Sagas

Victor Osório

Agenda

Quando é necessário?

Vamos definir ACID?

Vamos definer Saga?

Quais os tipos de Saga?

Como implementar?

Quando é necessário?

Quando é necessário?

Transações de Longa Duração

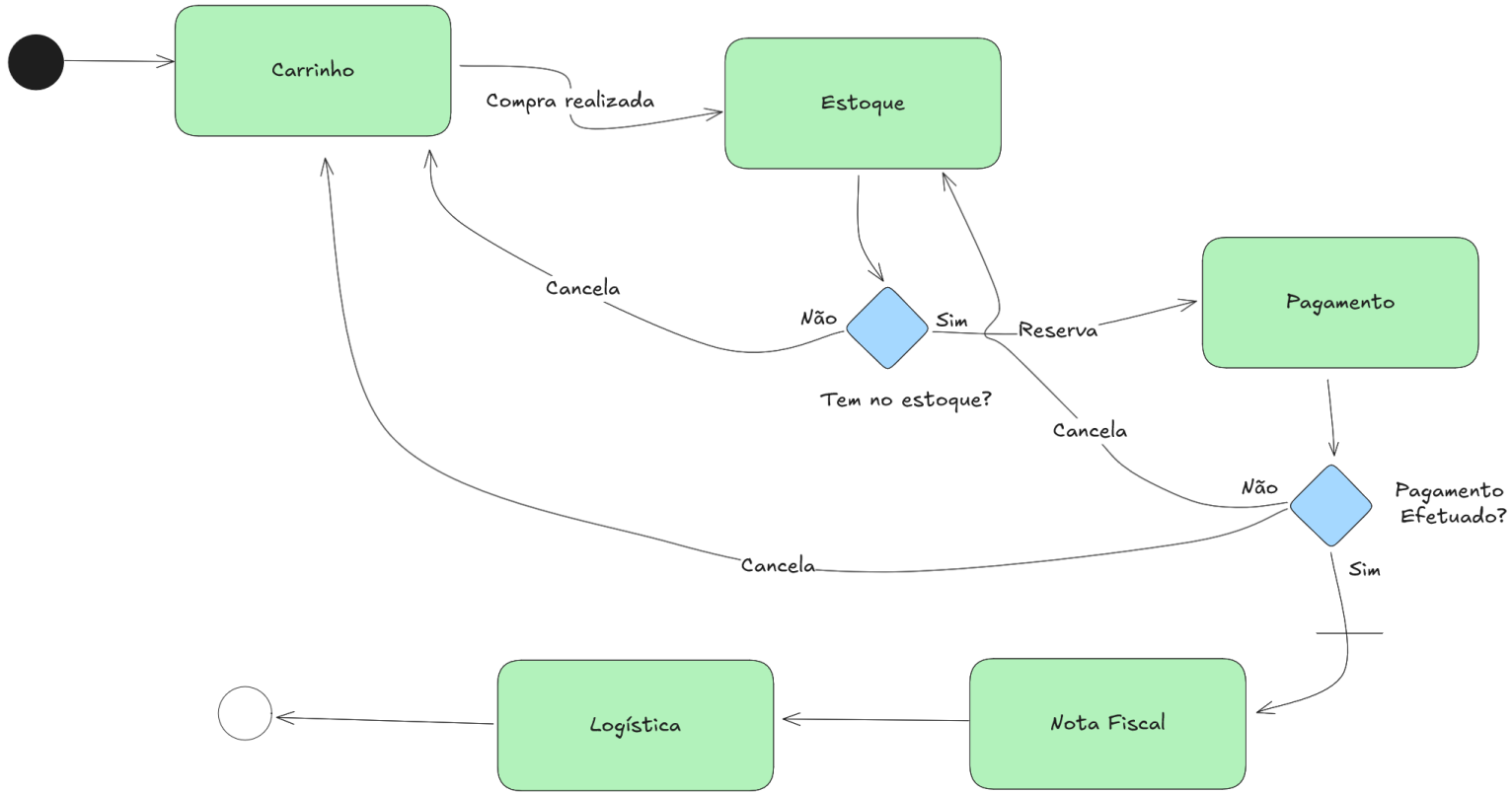
- *Long Lived Transactions*
- Podem durar horas ou dias

Quando é necessário?

Requisitos Complexos

- Retentativas no processo
- Dependências de sistemas externos

Transações de Longa Duração?!?!?!



Quando é necessário?

- ACID não é uma solução!!!!
  - ❌ ~~Atomicidade~~
  - ✔️ Consistência
  - ❌ ~~Isolamento~~
  - ✔️ Durabilidade

Vamos definir ACID?

