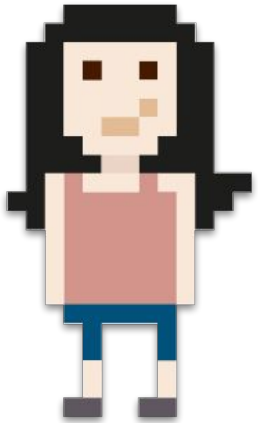
A Saga da consistência de dados entre Microservices

sensedia

Andreia Silva

andreia.silva@sensedia.com

Tô zen. Tô plena.
~~Mentira, tô nervosa pra caralho!~~

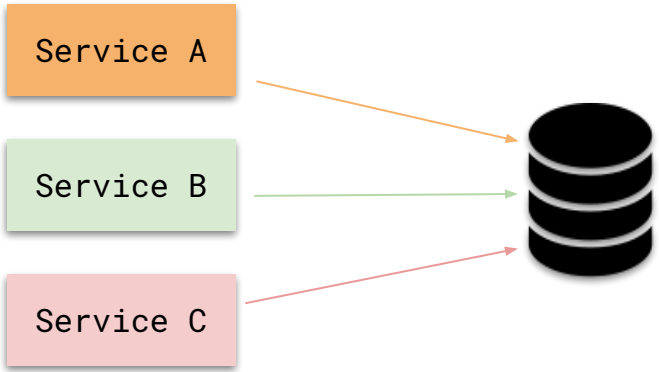


Como tudo começou

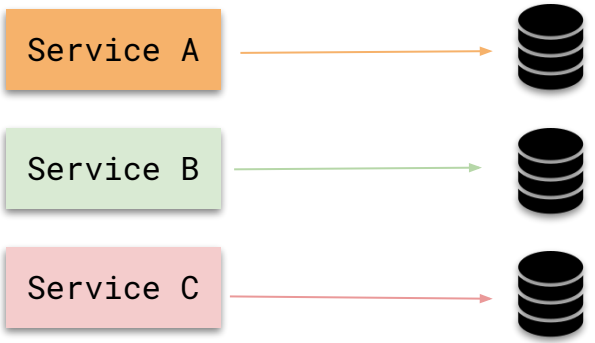
O crescente uso de Sistemas Distribuídos, nos levou a tratar nossos dados de forma diferente.

Como tudo começou

- Shared Database “Pattern”



- Database per Service Pattern



Chris Richardson Mod ➚ David Gonzalez Shannon • 2 months ago

This pattern is more of an anti-pattern. It's best to use the Database per Service pattern.

Bases individuais, problemas resolvidos?

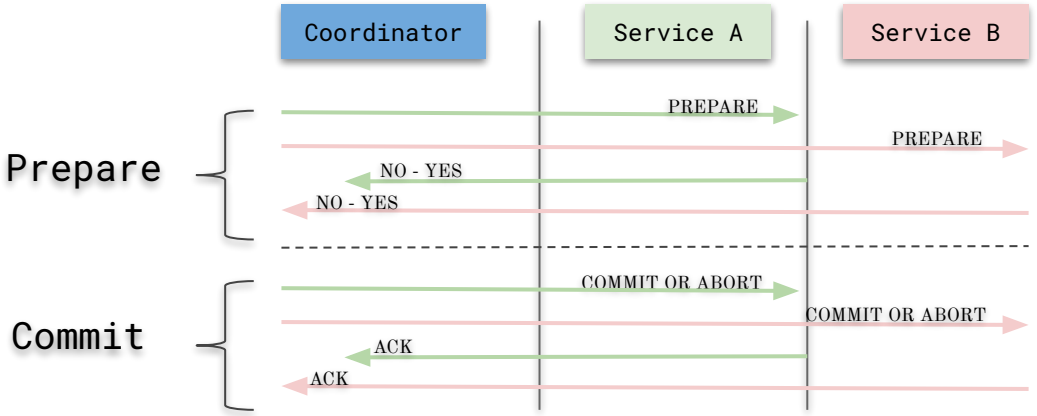
- O gerenciamento de múltiplas bases é + complexo
- Adeus **ACID** properties

Como lidar com transações que envolvem **mais de um** serviço?

