Advanced Database Systems: CSCI-GA.2434-001 Fall 2018
Project: Replicated Concurrency Control and Recovery
Design Document

Fatima Mushtaq (N15760010)

Basic Architecture:

Replicated Concurrency Control and Recovery mechanism for transactions is implemented **following the 'No undo, Redo' scheme for Recovery and 2-Phased Locking Protocol for Concurrency Control**. That is changes are written to database only if transactions is able to commit. Transaction log is maintained for each operation and upon commit, from log the transaction actually executes on the database.

A single threaded application is made simulating transaction processing using various data structures. A Site is simulated as a struct holding the variables data structure and local lock table as specified. Programming Language is C++.

There major program modules are as follows:

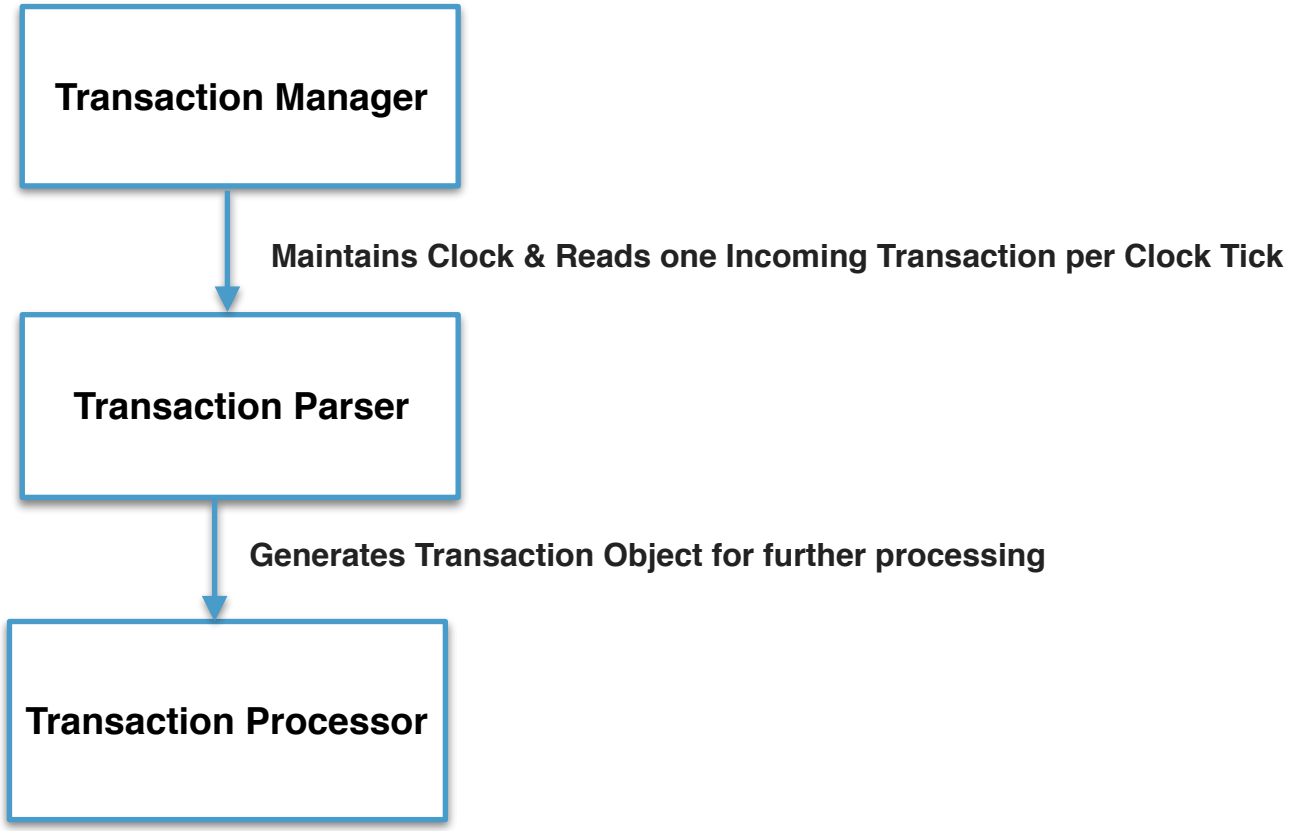
Single Transaction Manager:

A single Transaction Manager is broken down further into smaller modules based on functionality which are as follows:

- **Module 0 - Initialization Module:**
  This is to initialize all the data structures with initial or default values as the program starts. A C++ Struct Site is used to hold the variables and a lock-table per site. There's **no global lock table.** The odd variables are non replicated and even variable are replicated as per specifications.

