DISTRIBUTED TRANSACTIONS AND THE SAGA PATTERN

MASTER THESIS

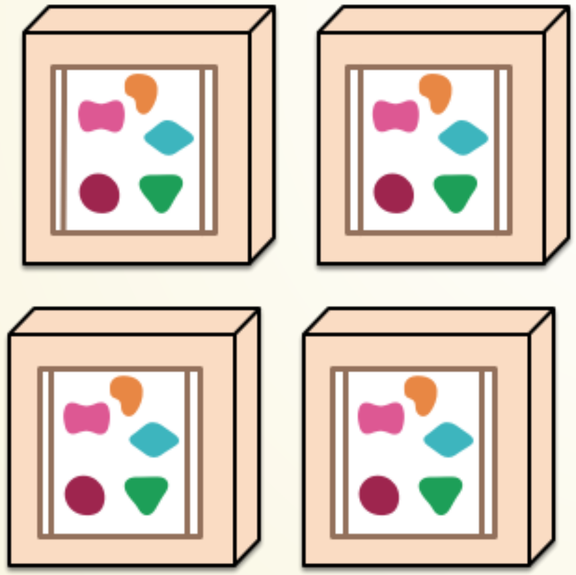
Use of Transactions within a Reactive Microservices Environment

https://issues.jboss.org/browse/JBTM-2920

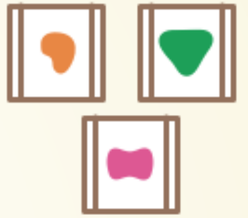
MICROSERVICES ARCHITECTURE

A monolithic application puts all its functionality into a single process...

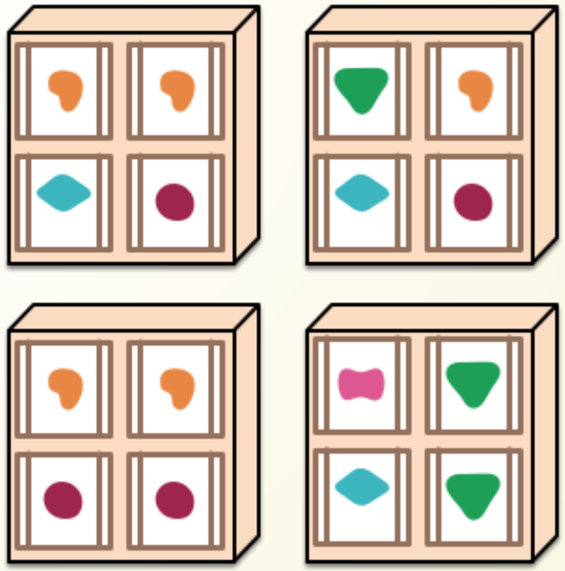
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...



... and scales by distributing these services across servers, replicating as needed.



ACID TRANSACTION

- Atomicity
- Consistency
- Isolation
- Durability

DISTRIBUTED TRANSACTIONS

- Two phase commit protocol
  - O(n^2) messages
  - blocking
  - coordinator - single point of failure

SAGA PATTERN

Hector Garcia-Molina and Kenneth Salem, Princeton Unversity, 1987

- long lived transactions
- compensations
- eventual consistency

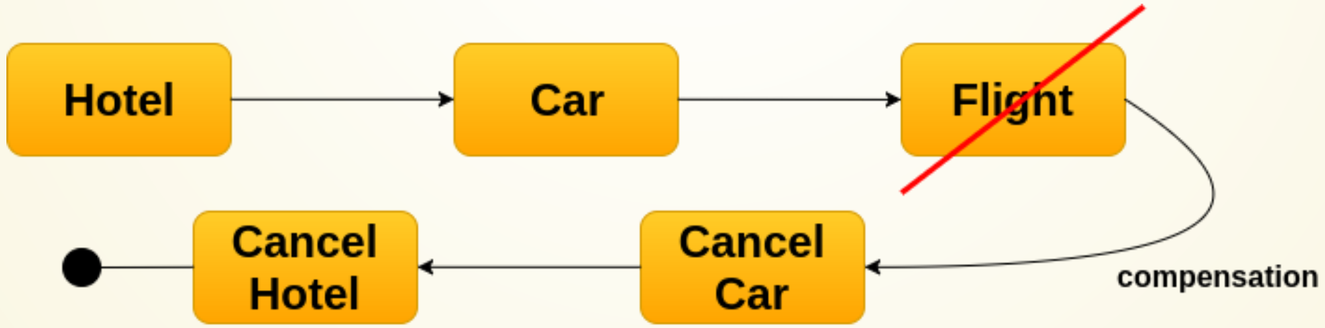
SAGA EXECUTIONS

- 2PC - T
- Saga
  - success - $T_1, T_2, T_3, \ldots, T_n$
  - compensation - $T_1, T_2, \ldots, T_k, C_k, C_{k-1}, \ldots, C_1$

