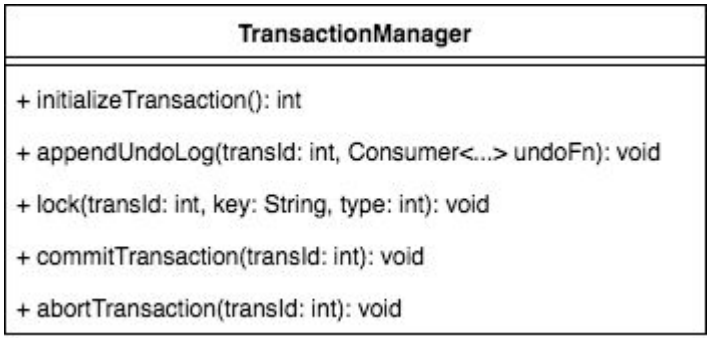
COMP 512 Project Phase 2

Centralized approach

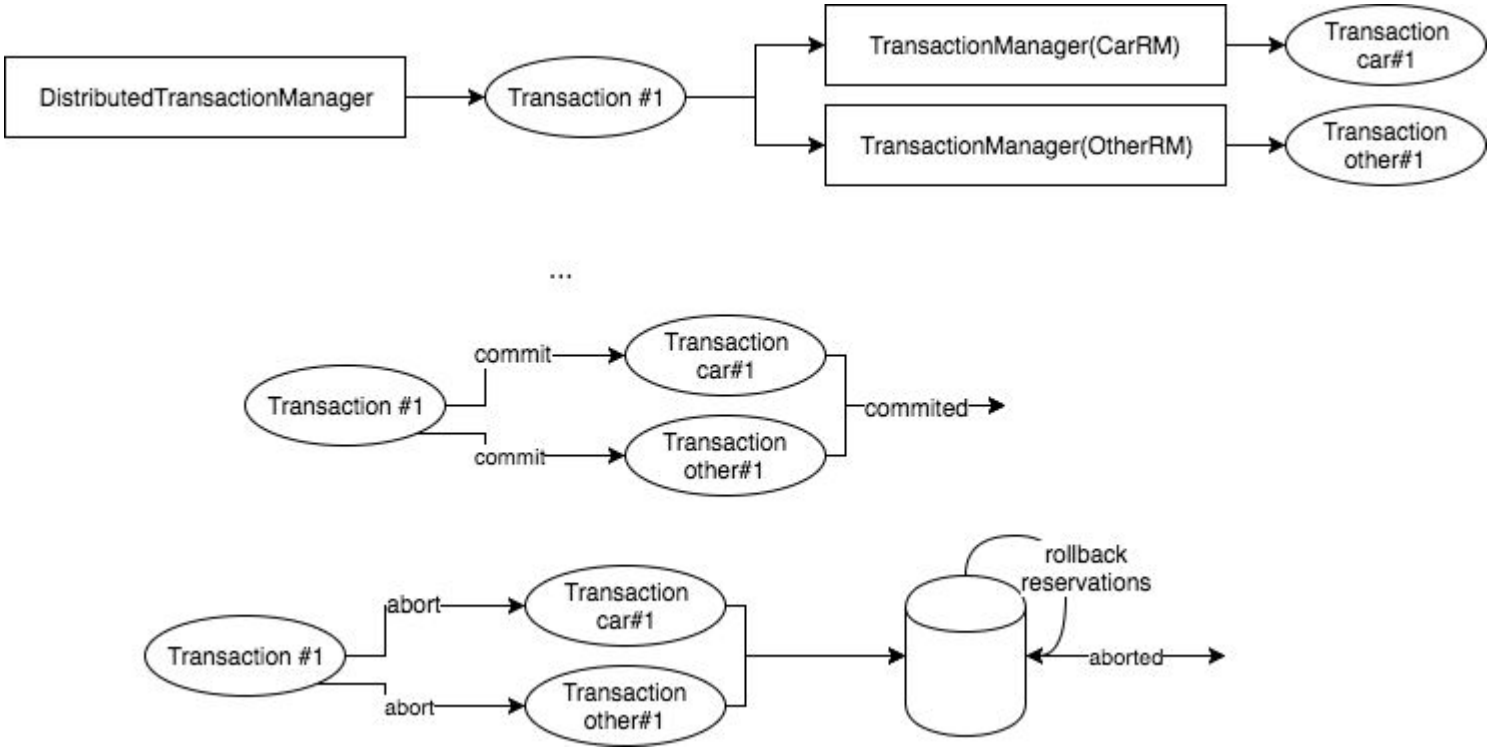
- At each resource manager
- Transaction Manager that keeps track of transactions
- Use an UndoLog of Stack<Consumer<ResourceManagerDatabase>>
- 1-phase commit:
  - In case of abort -> we rollback the undo log



