitoramento e logs para auxiliar a tomada de decisão?

A partir da questão essencial, que desafio será escolhido pelo time?

 Emerson Victor 26/10/2020

Criar um sistema de monitoramento e centralização de logs com intuito de gerar informações e auxiliar a tomada de decisão (métricas da aplicação, métricas da plataforma e eventos do sistema).

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model
bit.ly/35NfdKH

O que é o Business Model Canvas
bit.ly/2YIutc3

Proposta de valor
bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example
bit.ly/3be7kAG

Startup: qual sua proposta de valor?
bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor
bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

 Danilo Lira 26/10/2020

Dispersão de dados Falta de conhecimento do estado dos serviços Perda de dados gerados pelo serviços Falta de insights no negócio

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento

 **Danilo Lira** 26/10/2020

Centralização de dados para gerar informação Conhecimento das partes do e-commerce
Acompanhamento dos serviços críticos da plataforma

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

 **Danilo Lira** 26/10/2020

Segurança Conhecimento Gestão da informação Escalabilidade

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

 **Danilo Lira** 26/10/2020

Não

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

 **Danilo Lira** 26/10/2020

Sistema de fácil utilização que fornece os dados de forma acessível e gerencia o e-commerce com intuito de garantir a segurança e a estabilidade dos serviços

desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

Referências

CBL na Windward School
bit.ly/2JaRJGP

framework CBL
bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

 **José Sheldon Brito Fekete** 26/10/2020

Disponibilidade

Quais perguntas podem guiar o time a entender melhor o tema definido?

 **Igor Fernandes** 26/10/2020

Como priorizar a comunicação de serviços importantes?

 **Igor Fernandes** 26/10/2020

Qual serviço deve ter atenção durante uma sobrecarga de trabalho?

 **Igor Fernandes** 26/10/2020

Como é a árvore de dependências dos microsserviços da nossa aplicação?



 **José Sheldon Brito Fekete** 26/10/2020

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Como definir a prioridade de um microserviço?

Como condensar as informações recolhidas em uma questão essencial?

 **Igor Fernandes** 26/10/2020

Como garantir que os serviços da nossa aplicação tenham a prioridade adequada na comunicação, baseado na sua importância para o sistema?

A partir da questão essencial, que desafio será escolhido pelo time?

 **Igor Fernandes** 26/10/2020

Desenvolver um load balancer capaz de gerenciar de forma intuitiva as prioridades dos serviços do sistema

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model
bit.ly/35NfdKH

O que é o Business Model Canvas
bit.ly/2Ylutc3

Proposta de valor
bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example
bit.ly/3be7kAG

Startup: qual sua proposta de valor?
bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor
bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?

 **Igor Fernandes** 28/10/2020

Criar árvore de dependência entre os microsserviços

 **Igor Fernandes** 28/10/2020

Definir prioridade da comunicação dos microsserviços manualmente baseado nas regras de negócio

 **Igor Fernandes** 28/10/2020

Definir prioridade da comunicação dos microsserviços de forma automática baseado em padrões da demanda

 **Igor Fernandes** 28/10/2020

Observar demanda dos microsserviços

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento

 **Igor Fernandes** 28/10/2020

Usar os recursos de cloud de forma eficiente evitando gastos desnecessários

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade

 **Igor Fernandes** 28/10/2020

Entender melhor a arquitetura do sistema

 **Igor Fernandes** 28/10/2020

Entender como funciona as demandas da plataforma (de cada microserviço) para conseguir alocar os recursos de forma inteligente a fim de ter um alvo de melhoria

Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

 **Igor Fernandes** 28/10/2020

Não existe mais de um segmento de cliente

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

 **Igor Fernandes** 28/10/2020

Através da plataforma o cliente poderá identificar o engajamento dos usuários nos serviços do sistema e alocar esforços de melhorias na experiência do usuário de forma direcionada.

desafio de projeto

Métodos [de trabalho]

Método [de trabalho] para definir um desafio a ser explorado pelo time

Referências

CBL na Windward School

bit.ly/2JaRJGP

framework CBL

bit.ly/2J9RagA

Qual tema servirá de base para o time seguir a partir da oportunidade encontrada?

 **Rafael Mota Alves** 23/10/2020

Escalabilidade & Elasticidade

Quais perguntas podem guiar o time a entender melhor o tema definido?

 **Rafael Mota Alves** 23/10/2020

como garantir consistência e ordenação na comunicação entre serviços em uma arquitetura escalável?

 **Rafael Mota Alves** 23/10/2020

como balancear a carga de mensagens entre instâncias do mesmo serviço ?

 **Rafael Mota Alves** 23/10/2020

como implementar a comunicação entre serviços de forma a não degradar o desempenho das aplicações?

 **Rafael Mota Alves** 23/10/2020

quais são os tipos de comunicação entre sistemas distribuídos?

Como condensar as informações recolhidas em uma questão essencial?

 **Heitor Carvalho** 25/10/2020

Numa arquitetura escalável e elástica, como podemos garantir que a comunicação entre serviços, seja por aplicações de mensageria ou outras formas, aconteça de forma consistente, confiável, robusta e segura?

A partir da questão essencial, que desafio será escolhido pelo time?