Atomicidade

- Todas as operações são definidas como uma única operação.
- Tudo ou nada.

Atomicidade (em Sagas)

- Todas as operações são definidas como uma única operação.
- Tudo ou nada.

Mas em uma Saga

- Não é garantida!
- Ao invés de uma operação temos um processo, uma máquina de estados!

Consistência

- Os dados estão em um estado consistente antes e depois da operação.
- Se refere aos dados! Se a operação for cancelada, tudo tem que ser desfeito!

Consistência

- Os dados estão em um estado consistente antes e depois da operação.
- Se refere aos dados! **Se a operação for cancelada, tudo tem que ser desfeito!**

Consistência (em Sagas)

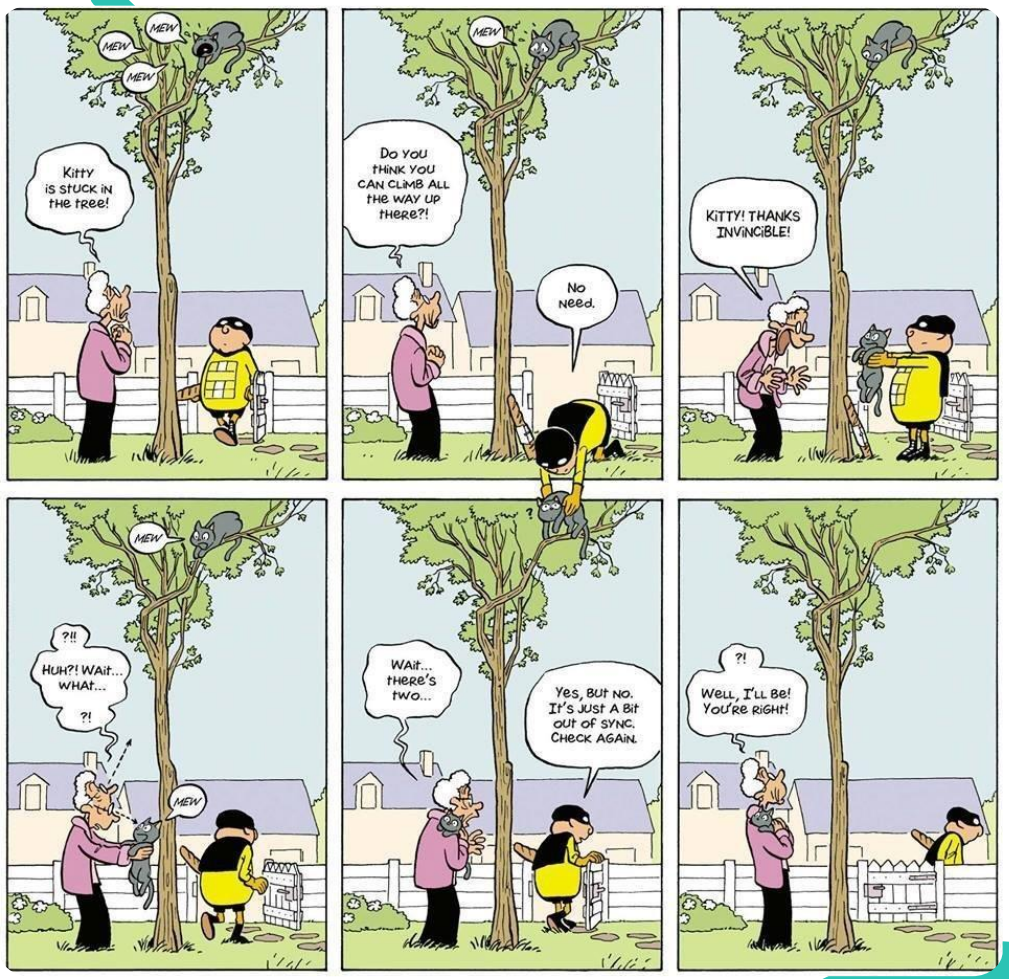
- Os dados estão em um estado consistente antes e depois da operação.
- Se refere aos dados! **Se a operação for cancelada, tudo tem que ser desfeito!**

Mas em uma Saga

- É garantida, mas a Consistência Eventual

Consistência

- No final o processo é consistente
- Mas pode haver inconsistências durante o processo.



Isolamento

- O estado intermediário de uma transação é invisível para outras transações.
- Serialização (*colocadas em ordem*) de transações

Isolamento (em Sagas)

- O estado intermediário de uma transação é invisível para outras transações.
- Serialização (*colocadas em ordem*) de transações

Mas em uma Saga

- Impossível de se garantir!