Distributed approach

- Wanted a 2PL / 1-phase commit approach
- Kept the centralized approach at each resource manager
- DistributedResourceManager
  - Keeps track of every enlisted ResourceManager along with their transactionId
  - DistributedTransaction
  - Commit -> commit every enlisted transaction @ each RMs
    Abort -> abort every enlisted transaction @ each RMs
- Problem: Reservations
  - Use a similar UndoLog for transactions in each DistributedTransaction
  - Stack<Consumer<MiddlewareCustomerDatabase>>

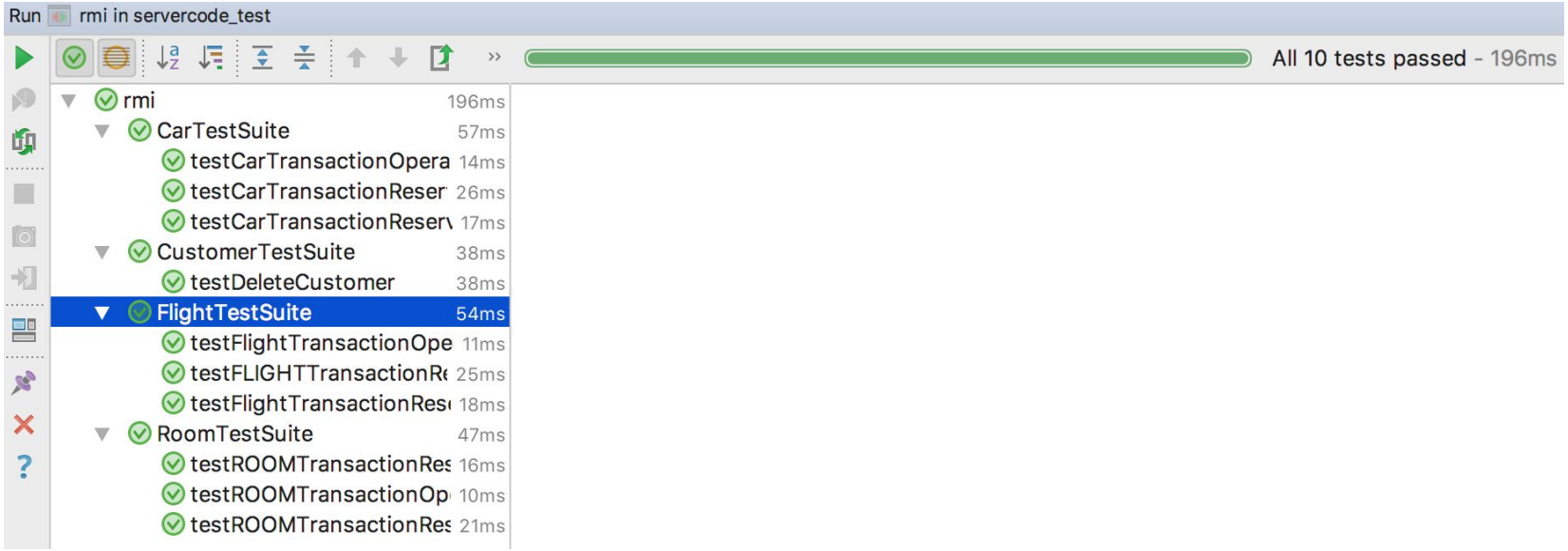
Distributed approach



Testing

- Extended our test suites library from phase 1
- Testing focuses on transactions and aborts specifically
- Strategy
  - For each Reservable Items (cars, flights, rooms)
    - Basic operations not involving reservations (creation, query)
    - Reservation operations (resources given back after abort)
    - Deletion operations (resource released after being deleted)

Testing


Run
rmi in servercode_test
All 10 tests passed - 196ms
rmi 196ms
CarTestSuite 57ms
testCarTransactionOpera 14ms
testCarTransactionReser 26ms
testCarTransactionReserv 17ms
CustomerTestSuite 38ms
testDeleteCustomer 38ms
FlightTestSuite 54ms
testFlightTransactionOpe 11ms
testFLIGHTTransactionR 25ms
testFlightTransactionRes 18ms
RoomTestSuite 47ms
testROOMTransactionRes 16ms
testROOMTransactionOp 10ms
testROOMTransactionRes 21ms