Heitor Carvalho 25/10/2020

Implementar uma infraestrutura (escalável e elástica) consistente, confiável, robusta e segura para comunicação entre microsserviços.

proposta de valor

Métodos [de trabalho]

Método [de trabalho] para identificar o motivo pelo qual os clientes vão escolher o produto ou serviço oferecido em vez dos produtos ou serviços concorrentes. Essa proposta de valor deve ser transformada numa promessa a ser entregue pelo negócio, que é comunicada e percebida pelos clientes

Referências

what is a business model

bit.ly/35NfdKH

O que é o Business Model Canvas

bit.ly/2Ylutc3

Proposta de valor

bit.ly/35Cn9A6

Value Proposition Canvas explained through the Uber example

bit.ly/3be7kAG

Startup: qual sua proposta de valor?

bit.ly/2A6ZhJd

Como Desenvolver sua Proposta de Valor

bit.ly/2W9CTYd

Entre os vários problemas (dores) dos possíveis clientes mapeados, qual o produto ou serviço se propõe a resolver?



Rafael Mota Alves 27/10/2020

Serviços que se comunicam diretamente estão mais acoplados e mais vulneráveis a falhas



Rafael Mota Alves 27/10/2020

Serviços de mensageria são difíceis de escalar



Rafael Mota Alves 27/10/2020

A comunicação é difícil de gerenciar

Qual o valor prático entregue aos clientes no uso do produto ou serviço? Exemplo: Mobilidade, deslocamento



Rafael Mota Alves 27/10/2020

Comunicação Escalável

Qual o valor simbólico entregue aos clientes no uso do produto ou serviço? Exemplo: Conforto, segurança, praticidade



Rafael Mota Alves 27/10/2020

Confabilidade, Segurança e Abstração

C Existe um pacote de produtos/serviços diferente para cada segmento de clientes? Quais?

RM Rafael Mota Alves 27/10/2020

Desenvolvedores de aplicação: utilizarão a infraestrutura para implementar comunicações entre serviços

RM Rafael Mota Alves 27/10/2020

Desenvolvedores de plataforma: gerenciarão a infraestrutura

RM Rafael Mota Alves 27/10/2020

Gerentes e Usuários: serão beneficiados indiretamente

Como a proposta de valor deve ser comunicada para os clientes e experimentada por eles?

RM Rafael Mota Alves 27/10/2020

Apresentando os benefícios que a infraestrutura irá trazer para a organização, pesando com os custos e riscos. Podendo fazer POCs (provas de conceito) para apresentar e validar a solução

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?

LB Luan Brito 29/10/2020

Uma infraestrutura de comunicação para desenvolvedores de microserviços em uma MBA utilizarem

Resuma em algumas linhas o que deverá ter o seu produto:

LB Luan Brito 29/10/2020

fornecer APIs para que os serviços possam enviar e receber mensagens; fornecer API para que a infra possa ser redimensionada; e utilizar mecanismos de load balancing na comunicação

LB Luan Brito 29/10/2020

bônus: fornecer uma API de monitoramento da infra

Quais as CERTEZAS que se têm do que será produzido?

HC Heitor Carvalho 29/10/2020

Temos certeza que não será uma solução de tão alto nível. Será necessário um esforço para acoplar em arquiteturas existentes e um certo know-how por parte da equipe de desenvolvimento para implementar e fornecer manutenção, apesar das abstrações proporcionadas.

HC Heitor Carvalho 29/10/2020

Temos certeza que não será auto escalável, posto que isto exige alto esforço para ser implementado em conjunto com serviços de mensageria

Quais as SUPOSIÇÕES que se têm?

LB Luan Brito 29/10/2020

Supomos que irá proporcionar uma melhora na manutenibilidade e performance geral do sistema que implementar a solução; Supomos que haja benefícios suficientes que justifiquem o custo de implementação da solução



Rafael Mota Alves 04/11/2020

Usaremos algum sistema de mensageria como Kafka ou RabbitMQ para implementar a comunicação

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?



Luan Brito 03/11/2020

Qual a melhor ferramenta para gerenciar mensageria no contexto de MBAs escaláveis?



Luan Brito 03/11/2020

Que APIs devem ser disponibilizadas?



Luan Brito 03/11/2020

como será feito dimensionamento da infraestrutura de comunicação?



Vinicius Garcia 05/11/2020

o legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?



Rafael Mota Alves 29/10/2020

escalabilidade, interoperatividade e desempenho

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?



Rafael Mota Alves 02/11/2020

Redimensionamento da infra de comunicação: facilidade e abstração de operações de escalonagem e escalabilidade



Rafael Mota Alves 02/11/2020

balanceamento de carga: estabilidade, confiabilidade e abstração



Rafael Mota Alves 02/11/2020

troca de mensagens: interoperatividade, consistência e modularização

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agraga valor, não faz sentido, ou qualquer outro motivo)?

Nenhum comentário realizado



Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software
<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos
<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura
<https://bit.ly/30Cbn0E>

The Twelve-Factor App methodology
<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

 **Rafael Mota Alves** 02/11/2020

o sistema deverá ser fácil de usar por usuários não especialistas

Quais são os cinco principais objetivos de negócios para o sistema a ser projetado?