Durabilidade

- Após a transação ser concluída, as mudanças persistem e não são desfeitas

Durabilidade (em Sagas)

- Após a transação ser concluída, as mudanças persistem e não são desfeitas

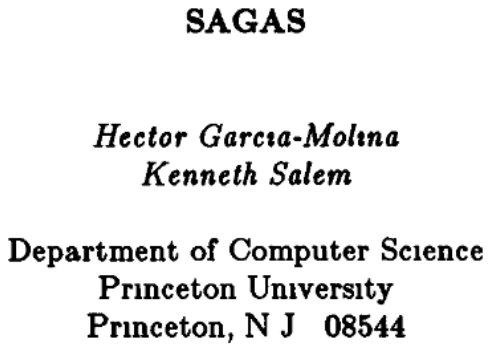
Mas em uma Saga

- O processo precisa ser corretamente modelado
- Transações de compensação são necessárias

O que são?

Onde surgiram?

- Definidas por Hector Garcia-Molina
- Solução para Transações de Longa Duração (Long Lived Transactions)
- Alternativas?
  - Segundo Microservices.io 2PC 🧐

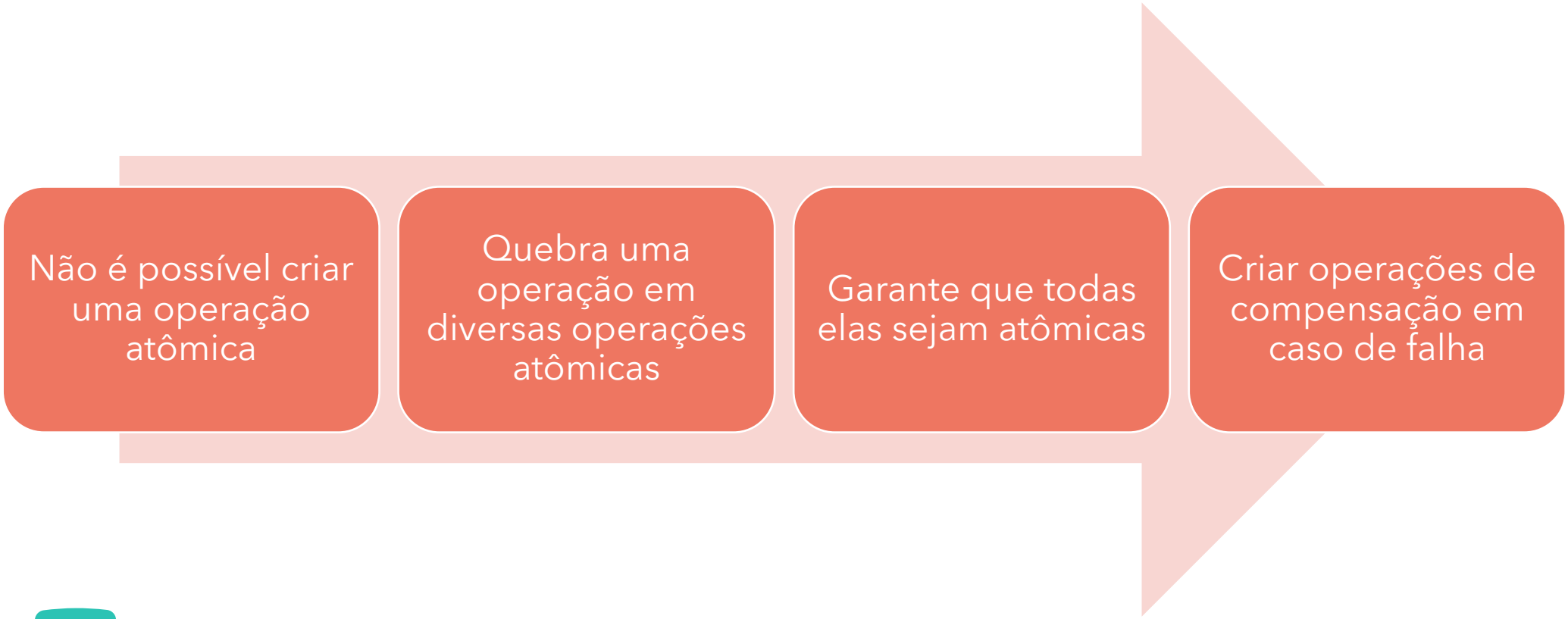


SAGAS

Hector Garcia-Molina
Kenneth Salem

Department of Computer Science
Princeton University
Princeton, N J 08544

A Idéia


Não é possível criar uma operação atômica
Quebra uma operação em diversas operações atômicas
Garante que todas elas sejam atômicas
Criar operações de compensação em caso de falha

A Idéia

- Relaxar os requisitos
  - Atomicidade
  - Consistência
  - Isolamento
- Garantir os requisitos
  - Atomicidade Global
  - Consistência eventual
  - Durabilidade

Tipos de Transação

- Transação Saga
- Transação de Compensação
- Transação Pivô
- Transação Recuperável

Tipos de Transação

Cada **Transação Saga** deve ser provida com uma **Transação de Compensação** para ser executada em caso de falha. Exceto se ela for após o ponto de não retorno.

