ADBMS Project Design Document

Aayush Agrawal (aka7919), Saksham Bassi (sb7787)

October 2022

Introduction

In order to gain a deep understanding of Distributed systems and their design, we aim to work on building a distributed database. In this work, we would implement a distributed database model which supports the following features:

- Multiversion concurrency control
- Deadlock detection
- Replication
- Failure recovery

In the following sections, we will discuss some important modules of the project.

IO Manager

IO Manager has two main responsibilities, they are,

- Read the input file and translates the requests into transactions.
- Outputs the transaction details to the standard output.

It coordinates with the transaction manager and transaction modules.

Transaction Manager

There are three kinds of transactions, and the following scenarios for these are as follows.

- **Write transaction:** Firstly the locks on all the variables in all the available sites should be acquired. Once the locks are acquired, the value can be written to all those sites.
- **Read transaction:** It needs to acquire a read lock on the desired variable. If unavailable, it reads from another site. If no site is available from where the read lock can be acquired, the transaction gets pushed to the wait queue.

- **Read Only transaction:** To accommodate RO transactions, the Multiversion concurrency algorithm will be implemented. The transaction need not acquire a lock. However, it can read the desired variable only from those sites where it was committed before the transaction began. Additionally, the site must not have failed or shutdown since that commit time.

  If no site is available from where the requested variable can be read as per the protocol above, the transaction is aborted.

The transaction manager also has access to the list of transactions and the timestamps for the same. It uses these details to implement **cycle detection** to resolve the deadlock. In case of a deadlock, the transaction that has the latest timestamp (amongst the ones in the cycle) is aborted.

Site

The Site module contains information about a given Site. Each Site maintains a Lock Table along with containing details such as id, the status of the site, and data values. The Site module has functionalities such as initializing all the sites with the data, dumping a site, retrieving the status of a site, activating a site, shutting down a site, and acquiring and releasing locks. The initialize functionality implements the database with all the details based on the given initial instructions.

Transaction

The Transaction module (enum) is closely linked with the Transaction Manager module. It contains important details about a transaction such as its id, instruction, the id of the site, variable, and value.

Lock

The Lock module contains an important attribute Lock Table. The Lock module is linked to the Site module because each Site has a Lock module of its own. Some of the functionalities of the Lock module are to acquiring the lock and releasing the lock. Locks are acquired by providing details of transaction, variable, and type of lock. Release lock functionality requires a variable as a parameter. The Lock module also retrieves the lock type of the lock on a given variable.

Instruction

This is an enum class for the instruction types expected in the input. They are, begin, beginro, fail, end, dump, and recover. They are a part of the transaction detail.

UML Design

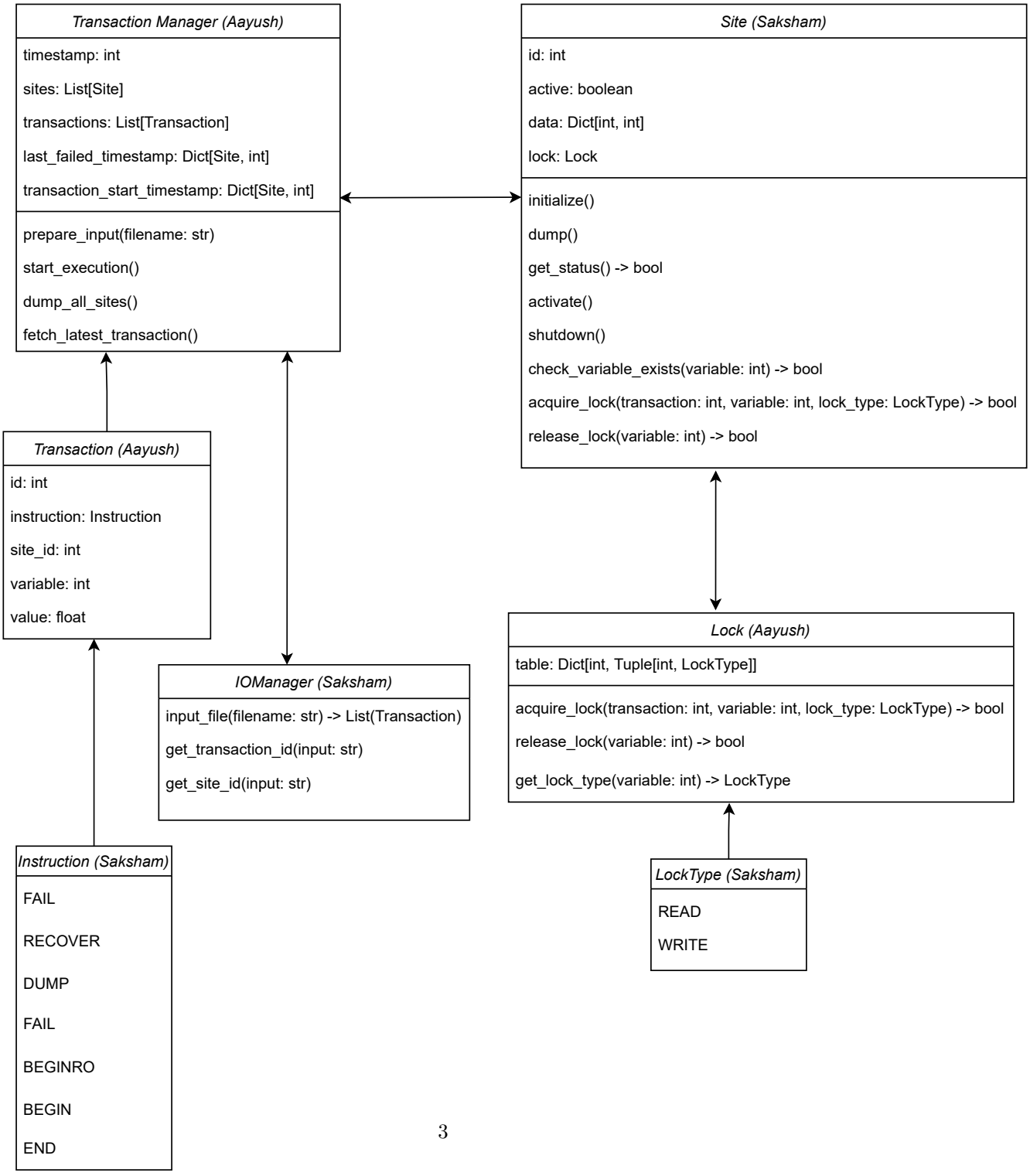

Figure 1: UML Design