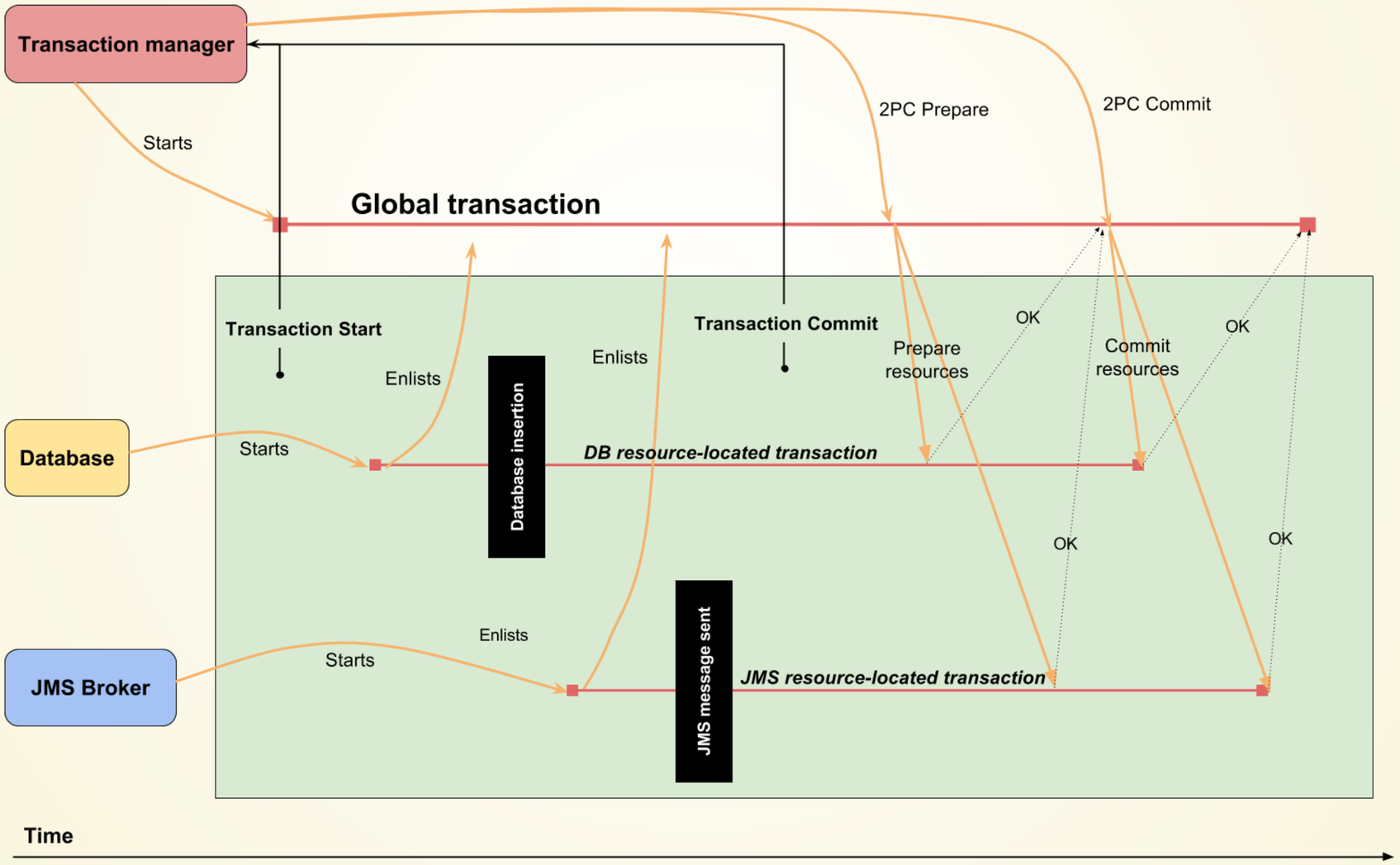
EXAMPLE SAGA

Success

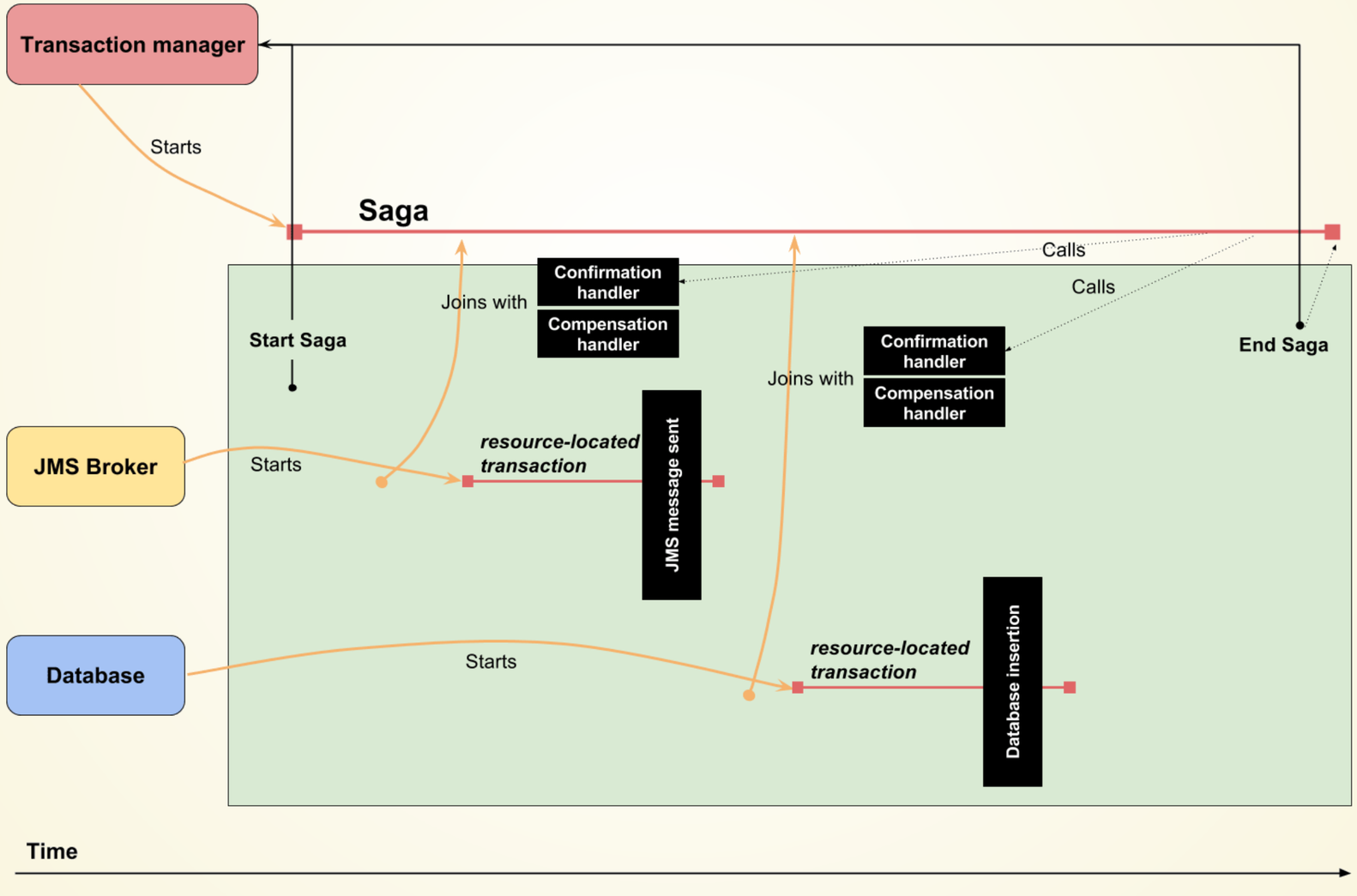
Failure / Compensation



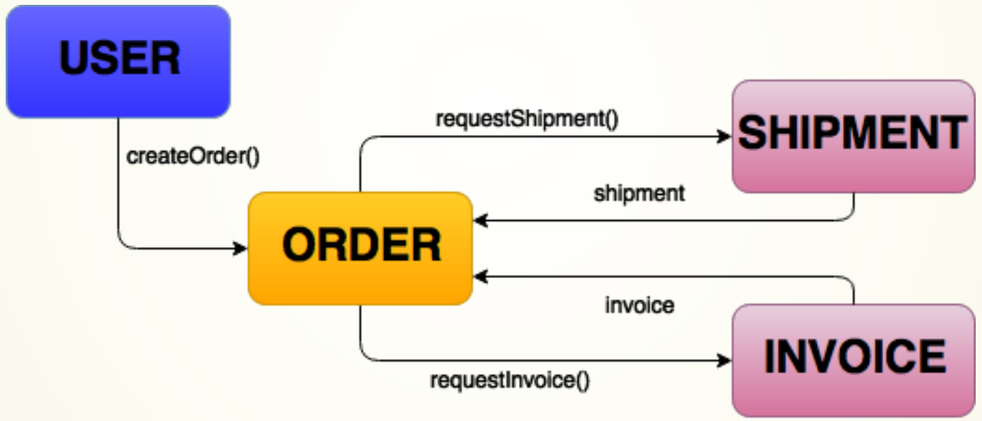
TWO PHASE COMMIT PROTOCOL



SAGA PATTERN



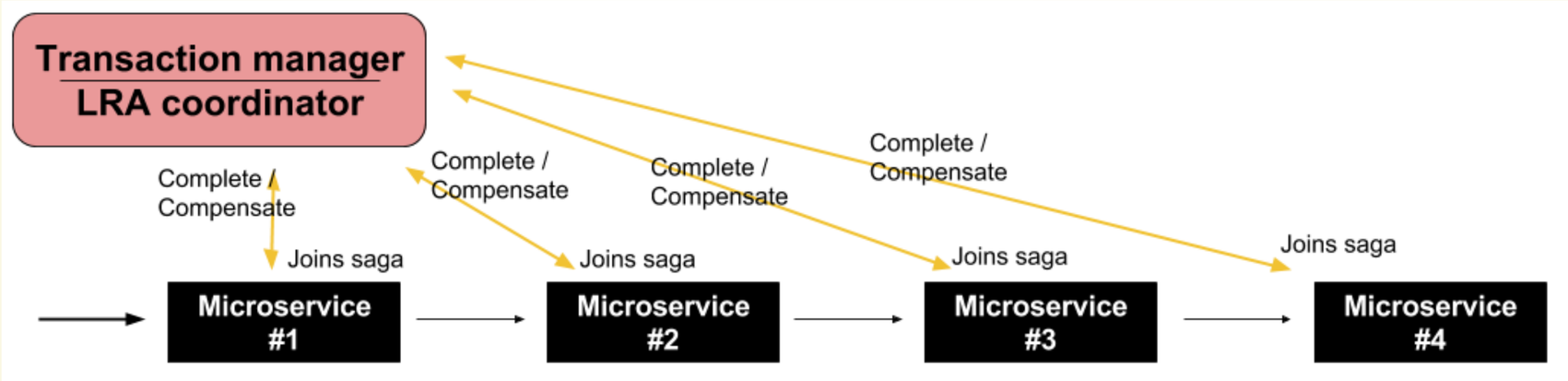
SAGA INVESTIGATION EXAMPLE



SAGA IMPLEMENTATION INVESTIGATIONS

- Axon framework
- Eventuate.io
- Narayana LRA
- Eventuate Tram
- blog post

LRA EXECUTOR MOTIVATION



LRA EXECUTOR EXTENSION

- proof of concept / prototype
- LRA definition
- asynchronicity and scalability
- recovery capabilities
- protocol/platform independent
- https://github.com/xstefank/lra-executor-extension

SAGAS

- long lived transactions
- distributed environment
- compensations and failure recovery
- atomicity vs availability

QUESTIONS

THANK YOU FOR YOUR ATTENTION

mstefank@redhat.com

@xstefank

http://www.slideshare.net