Two-phase Commit - 2PC

*Uma velha conhecida (**e não recomendada**) prática.*

- É um protocolo utilizado para implementar e coordenar **transações distribuídas**
- Requer um coordenador global que mantenha o ciclo de vida da transação
- Possui duas fases:
  - Preparação / Votação
  - Decisão: Commit ou Abort?



O Starbucks não implementa 2PC! - *por Gregor Hohpe, Diretor Técnico na Google*

Two-phase Commit: Vantagens e Desvantagens

- É considerado um protocolo de Consistência Forte

- Síncrono, bloqueante
- O coordenador pode ser um ponto único de falha
- Impacto de performance
- Todos os componentes envolvidos na transação devem dar suporte a transações distribuídas (XA)


Consistência Eventual
O meio-termo.

- BASE (**B**asically **A**vailable, **S**oft State, **E**ventual Consistent)

- Estados intermediários
  - Não fortemente consistentes

- Garanta que as inconsistências sejam **temporárias**
  - E que as janelas de inconsistência não durem por muito tempo, Amém 

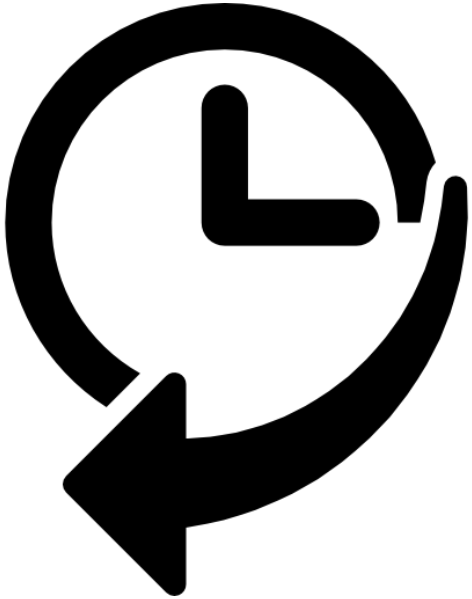
Event Sourcing

Um conjunto de peças, que unidas, remontam um estado.

- Pattern de armazenamento de dados

- O estado da entidade não é armazenado
  - Ao invés disso, armazenamos um histórico de eventos.
  - Para obter o estado atual da entidade, é necessário um **replay**

- Não existem updates
  - Um evento é um fato e um fato não pode ser modificado



- Mas, quando devemos pensar em usar Event Sourcing?
  - Quando é importante saber **como** aquela entidade ou agregado chegou naquele estado.

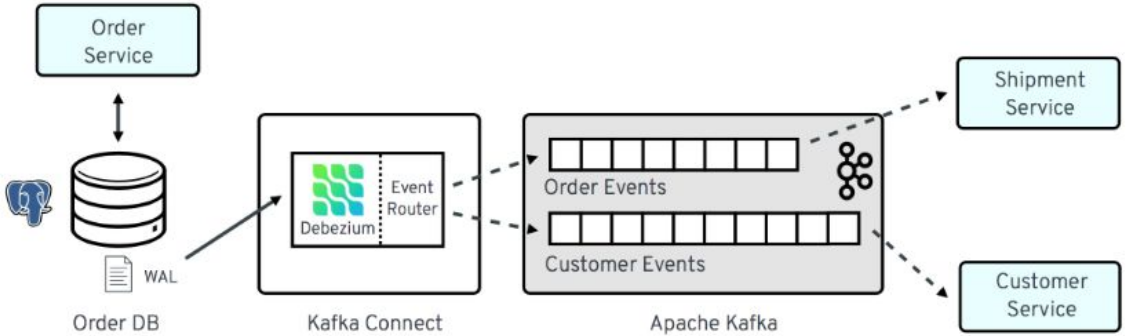
- Exemplo de armazenamento convencional:

