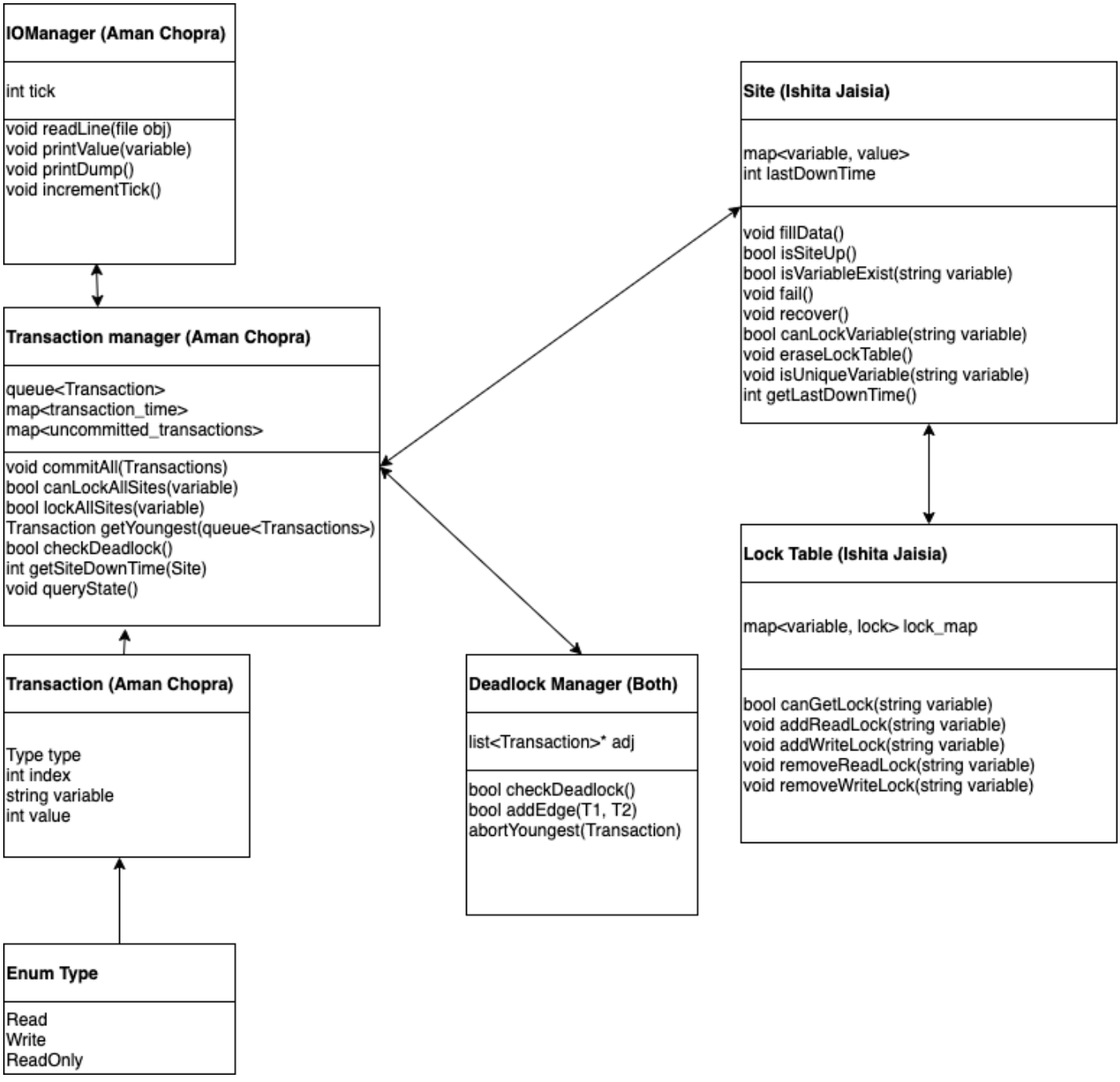
Design Document

Authors:
Aman Chopra(ac8511)
Ishita Jaisia(ij2056)



There are two major components, those are the Transaction Manager and the Site which is our Data Manager. I will try to give a brief description of each component and their interaction with each other.

IOManager: This component will take care of the input and output. It will read the lines which are our transactions and the test user input from the file and print the output to the console. This component maintains the tick value, and is also responsible to dump the data of all the variables of all the sites to the standard output. This communicates with the Site and the Transaction Manager.

Transaction Manager: This is the most important module which communicates among all the sites. To implement the available copies algorithm, we need to maintain a queue of transactions as to which operation of a transaction can get the locks and hence go through, otherwise we push the transaction to the wait queue. For write transactions, we need to get the locks on all sites where the variable is present and then write the value. For read only transactions , we will implement multi-version read consistency algorithm and hence we need not require any locks but we need to check if the site was down after the last commit and before accessing that variable. For read transactions we need the read locks. The transaction manager is also responsible to abort the transaction in case the site is down or the site goes down before the transaction ends. This component also communicates with the Deadlock Manger to check if there is a deadlock and resolve it. The transaction Manager will communicate with the Transaction class which will maintain the state of a single transaction and will be using some enumerations to get the status of the transaction.

Deadlock Manager: This module is responsible for detecting any deadlocks and resolving them. We will be using a cycle detection algorithm in a directed graph to implement this. First, we will create a wait for graph and keep on adding edges. At the beginning of the tick, if we detect a cycle in the wait for graph, we will abort the youngest transaction.

Site: The Site is also our data manager. This will maintain the state of a particular site. Since we have 10 sites, we will have 10 site objects. Each site will have its own lock table and maintain the state of its own variables. First, we will need to fill the variable values as per the instructions given in the question. There are functions like fail in which case we will clear the lock table, we also have functions like recover and dump. Every site should be able to say whether a variable is present in that or not, whether the variable is unique or not. It should also maintain what all variables are locked in that site and when was its last down time in order to implement multi-version read consistency.

Lock Table: The lock table is unique to a site. It maintains the lock state of all the variables of a particular site. It has functions to add read lock and add exclusive locks. It also has functions to remove those locks when the transaction ends according to the strict two phase locking algorithm. The state of the lock table is important to know whether the next operations of a transaction can go through or they have to wait or abort in case of a deadlock.