 **Rafael Mota Alves** 02/11/2020

padronizar a comunicação entre serviços

 **Rafael Mota Alves** 02/11/2020

fazer com que a comunicação entre serviços não se torne um gargalo

 **Rafael Mota Alves** 02/11/2020

usar o sistema de comunicação para possibilitar e facilitar o processo de escalar serviços

 **Rafael Mota Alves** 02/11/2020

centralizar o trabalho de gerenciar a infraestrutura de comunicação

 **Rafael Mota Alves** 02/11/2020

abstrair as especificidades do sistema para os desenvolvedores da plataforma

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

 **Rafael Mota Alves** 02/11/2020

Lidar com as dificuldades envolvendo escalonamento com serviços de mensageria criando uma abstração amigável para o nível dos usuários

 **Rafael Mota Alves** 02/11/2020

lidar com a dificuldade de abstrair comunicação entre serviços diferentes

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

 **Rafael Mota Alves** 02/11/2020

os desenvolvedores precisam ter os dados das mensagens e dos serviços que estão enviando e recebendo mensagens

 **Rafael Mota Alves** 02/11/2020

Admins do sistema precisariam além das informações de que serviços estão usando, como eles estão usando e também a saúde do sistema

 **Rafael Mota Alves** 02/11/2020

Os líderes podem querer saber o nível de uso da infra por cada serviço e quantas instâncias da infra estão sendo executadas no momento

C

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?



Igor Fernandes 02/11/2020

Uma Plataforma Web sobre um load balancer para gerenciar as prioridades dos serviços do sistema. Para quem? Software Engineer, Network manager

Resuma em algumas linhas o que deverá ter o seu produto:



Igor Fernandes 02/11/2020

- Registrar serviços no load balancer - Listar de serviços registrados no load balancer - Árvore de dependências dos microsserviços - Definir prioridades da comunicação entre os microsserviços



Igor Fernandes 03/11/2020

Baixa prioridade: - Configurar automaticamente as prioridades dos serviços - Observar quantidade de requisições para os microsserviços

Quais as CERTEZAS que se têm do que será produzido?



Igor Fernandes 02/11/2020

Não vamos criar um load balancer do zero



Igor Fernandes 02/11/2020

Todo o gerenciamento será feito por uma plataforma Web



Igor Fernandes 02/11/2020

Back-end será em Python



Igor Fernandes 02/11/2020

Usaremos Docker



Igor Fernandes 03/11/2020

O produto deve ser capaz de redirecionar a comunicação para qualquer instância de um microsserviço



Igor Fernandes 03/11/2020

Alterar as configurações deve ter uma boa usabilidade, ou seja menos código e mais UI.

Quais as SUPosições que se têm?



Igor Fernandes 02/11/2020

O load balancer base será o Nginx



Igor Fernandes 03/11/2020

Integrantes vão ter o tempo necessário para concluir todos funcionalidades listadas do produto



Igor Fernandes 03/11/2020

É possível alterar as configurações do load balancer sem a necessidade de reiniciar toda a aplicação

IF

Igor Fernandes 03/11/2020

A comunicação entre microsserviços deve passar pelo load balancer, caso contrário não será possível controlar a comunicação entre eles

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

IF

Igor Fernandes 03/11/2020

Como a aplicação vai lidar com o auto-scaler? O operador não deveria precisar adicionar manualmente cada instância de um microsserviço sempre que o auto-scaler atuar

IF

Igor Fernandes 03/11/2020

Como iremos implementar a funcionalidade de priorização da comunicação usando um load balancer base? Caso o load balancer não tenha uma funcionalidade priorização isso será um problema

IF

Igor Fernandes 03/11/2020

Como diferenciar uma comunicação entre microsserviços de cliente-microsserviço? Isso importa?

VG

Vinicius Garcia 05/11/2020

O legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

🔗 Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?

VS

victor sena de lima attar 04/11/2020

Um sistema de monitoramento e centralização de logs para empresas que possuam e-commerce

Resuma em algumas linhas o que deverá ter o seu produto:

VS

victor sena de lima attar 04/11/2020

Centralização de logs, gerar relatórios com informações relevantes sobre o negócio e fornecer informações sobre o funcionamento dos serviços

Quais as CERTEZAS que se têm do que será produzido?

VS

victor sena de lima attar 04/11/2020

Não irá reagir às informações sobre os serviços (autogerenciamento) Será acoplado ao e-commerce com intuito de receber os logs Coletar os logs e centralizar Criação de um dashboard com informações sobre o negócio, geradas a partir dos dados do log, e sobre o funcionamento dos serviços

Quais as SUPosições que se têm?

Gabriel Ramos 04/11/2020

Utilizar o online boutique como base para fornecer os dados Utilizar o ELK Stack / Fluentd

C

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

Gabriel Ramos 04/11/2020

Linguagem que vai ser utilizada

VG Vinicius Garcia 05/11/2020

apenas essa incerteza? tem alguma outra com relação ao negócio em si? as regras de negócio, características, comportamento... algo nessa linha?

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?

SG Saulo Guilhermino 02/11/2020

Uma estratégia de API Gateway com Alta Observabilidade para desenvolvedores de aplicações em Microsserviços

Resuma em algumas linhas o que deverá ter o seu produto:

SG Saulo Guilhermino 02/11/2020

Um componente de API Gateway que gera um alto número de métricas de tráfego (com informações de origem, destino, tamanho de pacote, tipo de pacote, status da conexão, etc.) que são coletadas e disponibilizadas em alguma ferramenta de visualização

Quais as CERTEZAS que se têm do que será produzido?

SG Saulo Guilhermino 02/11/2020

Terá boa uma ferramenta de agregação de logs e visualização de gráficos.