#	id	nome	is_minha_namorada
1	1	Jenifer	0

- Com Event Sourcing:

event_id	aggregate_id	event_type	event_date
1	1	EncontreiElaNoTinder	2019-07-18
2	1	PediEmNamoro	2019-07-18
3	1	NãoAceitouOPedido	2019-07-19

Event Sourcing + Log Tailing



Id	AggregateType	AggregateId	Type	Payload
ec6e	Order	123	OrderCreated	{ "id" : 123, ... }
8af8	Order	456	OrderDetailCanceled	{ "id" : 456, ... }
890b	Customer	789	InvoiceCreated	{ "id" : 789, ... }

Outbox Table

Fonte: https://debezium.io/blog/2019/02/19/reliable-microservices-data-exchange-with-the-outbox-pattern

Event Sourcing: Vantagens e Desvantagens

- Performance - uma vez que tenho somente operações de **escrita**
- Audit Log
- Dados podem ser utilizados para gerar valor
- Debug

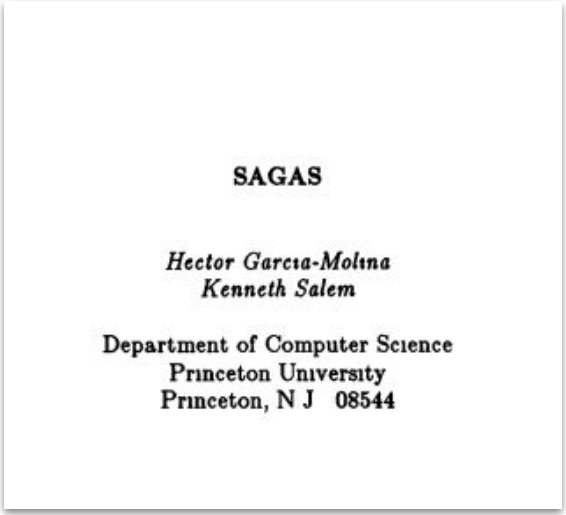
- Complexidade
- Eventos também representam um histórico de decisões ruins
- Tratamentos para ignorar eventos repetidos, executá-los na ordem, etc

Saga

Um padrão de gerenciamento de falhas.

- Esse padrão foi descrito por Hector Molina em 1987, não sendo inicialmente idealizado para ser aplicado em Sistemas Distribuídos

- Criado para lidar com transações longas



SAGAS

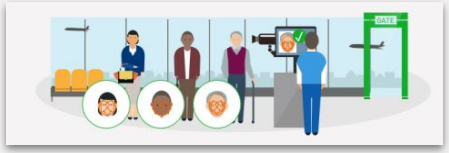
Hector Garcia-Molina
Kenneth Salem

Department of Computer Science
Princeton University
Princeton, N J 08544

- Uma saga representa uma coleção de **sub-transações locais**
  - Comunicação síncrona ou assíncrona

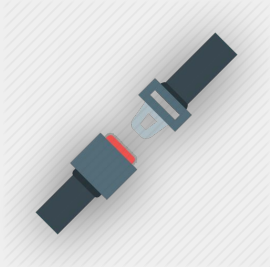
Saga - Caso de Sucesso

Boarding Service



Passenger On Board

Passenger Security Service



Passenger Safe

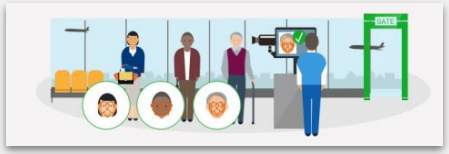
Airplane Check Service

Passenger On Flight!

- Cada sub-transação deve possuir uma **transação de compensação**
  - A transação de compensação **desfaz** o que foi feito na sub-transação local

Saga - Caso de Falha

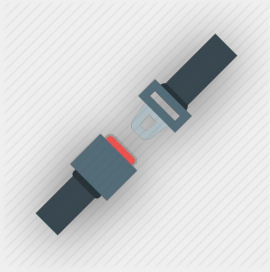
Boarding Service



Passenger On Board

Passenger Landed!

