Masaryk University
Faculty of Informatics

Use of Transactions within a Reactive Microservices Environment

Master's Thesis

Martin Štefanko

Brno, Fall 2017

Masaryk University
Faculty of Informatics

Use of Transactions within a Reactive Microservices Environment

Master's Thesis

Martin Štefanko

Brno, Fall 2017

Replace this page with a copy of the official signed thesis assignment and the copy of the Statement of an Author.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Martin Štefanko

Advisor: Bruno Rossi, PhD

Acknowledgement

thanks

Abstract

abstract

Keywords

transactions, Narayana, JTA, reactive, microservices, asynchronous, saga

Contents

1 Introduction

2 Transactions

This chapter introduces the basic concepts of transactions, their properties and common problems of the management of transactions across multiple nodes in the distributed systems.

2.1 Atomic transaction

An atomic transaction is an unit of processing that provides all-or-nothing property to the work that is conducted within its scope, also ensuring that shared resources are protected from multiple users [1]. It represents an unified and inseparable series of operations that are either all provided or none of them take effect.

Every transaction is associated with a transaction coordinator or transaction manager which is responsible for the control and supervision of the participants performing individual operations. It also administer the comprehensive result of the transaction. Applications are commonly required only to contact the transaction manager about the start of the transaction.

The transaction can end in two forms: it can be either *commited* or *aborted*. The commit determines the successful outcome - all operations within the transaction have been performed and their results are permanently stored in a durable storage. The abort means that all performed operations have been undone and the system is in the same state as if the transaction have not been started.

2.1.1 ACID properties

2.1.2 Two phase commit protocol

2.2 Distributed transactions

A distributed transaction is the transaction performed in a distributed system. The distributed system consists of a number of independent computers connected through a communication network. Such systems are liable to the frequent failures of individual participants or communication channels between them.

The transaction manager can be implemented as a separate service or being placed with some participant or the client. **TODO**

2.3 Saga pattern

A saga, as described in the original publication [2], is a long lived transaction that can be written as a sequence of transactions that can be interleaved with other transactions. Each operation that is a part of the saga represents an unit of work that can be undone by the compensation action. The saga guarantees that either all operations complete successfully, or the corresponding compensation actions are run for all executed operations to cancel the partial processing.

2.3.1 Participants

3 Microservices architecture pattern

4 Transactions in distributed environments

5 Communication patterns

5.1 2PC

5.2 CQRS

5.2.1 Axon framework

5.2.2 Eventuate.io

6 Conclusion

Bibliography

[1] M. Little, J. Maron, and G. Pavlik, *Java transaction processing*. Prentice Hall, 2004.

[2] H. Garcia-Molina and K. Salem, "Sagas," *ACM SIGMOD Record*, vol. 16, no. 3, pp. 249–259, 1987.