LC Lucas Cardoso 02/11/2020

Que será open source

SG Saulo Guilhermino 02/11/2020

Será composto de pelo menos uma ferramenta de API Gatweay

SG Saulo Guilhermino 02/11/2020

É Destinado ao desenvolvedor

Quais as SUPosições que se têm?

SG Saulo Guilhermino 02/11/2020

Supomos que as métricas fornecidas produzirão bons insights para o time de desenvolvimento otimizar a aplicação e torná-la mais confiável.

SG Saulo Guilhermino 02/11/2020

Supomos que as métricas ajudarão o time de desenvolvimento da aplicação a monitorá-la e realizar procedimentos de debug

SG Saulo Guilhermino 02/11/2020

Supomos que o produto não produzirá grandes impactos negativos de performance

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

 **Saulo Guilhermino** 02/11/2020

Quais métricas devem ser disponibilizadas aos desenvolvedores? (Q: Os gráficos apresentados são de grande valia para o monitoramento e otimização da aplicação?)

 **Saulo Guilhermino** 02/11/2020

Como essas métricas devem ser apresentadas aos desenvolvedores? (Q: Este tipo de dado fica mais compreensível através de um gráfico de pizza, uma lista ou um gráfico de barras?)

 **Lucas Cardoso** 02/11/2020

Quais tecnologias apresentam uma melhor relação entre performance e a coleta de métricas?

 **Vinicius Garcia** 05/11/2020

o legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?

 **Ramom Pereira dos Santos Silva** 03/11/2020

Uma plataforma de transparência sobre o uso da aplicação para os usuários e equipes de desenvolvimento.

Resuma em algumas linhas o que deverá ter o seu produto:

 **Ramom Pereira dos Santos Silva** 03/11/2020

Alta Prioridade: - Captura de Logs; - Armazenamento desses Logs; - Gerar visualizações dos Logs;

 **Ramom Pereira dos Santos Silva** 03/11/2020

Baixa Prioridade: - Funcionalidades mais utilizadas; - Serviços que apresentam mais falhas;

Quais as CERTEZAS que se têm do que será produzido?

 **Ramom Pereira dos Santos Silva** 03/11/2020

- Captura de todos os logs gerados na aplicação; - Utilização da Elastic Stack para captura de Logs; - Ambiente de execução no Kubernetes; - Kibana para visualização dos dados de transparência;

 **Ramom Pereira dos Santos Silva** 03/11/2020

- Dashboards de fácil entendimento para usuários comuns, que é o público alvo;

Quais as SUPosiÇÕES que se têm?

 **Ramom Pereira dos Santos Silva** 03/11/2020

- A comunicação entre os microserviços será capturada pela ferramenta de logs; - Adicionar novos microserviços sem reiniciar a aplicação;

 **Ramom Pereira dos Santos Silva** 03/11/2020

- Será possível extrair informações relevantes dos dados gerados; - Utilização desses dados

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

será possível extrair informações relevantes dos dados gerados, combinando esses dados para gerar valor para os clientes;

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

 **Ramon Pereira dos Santos Silva** 03/11/2020

- Como funciona a integração entre os componentes da Stack ELK? - A integração da Stack com os microserviços é simples? - O deploy no Kubernetes é simples? - Como extrair dados relevantes de uso da aplicação, apenas utilizando logs? - Quais informações são relevantes para armazenar nos logs?

 **Vinicius Garcia** 05/11/2020

o legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

Referências

Matriz CSD

<https://gobacklog.com/blog/matriz-csd/>

Links compartilhados

 **Ricardo Ebbers Carneiro Leão**

Matriz CSD

<https://github.com/ricardoebbers/distributed-transaction-saga/projects/1>

O que será produzido? E para quem?

 **Ricardo Ebbers Carneiro Leão** 02/11/2020

Um MVP de um orquestrador de transações distribuídas, exposto a partir de uma camada RESTful, que implementa o padrão SAGA e garante Atomicidade, Consistência, Isolamento e Durabilidade num contexto de microserviços. O público alvo desse software como serviço são empresas de tecnologia que já enfrentam problemas relacionados à transações distribuídas.

Resuma em algumas linhas o que deverá ter o seu produto:

 **Ricardo Ebbers Carneiro Leão** 02/11/2020

O produto deverá expor endpoints para configurar e executar Sagas. Cada Saga contém uma ou mais transações, cada transação possui um endpoint de execução e outro de compensação. Os endpoints de execução devem ser chamados em sequência pelo orquestrador e, caso ocorra uma falha numa dessas transações, os endpoints de compensação devem ser chamados em ordem reversa.

 **Ricardo Ebbers Carneiro Leão** 02/11/2020

O produto deve manter uma estrutura resiliente de persistência dessas chamadas e possibilitar o monitoramento a partir de logs bem formulados. Para cada transação deve ser passado um payload de contexto, a ser usado pelos serviços a serem chamados. Esse payload poderá ser mutado pelos serviços.

Quais as CERTEZAS que se têm do que será produzido?

 **Ricardo Ebbers Carneiro Leão** 02/11/2020

O produto a ser desenvolvido será um MVP

 **Ricardo Ebbers Carneiro Leão** 02/11/2020

Uma boa e fundamental prática em microserviços é a adoção do padrão "one database per service", que por consequência elimina a opção de utilizar transações nativas de banco de dados

  **Ricardo Ebbers Carneiro Leão** 02/11/2020

O padrão SAGA é um padrão de design comumente utilizado para solucionar o problema da

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

O padrão SAGA é um padrão de design comumente utilizado para solucionar o problema de transações distribuídas

RE Ricardo Ebbers Carneiro Leão 02/11/2020
Transações mal implementadas frequentemente são causas raiz de bugs

RE Ricardo Ebbers Carneiro Leão 02/11/2020
Transações num contexto de microserviços são difíceis de implementar

Quais as SUPosições que se têm?