Passenger Security Service



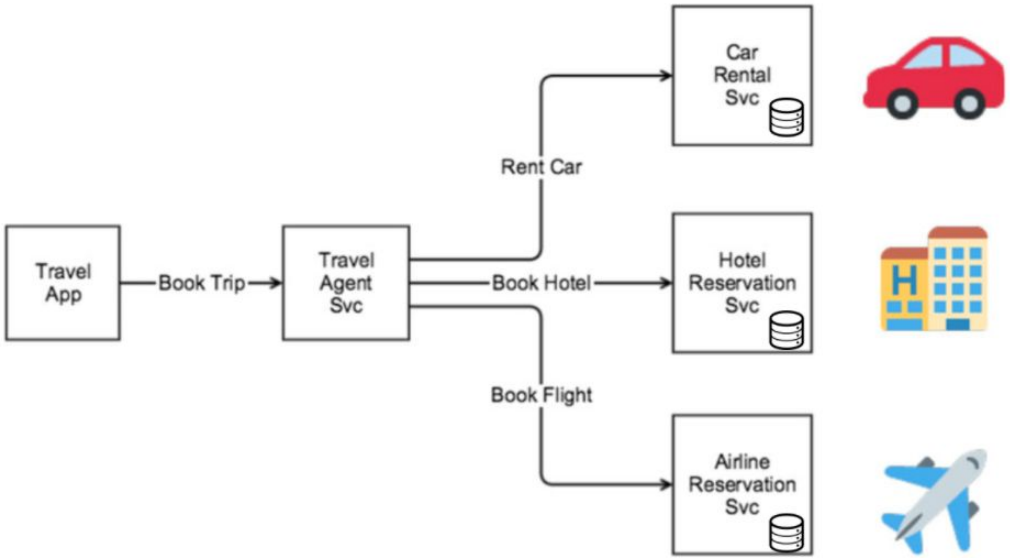
Passenger Safe

OpenDoors

Airplane Check Service

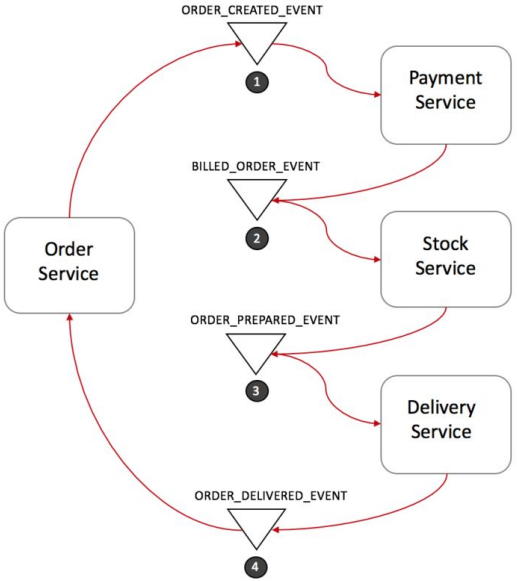
Flight Canceled

- **Orquestrada**: Neste caso, temos um orquestrador que conhece os participantes e coordena a execução da saga.



Fonte: https://www.youtube.com/watch?v=xDuwrtwYHu8

- **Coreografada**: Um participante invoca o outro ao final de sua própria transação através de eventos normalmente.



Fonte: https://blog.couchbase.com/saga-pattern-implement-business-transactions-using-microservices-part/

Saga coreografada ou orquestrada?

