CRIADO PELA FERNANDA MAKI HIROSE

arquitetura em mensageria: temos uma aplicação web ou mobile, elas têm comunicações através de http, esse serviço tem um proxy de entrada e realizam serviços que vão gerar uma mensagem. esses serviços podem ser uma comunicação com uma API, que gera uma mensagem, a desvantagem de centralizar uma mensagem em um único message broker é só ter dependência dele para todos os serviços, o pattern para usar com essa arquitetura é o event source

como gerenciar erros com arquitetura mensageira?
- dead letter queue (filas de re-tentativas): se a mensagem não foi concluída com sucesso vai ser colocada em uma fila e ser tentada de novo no futuro
- monitoramento entre serviços: é necessário monitorar os serviços e rastrear todos os pontos de onde chegou a mensagem
- rastreamento de fluxo: é rastrear o ponto inicial de onde chegou a mensagem, para rastrear um fluxo em uma arquitetura de mensagens assíncronas é usar um metadata nos logs e indexá-los em um único lugar

big data: é um conjunto muito grande de dados, estruturados ou não, não há um banco de dados específico, mas sim uma composição de várias origens.

ferramentas em Analytics em BI: Tableu, Sap Business Objects, Microsoft Power BI, IBM Cognos Analytics

OLTP: On-Line Transaction Processing, são dados voláteis, passíveis de modificação e exclusão.

OLAP: On-Line Analytical Processing, dados voláteis, passíveis de modificação

dados semi-estruturados: possuem uma estrutura de organização, mas é flexível, podem também ser utilizados substituindo banco de dados estruturados

ferramentas big data: Spark, DataTorrent, Storm, Hadoop

Power BI: é importante para decisões estratégicas da organização, possui diversas ferramentas completas ou parciais, é dimensionada de acordo com o que deseja investir a os requisitos computacionais da solução

BI: é composto por ferramentas, estruturas, profissionais (corpo técnico), dados

dimensões em Data Warehouse: tempo, geografia e membros

banco de dados NoSQL: Redis, Cassandra, CouchDB, MongoDB

dados não-estruturados: twitter, wordpress, linkedin, facebook

data lake: é um repositório que centraliza e armazena todos os tipos de dados gerados pela e para a empresa.

Baas: significa backend como serviço, um provedor Baas é o Firebase
Paas: significa plataforma como serviço (auto scale on the go), proporciona um meio escalar a infraestrutura de forma automatizada, exemplos: Terraform, Cloud Formation
arquitetura não serverless: o melhor jeito para obter alta disponibilidade em cloud é utilizar um orquestrador de containers com kubernetes
Terraform: é uma ferramenta de receita provisionamento

cloud computing: é a solução de servidores por um provedor, um provedor de cloud computing é o Amazon Web Services, ele é um gerenciamento de hardware/software, provedores de servidor e armazenamento, Iaas/Paas/Baas

serverless: é um modelo de execução de computação no qual o provedor de nuvem executa o servidor e gerencia dinamicamente a alocação de recursos da máquina, exemplos: API Gateway, Amazon Kinesis, AWS Lambda, Amazon S3

ferramentas para deploy: Azure Pipelines, Circleci, App Veyor, Gitlab CI, Travis CI, Jenkins

ferramentas para operação: Kubernetes, Rancher, Microsoft Azure, Puppet, Terraform, CHEF, Open Shif, AWS

ferramentas para coletar métricas: DATADOG, Prometheus, App Metrics, Rollbar, ZABBIX, New Relic, Pushover, Seq, Monitis

IaaS: significa infraestrutura como serviço (hardware e internet), exemplos: Google Cloud, Amazon Web Services, Microsoft Azure

devops: é um conjunto de práticas que integram e automatizam os processos entre as equipes de desenvolvimento, operações e de apoio para a produção rápida e confiável de software

os 3 caminhos: flow (otimização do fluxo, via eliminar desperdícios, gargalos no processo, transferência de responsabilidades e tempos de espera é trilhado entre a demanda e a entrega em produção), feedback (visam responder os problemas o quanto antes, testando tudo, alertando em qualquer falha), learning (visa gerar conhecimento através da experimentação)

integração contínua: tem etapas de construção, teste unitário, análise da qualidade do seu código, e empacotamento, ela gera o artefato, ela não faz a implantação em produção. implantação contínua: o deploy é feito de modo automático. entrega contínua: ela necessita de um aprovador para que a aplicação seja implantada em produção. principais ferramentas: circleci, app vayor, azure pipelines, travis ci, jenkins, gitlab ci

inspeção contínua: principais ferramentas: sonarqube, code climate, codady, servem para resolver problemas como descobrir se seu código está ficando mais complexo ou não, complexidade ciclomática: quantos caminhos independentes o seu código pode seguir em determinado pedaço da sua aplicação, consegue descobrir códigos duplicados, code smell: é uma senha para colocar no seu código, ele dá um histórico do débito técnico, tem cobertura de testes, apresenta métricas, valida pull-requests

principais etapas para a produção de software em conjunto entre desenvolvimento e operações são: planejamento, codificação, teste, release, deploy, operação e monitoramento

serviços web: possibilita a integração de aplicações desenvolvidas com tecnologias diferentes e se comunicam através do protocolo HTTP, facilitam integrações e reutilizações de implementação. um serviço web pode ser chamado de API.