RE Ricardo Ebbers Carneiro Leão 02/11/2020
É possível gerar valor suficiente para um MVP a partir da exposição de um serviço RESTful.

RE Ricardo Ebbers Carneiro Leão 02/11/2020
É possível fazer uma solução de transações distribuídas genérica suficiente para ser vendida como SaaS

RE Ricardo Ebbers Carneiro Leão 02/11/2020
É possível criar um produto completo em cima de um design pattern

RE Ricardo Ebbers Carneiro Leão 02/11/2020
Desenvolvedores e empresas de tecnologia tem interesse em contratar Software como Serviços para facilitar sua operação

RE Ricardo Ebbers Carneiro Leão 02/11/2020
Garantiremos que todos os endpoints serão chamados em sequência e ao menos uma vez cada, mas não garantiremos idempotência no MVP.

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

RE Ricardo Ebbers Carneiro Leão 02/11/2020
Quais dos princípios ACID são os mais fáceis de garantir num MVP?

RE Ricardo Ebbers Carneiro Leão 02/11/2020
É possível reduzir complexidade ao integrar uma solução terceira de orquestração de transações?

VG Vinicius Garcia 05/11/2020
o legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Matriz CSD

Métodos [de trabalho]

Método de trabalho que avalia o grau de certeza do que vai ser produzido

🔗 Referências

Matriz CSD
<https://gobacklog.com/blog/matriz-csd/>

O que será produzido? E para quem?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020
um MVP de micro serviços baseado em cadastro e persistência de dados em banco, utilizando front/backend e um API gateway para sua comunicação. Será produzidos para os iniciantes em micro serviços que desejam por a 'mão a massa'.

C Resuma em algumas linhas o que deverá ter o seu produto:

RI Renato Joaquim de Miranda Ferreira 03/11/2020

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

-Banco de dados simples -Front end -Back end -API Gateway

Quais as CERTEZAS que se têm do que será produzido?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020

Todos os pontos descritos no resumo do produto no questionamento anterior.

RJ Renato Joaquim de Miranda Ferreira 04/11/2020

Além de ser voltado ao desenvolvedor.

Quais as SUPosições que se têm?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020

Ainda não possuímos suposições do produto no momento, quadro deve ser alterado conforme o tempo passado durante do desenvolvimento do projeto.

Quais as DÚVIDAS que se têm e o que se deve perguntar nos testes?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020

Análises dos dados cadastrados no banco por meio de gráficos

VG Vinicius Garcia 05/11/2020

apenas essa? vale uma reflexão. o legal agora é tratar essas incertezas como riscos e aí pensar no que fazer para mitigar esses riscos

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020

Simplicidade e confiabilidade.

RJ Renato Joaquim de Miranda Ferreira 04/11/2020
desempenho

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

RJ Renato Joaquim de Miranda Ferreira 03/11/2020

Facilidade de implementação, confiabilidade que o dado será persistido da maneira correta e a confiabilidade que a integração deverá funcionar conforme é esperado.

recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agrega valor, não faz sentido, ou qualquer outro motivo)?

Nenhum comentário realizado

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software

<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos

<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura

<https://bit.ly/30CbnoE>

The Twelve-Factor App methodology

<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

 Renato Joaquim de Miranda Ferreira 03/11/2020

- Limitação da equipe em relação ao horário e técnica - Conhecimento das tecnologias usadas pode ser uma pedra no sapato - Viabilidade do projeto

Quais são os cinco principais objetivos de negócio para o sistema a ser projetado?

 Renato Joaquim de Miranda Ferreira 03/11/2020

- Facilidade no entendimento da aplicação - Guia para futuros desenvolvedores - Simplicidade na aplicação - Entregar o resultado esperado do sistema - Praticidade no seu uso

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

 Renato Joaquim de Miranda Ferreira 03/11/2020

Tentaremos contornar tanto a barreira técnica, no sentido das tecnologias usadas, quanto organizacional para conseguir entregar o produto.

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

 Renato Joaquim de Miranda Ferreira 03/11/2020

Além das possíveis dependências (kubernetes e/ou outras tecnologias que necessitem de uma instalação anterior), idealmente apenas o 'plug-n-play' baixado diretamente do repositório

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126

https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

RE Ricardo Ebbers Carneiro Leão 02/11/2020

Portabilidade, no sentido que o produto será genérico e adaptável a qualquer arquitetura distribuída; terá estabilidade funcional, visto que o cerne do produto será a implementação de um padrão de design amplamente conhecido (SAGA) e confiável, visto que garantirá os princípios ACID de transações num cenário distribuído.

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

RE Ricardo Ebbers Carneiro Leão 02/11/2020

Implementação de um orquestrador de SAGAS para solucionar transações distribuídas. Interface RESTful para facilitar a integração com backends web, observabilidade a partir de streams de logs orientados a eventos.

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agrega valor, não faz sentido, ou qualquer outro motivo)?