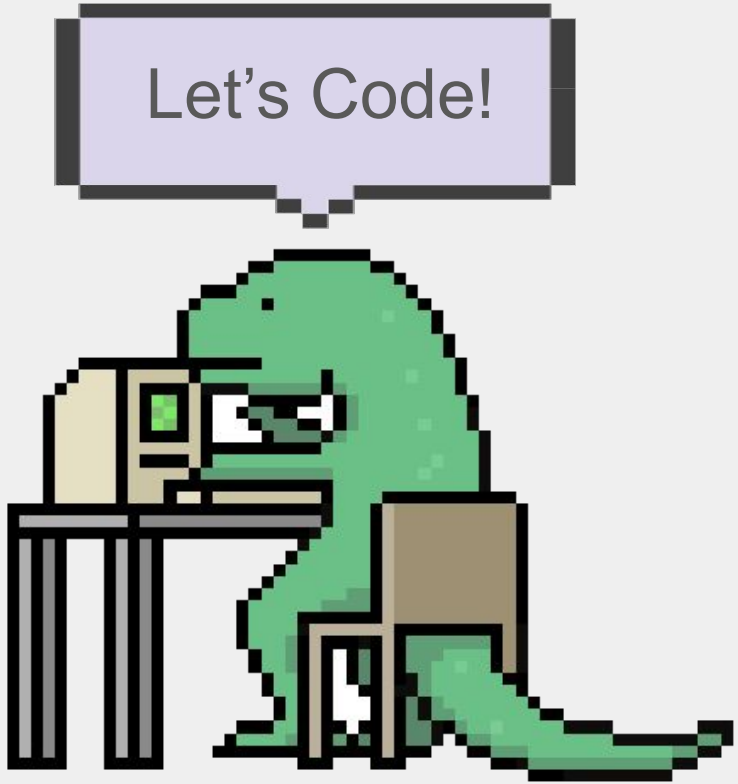
“Tu te tornas eternamente responsável pela gambiarra que codificas”

- *A Pequena Dev*

Saga: Vantagens e Desvantagens

- Sem transações distribuídas
- Integridade referencial dentro do serviço, garantida por **ele mesmo**
- Integridade referencial fora do serviço, garantida pela **aplicação**

- Complexidade (*again?!*)
- Perda de Isolamento: concorrência



https://github.com/andreiac-silva/camel-saga-demo.git

Importante!

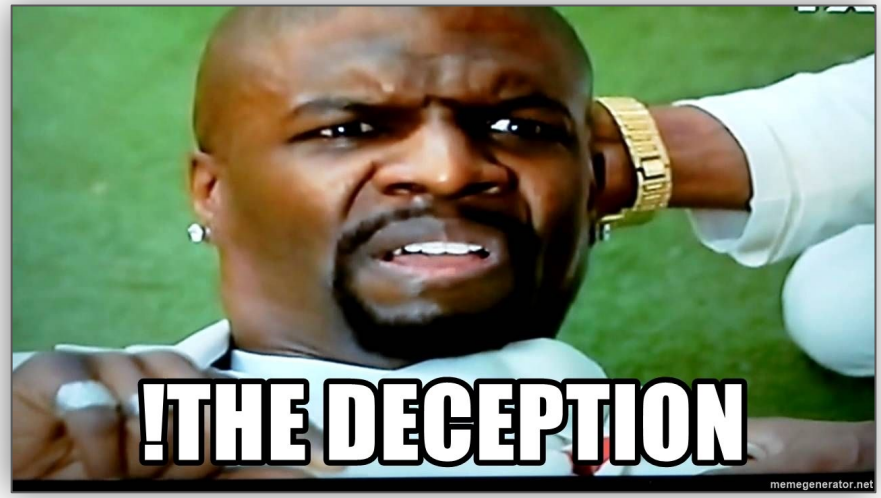
LRA coordinator can be scaled?

reply from Ondrej Chaloupka in *Narayana* - View the full discussion

---

Hi silva.andreiacamila,

thank you for asking. Unfortunately the LRA is still under development and we have not implemented the failover and scalability capabilities yet. Currently the Narayana LRA is a single node implementation where failover needs to be done manually.
Nevertheless we are aware of the importance of this functionality, we have this on our roadmap and we expect to address it soon. Just I can't tell you exact time frame for it.



Existem ferramentas que auxiliam na implementação do que vimos?

eventuate

AxonIQ

APACHE Camel

camunda
the business process company

AWS Step Functions

fn

Obrigada!

@andreiacamila

andreia.silva@sensedia.com

Links para outras demos interessantes:

https://github.com/andreiac-silva/camunda-saga-demo

https://github.com/yosriady/serverless-sagas/tree/master/functions

https://github.com/xstefank/axon-service

https://github.com/xstefank/eventuate-sagas