- **Module 1 - Transaction Parser:**
  As transactions are read from a file or Standard Input, a parser is written to interpret the transaction string read from file and give semantics to it and generates a **Transaction (c++ struct)** object. Regular expressions are used to identify transactions into each category : (BEGIN, BEGINRO, READ, WRITE, END, FAIL, RECOVER, DUMP=8)

- **Module 2 - Trigger Process Trans Simulation:**
  After parsing the transaction, as it receives a Transaction Object with it's operation, the transaction manager then initiates processing the transaction using 2-Phase Locking and Multiversion Concurrency.



<u>Transaction Processor/Simulator</u>:

A transaction as it begins is added to a **Transaction Queue.** As the operations of the transaction are received: Read/Write Transactions are processed via **2-Phase Locking** and Read-Only are processed via **Multiversion Concurrency Scheme.**

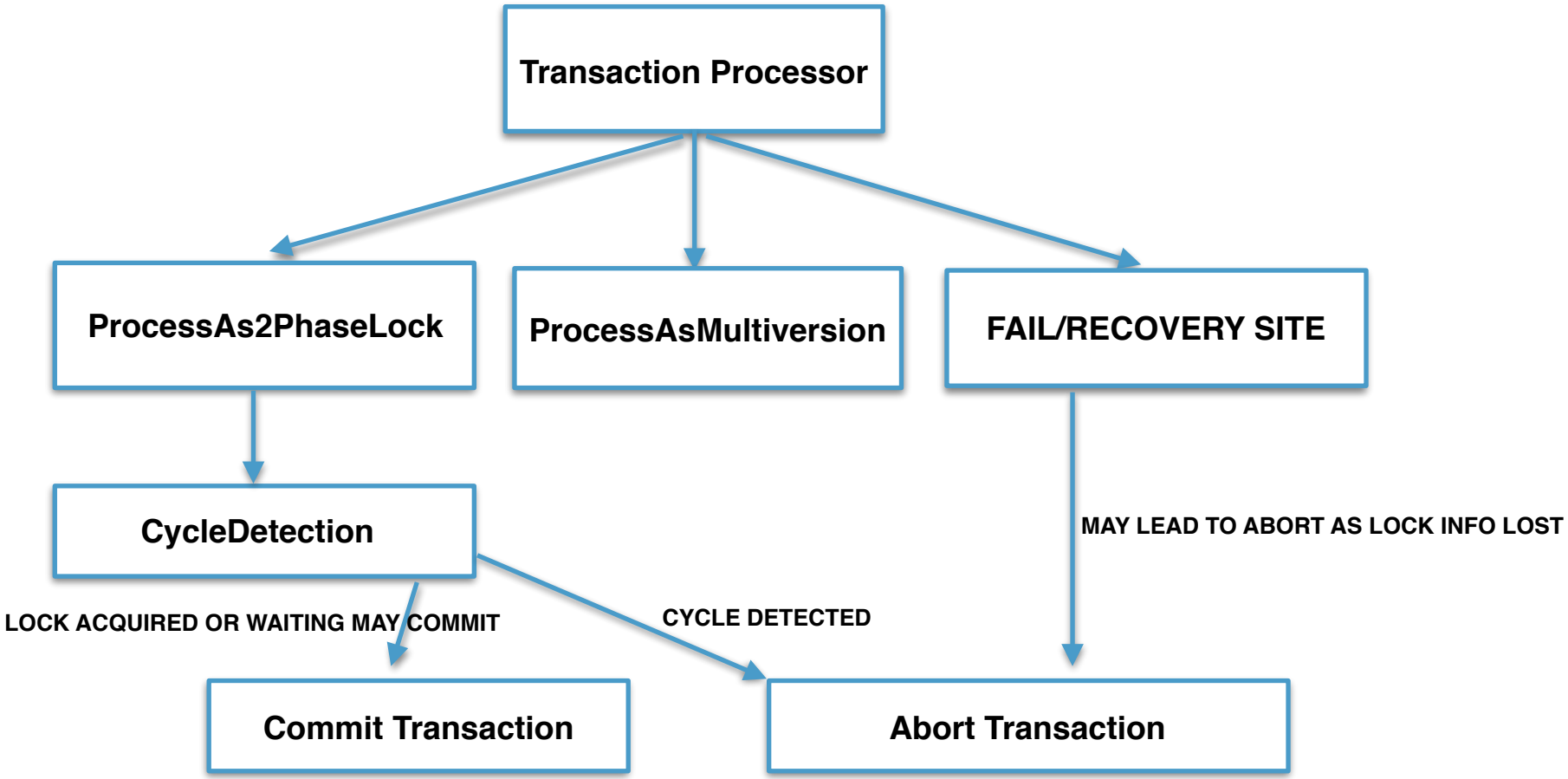
- **Module 3: 2Phase Locked Acquisition Module:**