RE Ricardo Ebbers Carneiro Leão 02/11/2020

Entendemos que é importante que o sistema tenha uma interface e arquitetura assíncrona, onde o consumidor do serviço deveria poder escolher qual arquitetura (síncrona ou assíncrona, via REST ou mensageria) a partir da configuração de suas SAGAS. Porém, devido à complexidade adicionada em fazer essa implementação e ao fato que temos um prazo curto para entregar o MVP, optamos por atender apenas os cenários síncronos via REST.

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

🔗 Referências

Arquitetura de Software
<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos
<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura
<https://bit.ly/30CbnOE>

The Twelve-Factor App methodology
<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

RE Ricardo Ebbers Carneiro Leão 08/11/2020

Devido ao conhecimento atual da equipe, faremos o projeto utilizando Java

RE Ricardo Ebbers Carneiro Leão 08/11/2020

Embora a melhor opção para a arquitetura da solução seja utilizar soluções de mensageria, optaremos por comunicação síncrona via endpoints REST, também devido à capacidade técnica da equipe

RE Ricardo Ebbers Carneiro Leão 08/11/2020

Acreditamos que é possível criar um SaaS ao redor desse projeto, porém nos limitaremos ao MVP sendo um microserviço

C

Quais são os cinco principais objetivos de negócio para o sistema a ser projetado?

- RE Ricardo Ebbers Carneiro Leão 08/11/2020**
Eliminar problemas de transações inconsistentes num cenário de microserviços que aplicam o pattern "Database per service"
- RE Ricardo Ebbers Carneiro Leão 08/11/2020**
Reducir a complexidade de executar fluxos complexos transacionais num contexto de microserviços
- RE Ricardo Ebbers Carneiro Leão 08/11/2020**
Garantir consistência de dados sem gerar acoplamento entre os microserviços
- RE Ricardo Ebbers Carneiro Leão 08/11/2020**
Permitir rollback/compensação de transações de forma confiável
- RE Ricardo Ebbers Carneiro Leão 08/11/2020**
Gerar logs e insights das transações que ocorrem no sistema

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

Nenhum comentário realizado

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

Nenhum comentário realizado

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

🔗 Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

- RP Ramom Pereira dos Santos Silva 08/11/2020**
- Interoperabilidade - Usabilidade - Rastreabilidade

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

- RP Ramom Pereira dos Santos Silva 03/11/2020**
Interface amigável: facilidade de visualização do desempenho da aplicação para o cliente final, com métricas de desempenho, visualizações de dados relevantes, etc.
- RP Ramom Pereira dos Santos Silva 03/11/2020**
Observação de serviços: busca identificar quais serviços estão em maior uso e observar os mais relevantes para a aplicação.

RP

Ramon Pereira dos Santos Silva 03/11/2020

Recuperação de erros: rastreamento de requisições, identificando os erros que venham a ocorrer, para possíveis correções futuras.

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agrupa valor, não faz sentido, ou qualquer outro motivo)?

RP

Ramon Pereira dos Santos Silva 03/11/2020

Monitoramento da aplicação em termos de uso de recursos computacionais como CPU e Memória.

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software

<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos

<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura

<https://bit.ly/30Cbn0E>

The Twelve-Factor App methodology

<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

RP

Ramon Pereira dos Santos Silva 03/11/2020

Pouca experiência no uso de Logs e interfaces amigáveis para usuários comuns.

Quais são os cinco principais objetivos de negócios para o sistema a ser projetado?

RP

Ramon Pereira dos Santos Silva 03/11/2020

- Visualizações com dados de uso relevantes;
- Visão geral de utilização de toda a aplicação;
- Identificação dos gargalos da aplicação, em termos de serviços;
- Gerar dados relevantes para possíveis análises futuras;
- Rastreamento de requisições;

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

RP

Ramon Pereira dos Santos Silva 03/11/2020

Acredito que identificar quais informações são relevantes e quais dados armazenar nos logs deverão ganhar destaque. Além disso, quais dados mostrar aos usuários através de dashboards também é um ponto de destaque. Além de ter que lidar com configurações de ferramentas e execução.

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

RP

Ramon Pereira dos Santos Silva 03/11/2020

Os administradores do sistema precisam saber quais serviços e funcionalidades estão sendo usados em maior quantidade, quais estão apresentando falhas em maior quantidade.

RP

Ramon Pereira dos Santos Silva 03/11/2020

Os clientes precisam saber como está o seu desempenho na aplicação, o que está em maior uso ou não, estatísticas da utilização, transparência do desempenho.

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

 **Saulo Guilhermino** 04/11/2020

Operacionalidade, Interoperabilidade e Maturidade

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

 **Saulo Guilhermino** 02/11/2020

Coleta e visualização de métricas (Entrega observabilidade)

 **Lucas Barros** 04/11/2020

Centralização de atributos comuns aos serviços (autenticação, TLS, métricas), fazendo reuso e diminuindo a complexidade de cada serviço.

 **Lucas Barros** 04/11/2020

Rastreamento de requisições (observabilidade)

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agrega valor, não faz sentido, ou qualquer outro motivo)?

 **Lucas Barros** 04/11/2020

Proxy do API Gateway feito em algumas linguagens poderiam ser substituídos por proxies mais robustos (como Envoy) para maior segurança, confiabilidade e performance

 **Lucas Barros** 04/11/2020

Também acho que podemos descartar o logging estruturado (ex. EKS stack) porque logs podem ser vistos acessando o console da aplicação. Poderíamos substituir isso por tracing, que é algo a mais e dá uma visão interessante do sistema.