Tipos de Transação

Transação Pivô define o ponto de não retorno da Saga. É a última transação a fornecer **Transação de Compensação**.

Tipos de Transação

Transação Recuperável é uma transação que mesmo que aconteça uma falha ela pode ser refeita ou recuperável.

Quais os tipos de Saga?

Tipos de Sagas

Paralelismo	• Serial • Paralela
Coordenação	• Orquestradas • Coreografadas

Tipos de Sagas

- Orquestrada Serial
- Orquestrada Paralela
- Coreografada Serial
- Coreografada Paralela

Saga Serial

- As transações são alinhadas no tempo
- Para um mesmo recurso, existe apenas uma transação em execução em determinado momento

Saga Paralela

- As transações podem ser executadas em concorrência
- Para uma mesmo recurso, pode existir mais de uma transação em execução em um determinado momento

Saga Orquestrada

- Existe um processo que coordena todas as execuções
- Esse processo deverá inicar e finalizar todas as transações e transações de compensação.

Desvantagens

- Lógica de negócio centralizada

Saga Orquestrada

- Existe um processo que coordena todas as execuções
- Esse processo deverá inicar e finalizar todas as transações e transações de compensação.

Benefícios

- Dependências Simples
- Menos acoplamento
- Mais isolamento de responsabilidades

Saga Coreografada

- Não há um processo que coordena as transações
- Cada processo deve identificar quando iniciar uma transação

Benefícios

- Simplicidade
- Menos acoplamento

Saga Coreografada

- Não há um processo que coordena as transações
- Cada processo deve identificar quando iniciar uma transação

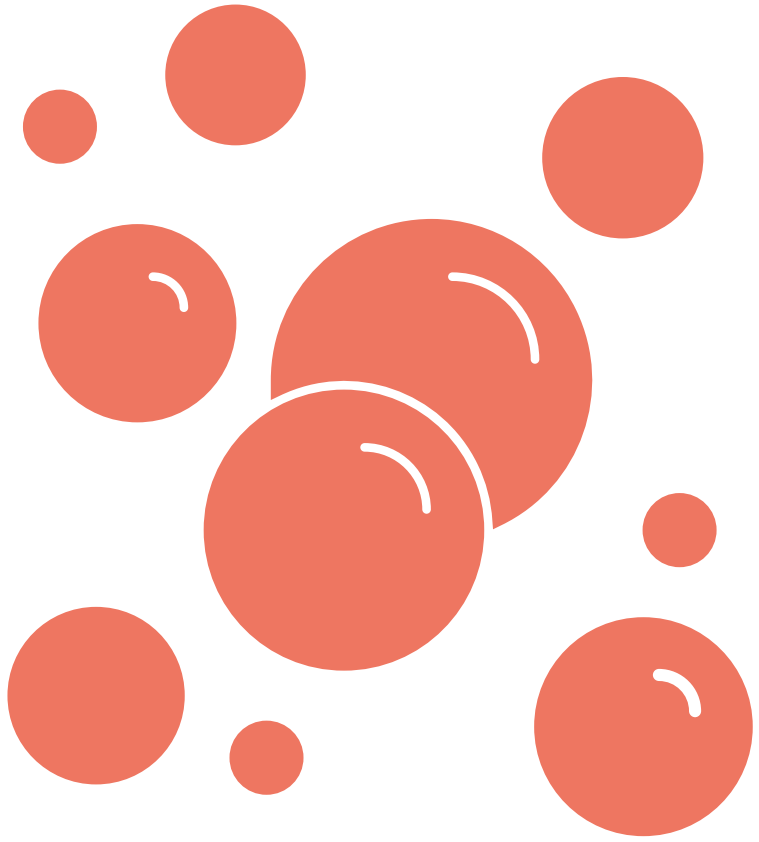
Desvantagens

- Maior complexidade
- Dependências ciclicas entre os serviços

Como implementar?

Como implementar?

- Baseada em Eventos
  - Garantir semântica Exactly-Once
  - Recomendável para sistemas mais complexos
  - Necessário o uso de Middleware Orientado a Mensagens
- Baseada em Máquina de Estados
  - A entidade deve ter um campo status
  - Recomendável para sistemas mais simples
  - Necessário o uso de base de dados comum



Dúvidas?

- @vepo