All locks conflict for a resource except read-read and those are only allocated in First Come First Served manner so Writes don't starve.

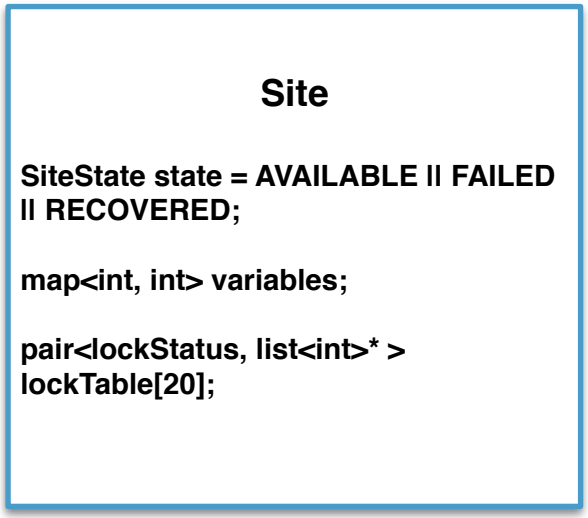
- **Module 4: Deadlock Cycle Detection:**
  Upon conflict of data, and when one transaction waits for another, cycle-detection is executed. A **waits-for-graph** in maintained order to trigger cycle detection. Upon cycle detection, youngest of all transactions based on order of their begin transaction is aborted.

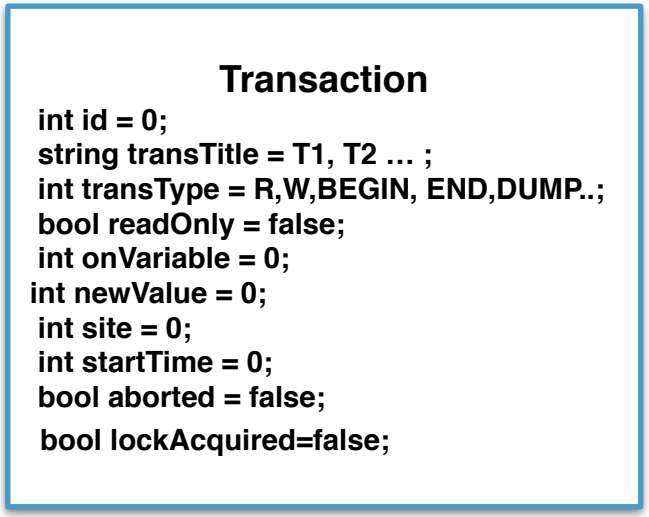
- **Module 5: Site Failure and Recovery:**
  Upon of a site, lock table is erased and **transaction that accessed the site are immediately aborted.** This is achieved as each site maintains it's site status and it changes from Available to Failed to Recovered.

- **Module 6: Output Log/Result:**
  Logging to standard output is part of each module based on the type of scenarios possible. Dump writes to Standard out for all sites.



A high level design flow diagram with modules is included to show the possible flow of a transaction.
Site and **Transaction are two main** objects that are used:





Some Design Considerations / Limitations:

- Multi-version Read-Only transactions will abort only in case if No Site for the variable is up and will not retry upon up.
- If a transaction is waiting on a resource, (No further Operations for that transaction are expected to receive.)
- If a transaction is waiting on a resource, then End Trans should not be issued.
- For aborting transactions, value read for an operation is not printed to console.
- At max 20 Transaction's Cycle can be detected.
- Transactions are always treated and processed in the order that the operations received and Locks Acquisition is also FCFS based.
- Transaction Processing follows as per requirements 2-Phased Locking, Multiversion Concurrency and Failure Recovery Scheme.
- 10 Test Cases are added in the folder with naming convention 'mytestcase_1.txt' for more testing.