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software
<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos
<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura
<https://bit.ly/30CbnoE>

The Twelve-Factor App methodology
<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

 **LC** **Lucas Cardoso** 02/11/2020

O projeto deve ser open source

 **LB** **Lucas Barros** 04/11/2020

A organização ficará presa as tecnologias da ferramenta e terá que ter um certo conhecimento sobre elas

Quais são os cinco principais objetivos de negócios para o sistema a ser projetado?

 **LB** **Lucas Barros** 04/11/2020

- Tracing de requisões - Exposição de métricas relevantes do sistema - Menor fricção operacional

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

 **LC** **Lucas Cardoso** 02/11/2020

O projeto terá que lidar com diferentes tecnologias, e integrá-las de maneira pouco onerosa para o desenvolvedor

 **SG** **Saulo Guilhermino** 02/11/2020

O projeto terá de contornar possíveis impactos de performance negativos no sistema cliente

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

 **LB** **Lucas Barros** 04/11/2020

Métricas das requisições para cada serviço e dados de rastreamento de requisições também para cada serviço

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

 **EV** **Emerson Victor** 04/11/2020

Interoperabilidade Apreensibilidade Utilização de recursos

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

 Emerson Victor 04/11/2020

Centralização de logs (facilitação da análise de dados e detecção de problemas) Análise de dados (obtenção de insights) Monitoramento (criação de inteligência para gestão dos serviços)

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agrupa valor, não faz sentido, ou qualquer outro motivo)?

 Danilo Lira 04/11/2020

Nenhum

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software
<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos
<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura
<https://bit.ly/30CbnoE>

The Twelve-Factor App methodology
<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

 Danilo Lira 04/11/2020

A equipe não tem conhecimento na implementação de logs e monitoramento

Quais são os cinco principais objetivos de negócios para o sistema a ser projetado?

 Danilo Lira 04/11/2020

- Facilitar a análise de logs e detecção de problemas - Facilitar o processo de gestão de logs - Utilizar logs para gerar informações relevantes de negócios - Utilizar logs para informar sobre o funcionamento dos serviços - Visualização rápida e acessível das informações

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

 Danilo Lira 04/11/2020

- Coleta, armazenamento e gestão de informações de diferentes serviços - Interface developer-friendly entre projeto e o e-commerce

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

 Danilo Lira 04/11/2020

Os desenvolvedores - logs e dados de monitoramento dos serviços Os gestores - informações do e-commerce, geradas a partir dos logs

Atributos de Qualidade

Sistemas [de tecnologia]

Para um sistema de tecnologia ter sucesso, não basta apenas satisfazer os requisitos funcionais. Os sistemas críticos em geral devem atender também a segurança, confiabilidade, desempenho e outros requisitos semelhantes. Segundo a IEEE 1061, qualidade de software é o grau em que o software possui uma combinação desejada de atributos (por exemplo, confiabilidade, interoperabilidade)

Referências

Os atributos de qualidade na ISO/IEC 9126
https://pt.wikipedia.org/wiki/ISO/IEC_9126

Quality attributes in Software Architecture
<https://bit.ly/31pPWpB>

Quais são os três principais atributos de qualidade - do ponto de vista tecnológico - do sistema de tecnologia (exemplo de atributos de qualidade: mobilidade, segurança, desempenho, usabilidade, confiabilidade)?

 José Sheldon Brito 03/11/2020

Funcionalidade, usabilidade, portabilidade

 Igor Fernandes 03/11/2020

Usabilidade - Facilidade na configuração do load balancer

 Igor Fernandes 03/11/2020

Portabilidade - O monitoramento e configuração pode ser feita sem a necessidade alterar diretamente um arquivo de configuração, numa interface Web, ou seja acessível de qualquer lugar

Quais são os três principais recursos do sistema de tecnologia? O que estes recursos entregam de valor?

 Igor Fernandes 03/11/2020

Balanceamento de carga: Priorização dos serviços mais importantes para o sistema

 Igor Fernandes 03/11/2020

Monitoramento do serviços: Observar o uso dos microserviços a fim de identificar os serviços mais importantes

 Igor Fernandes 03/11/2020

Interface de gerenciamento: abstraindo a configuração de um load balancer, melhorando a usabilidade

Em se tratando de um sistema que já existe uma versão (mesmo que especificação ou projeto), quais recursos existem no sistema de tecnologia, mas não deveriam existir no momento (não agraga valor, não faz sentido, ou qualquer outro motivo)?

 Igor Fernandes 03/11/2020

O balanceamento de carga por prioridade já existe no mercado (Citrix por exemplo)

Arquitetura de Software

Sistemas [de tecnologia]

A estrutura de um sistema de software, que engloba componentes ou serviços de software, suas propriedades visíveis externamente, restrições e os relacionamentos e interações entre eles

Referências

Arquitetura de Software
<https://bit.ly/3cT8VwN>

Princípios SOLID: qualidade em programação em 5 conceitos
<https://bit.ly/2B4pto2>

Arquitetura de Software: Desenvolvimento orientado para arquitetura
<https://bit.ly/30CbnOE>

The Twelve-Factor App methodology
<https://12factor.net/>

Existem restrições técnicas, organizacionais, culturais, legais, etc. que vão impactar o projeto de arquitetura do software?

 **Igor Fernandes** 03/11/2020

Conhecimento limitado da equipe sobre load balancers

Quais são os cinco principais objetivos de negócio para o sistema a ser projetado?