REST: Representational State Transfer, é uma arquitetura que define a implementação de um serviço web, utiliza métodos HTTP para representar a operação a ser realizada em um determinado recurso. JSON e XML podem ser usados em REST.

estrutura SOAP MESSAGE: SOAP envelope (encapsula a mensagem) > SOAP header (possui informações de atributos e metadados da requisição) > SOAP body (contém os detalhes das mensagens)

API: Application Programming Interface, é um conjunto de rotinas e padrões disponibilizados por uma aplicação

código de estado:
1xx: informativo
2xx: sucesso
3xx: redirecionamento
4xx: erro do cliente
5xx: erro do servidor

XSD: é um schema XML usado para definir a estrutura de dados do SOAP message

SOAP: Simple Object Access Protocol, é um protocolo baseado em troca de mensagens XML, para acessar serviços web, principalmente por HTTP, permite integrações independentes de plataforma e software

WSDL: é utilizado para desenvolver um serviço web SOAP

XML: Extensible Markup Language, facilita a separação de conteúdo, não tem limitação para criar tags, linguagem comum para integração entre aplicações

principais métodos HTTP: get (solicita a representação de um recurso), post (solicita a criação de um recurso), delete (solicita a exclusão de um recurso), put (solicita a atualização de um recurso)

JSON: Javascript Object Notation, formatação utilizada para troca de mensagens entre sistemas, usa uma estrutura chave-valor e também listas ordenadas.

prós - monolito: baixa complexidade, monitoramento simplificado
contra - monolito: stack única, compartilhamento de recursos, acoplamento, mais complexidade e escalabilidade

prós - microsserviços: stack dinâmica, simples escalabilidade
contra - microsserviços: acoplamento, monitoramento mais complexo, provisionamento mais complexo

prós - microsserviços 2: stack dinâmica, simples escalabilidade, desacoplamento
contra - microsserviços 2: monitoramento mais complexo, provisionamento mais complexo

gerenciamento de erros - onde é mais complexo:
- processos assíncronos (microsserviços 2)
- pipeline

- solução: dead letter queue
- filas de re-tentativas

acoplamento: lida com dependências ou não de uma parte em relação a outra, portanto o modelo de microsserviços possui menor acoplamento se comparando ao monolito

microsserviços: desenvolvem sistemas mais flexíveis, escaláveis e com fácil manutenção comparado aos sistemas tradicionais

gerenciador de pipeline: recebe a aquisição do proxy HTTP e conforme a etapa da requisição envia para um serviço específico e independente

message broker: identifica o motivo de um determinado serviço ter sido incapaz de responder no momento de uma requisição

MCUs: microcontrolador de chip único, sistema operacional real time, embarcado, uso industrial, médico, militar, transporte

raspberry pi: computador completo, hardware integrado em uma única placa, roda SO Linux ou Windows, uso doméstico e comercial

MQTT: base na pilha do TCP/IP, protocolo de mensagem assíncrona (M2M), criado pela IBM para conectar sensores de pipelines de petróleo a satélites, padrão OASIS suportado pelas linguagens de programação mais populares

QoS 0: nível mínimo de menor esforço, sem garantia de entrega, mensagem não é retransmitida
QoS 1: garante que a mensagem foi entregue no mínimo uma vez ao recebido, mensagem não pode ser retransmitida se não houver confirmação da entrega
QoS 2: é a mesma coisa que o nível QoS 1 porém mais elevado ainda, com maior segurança

cloud: TBs e PBs de informações, potencial de escala global

prova de conceito: app android, eclipse mosquito, node.js, banco de dados mysql
mínimo produto viável: gps embarcado, hivemq, akka scala jvm, banco de dados nosql
solução: gps embarcado, aws iot core, aws kinesis firehose, aw3 s3
iot na prática: gps embarcado - aws iot core - aws data stream - aws lambda - aws elasticcache redis - aws elasticcache redis - aws ec2 - feathersjs backend - dashboard

DDL: Data Definition Language
DML: Data Manipulation Language
SGDB: um sistema que gerencia o banco de dados