 **Igor Fernandes** 03/11/2020

Construção de uma interface amigável que entregue para o usuário as principais funcionalidades do load balancer base

 **Igor Fernandes** 03/11/2020

Não obrigar o usuário a realizar alguma adaptação nos seus microsserviços para que nossa plataforma funcione corretamente. Nossa sistema deve ser "plug-and-play".

 **Igor Fernandes** 03/11/2020

Sugerir ao usuário perfis de prioridade baseado em métricas de uso dos microsserviços da aplicação

 **Igor Fernandes** 03/11/2020

Registrar serviços na plataforma sem a necessidade de definir uma instância em específico do serviço, permitindo assim um auto-scaler funcionar em harmonia com nossa plataforma

 **Igor Fernandes** 03/11/2020

Configurar diferentes load balancers do sistema na mesma plataforma

 **Igor Fernandes** 03/11/2020

Essa ideia surgiu a pouco, seria interessante que a plataforma pudesse ser aplicável a qualquer tipo de load balancer. Um lugar onde seria gerenciado os load balancers do sistema e suas configurações.

Quais restrições/considerações o projeto vai ter que lidar, mitigar ou contornar?

 **Igor Fernandes** 03/11/2020

Não será possível implementar todas as configurações do load balancer numa interface de usuário, portanto o usuário deverá ser capaz de editar o arquivo de configuração manualmente caso ele queira um comportamento mais específico

Quais são os dados que os stakeholders, individualmente, precisarão obter da aplicação?

 **Igor Fernandes** 03/11/2020

Nível de uso dos microsserviços

 **Igor Fernandes** 03/11/2020

Árvore de dependência dos microsserviços

 **Igor Fernandes** 03/11/2020

Serviços registrados no load balancer

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O

Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

👉 Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado



O que foi feito (atividade - responsável)?

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?



Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

C O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

👉 Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como

lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>



Análise post-mortem de projetos: o que é e como fazer

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?



Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

C

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.



Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

█ Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado



Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são

rquentemente consideradas um componente-chave e um precursor contínuo ao gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

C

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?



Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

C O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um

<https://strateegia.digital/dashboard/analysis/5f3edbb952c20932f6597649>

projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

C

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Links compartilhados

 **Saulo Guilhermino**

Repositório do Projeto

<https://github.com/microservice-2020-3/observable-api-gateway>

 **Saulo Guilhermino**

Vídeo-demonstração (recomendo)

<https://drive.google.com/file/d/1D6-2wAp27TiU5-zRD5Kbo4INb6R2huH4/view?usp=sharing>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?



Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado



Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

Referências

Post-mortem documentation

https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence

<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer

<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante foi aprendido nesta iteração, seja de carácter individual e/ou coletivo)?

Nenhum comentário realizado

Post-mortem

Métodos [de trabalho]

O Método de [trabalho] post-mortem é um processo, geralmente executado na conclusão de um projeto, para determinar e analisar os elementos do projeto que foram bem ou mal sucedidos. O

Conjunto de conhecimentos em gerenciamento de projetos (PMBOK) se refere ao processo como lições aprendidas. As "autópsias" do projeto têm como objetivo informar as melhorias do processo que mitiguem os riscos futuros e promover as melhores práticas iterativas. As "autópsias" são frequentemente consideradas um componente-chave e um precursor contínuo do gerenciamento de risco eficaz.

🔗 Referências

Post-mortem documentation
https://en.wikipedia.org/wiki/Postmortem_documentation

Nine steps to IT post-mortem excellence
<https://www.zdnet.com/blog/projectfailures/nine-steps-to-it-post-mortem-excellence/1069>

Análise post-mortem de projetos: o que é e como fazer
<http://bit.ly/38dhML8>

Início e Término da Iteração (xx/xx/yyyy a xx/xx/yyyy)

Nenhum comentário realizado

O que estava planejado (atividade - responsável)?

Nenhum comentário realizado

O que foi feito (atividade - responsável)?

Nenhum comentário realizado

O que não foi feito (Levante o que não foi realizado e o seu respectivo impedimento)?

Nenhum comentário realizado

O que está planejado para próxima iteração?

Nenhum comentário realizado

Lições aprendidas (Levante o que foi aprendido nesta iteração, seja de caráter individual e/ou coletivo)?

Nenhum comentário realizado

💡 Síntese IF1007

Métodos [de trabalho]

Método [de imersão] para documentar informações na conclusão visando identificar se a jornada foi SUFICIENTE para o estudante entender o contexto e EFICIENTE para ser usada como base para o processo de ENSINO e APRENDIZAGEM na disciplina

🔗 Referências

Desenvolvimento de Aplicações Nativas da Nuvem com Arquitetura Baseada em Microservices
<https://bit.ly/vcg-microservices>

A sua EXPECTATIVA para esta jornada de aprendizagem foi atendida? Justifique com exemplos, se possível.

Nenhum comentário realizado



Qual área dentro da computação te interessa atualmente (que você pretende se especializar, construir carreira)?

Nenhum comentário realizado

Após o inicio da disciplina, você se envolveu ou está envolvido em algum projeto de construção de software (aplicativo, aplicação, plafaforma, sistema...)?

Nenhum comentário realizado

Como você lidou com os PONTOS CRÍTICOS que poderiam interferir no sucesso do seu aprendizado?

Nenhum comentário realizado

Qual CENÁRIO você vê na sua formação depois desta disciplina?

Nenhum comentário realizado