Project

Distributed Event Management System

Instructor: Dr Rajagopalan Jayakumar
Co-ordinator: Sukhinder J Narola

Prepared by: 40088004 Shalin Patel
40080612 Niralkumar Lad

The System

The distributed event management system provides the facility to manage events that are occurring in several cities. The system has basically 3 servers located at three different cities i.e. Toronto, Montreal and Ottawa.

The Client:
There are two types of users operating the system and this includes the manager and the customer. The roles of each are given below.

The role of manager in the system is to:

1. Add a new event in the system.
2. Delete an existing event.
3. View the number of free space in an event.
4. Book events for customer
5. Cancel Event for customer
6. Swap Event for customer
7. Get customer booking schedule

The role of customer is:

1. Book a space in an event.
2. View their respective booking schedule.
3. Cancel a booking for an event.
4. Swap their event.

Apart from this main role, both the user can login into the system using an unique ID and the user can exit from the system once their work is completed. The event can be created in 3 different time of a day that includes morning, afternoon and evening.

The Server:
There are 3 interconnected servers located in 3 different cities. Each server allows the manager to create one of the following events: Conferences, Trade shows and Seminars. Also, the server maintains the booking count for a specific event on a specific date as well as time and in the specific city. The facility that the count does not increase for an event is maintained by every server.

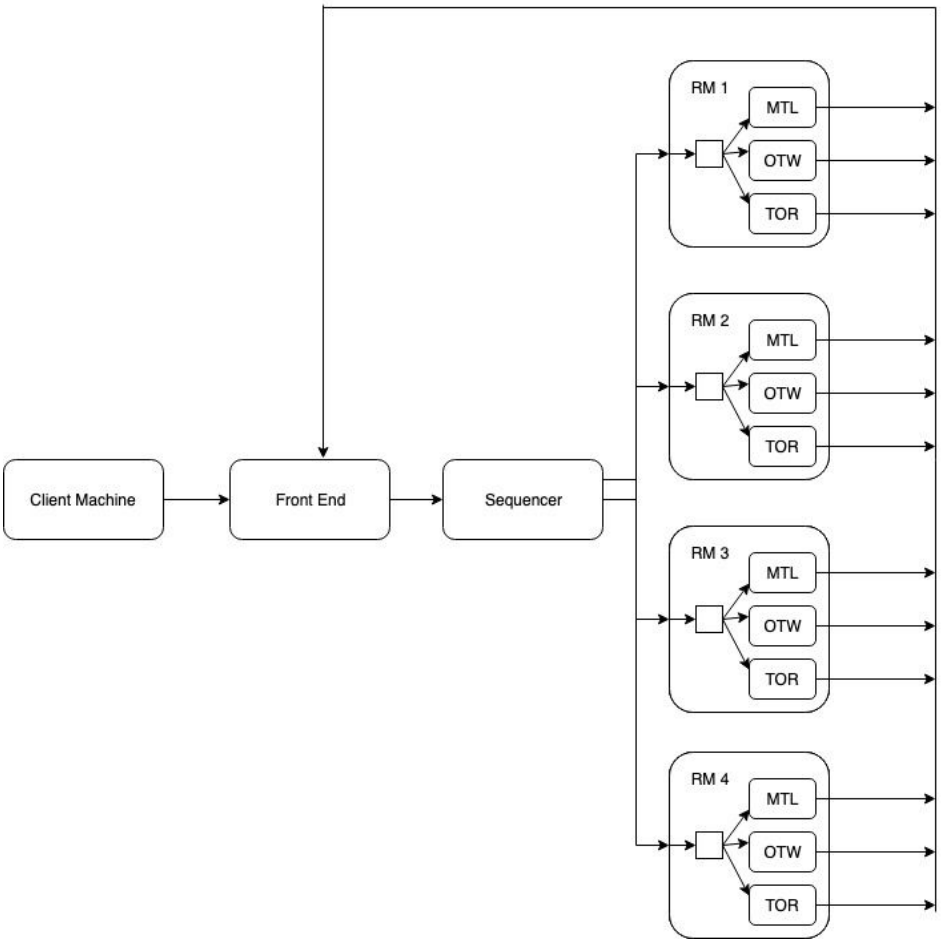
Every server keep track of every activity that is performed on it. These activities are:

1. Date and time of a request.
2. Request type.
3. Requesting user.
4. Server response to the request.

Also, the server maintain a separate log file for every user. There are some restrictions on users by the server. They are:

1. The customer of one city cannot book more than 3 events in another city.
2. The manager cannot perform any other operation other than viewing the events availability of other cities.

The System Architecture



Pic 1: System Architecture

As shown in the figure, the client is operated by the customer or the manager and every city has its own server which are interconnected.

The sketch also suggest that whenever the client tries to access the facility of other location then it has to contact it's front end which will connect to a sequencer. It will connect it to the Replica Manager and replica manager will start the server and the home server will internally communicate with the other servers and obtain the result from the other server and will finally revert it to the front end. Front end will communicate to the client and gives the output. It can also handle the software and hardware failures. Incase of hardware failure it will restart the server and gives the desired output. Hence, the client just have to communicate with the front end.

This System is designed to explain the entire functionality of the CORBA and UDP communication. We used CORBA for communicating client with the front and rest of the communication is UDP.

The Data Structure

The hashmap is used to store the data and it's structure is as follows:

General Structure: HashMap<String, HashMap<String, Event> >

Implemented in Code: HashMap<EventType, HashMap<EventID, Event> >

Event has these variables: eventID, eventType, bookingCapacity.

The Test Cases

Test on Customer:

Test Case ID	Test Case Name	Input	Output	Result
1	Login	Valid username	Logged in	Pass
2	Login	Invalid username	Rejects and ask to login again.	Pass
3	Selection menu	Valid Input	Program runs as per the selected input.	Pass
4	Selection menu	Invalid input	Rejects the option and ask for another input.	Pass
5	Book event	Valid input	Check the username, the event id, event type and event availability and then add the event.	Pass
6	Book event	Invalid input	Does not add the event.	Pass
7	Book event	Valid input	Does not allow the user to add more than 3 away events.	Pass
8	Delete event	Valid input	Allow the event to be deleted if the event exist.	Pass
9	Delete event	Invalid input	If the event does not exists then inform user about it.	Pass
10	View Schedule	Valid input	Show the event to the customer	Pass
11	Exit	Valid input	Logout the customer.	Pass

12	Swap Event	Valid Input	Events get Swapped	Pass

Test on Manager:

Test Case ID	Test Case Name	Input	Output	Result
1	Login	Valid username	Logged in	Pass
2	Login	Invalid username	Rejects and ask to login again.	Pass
3	Selection menu	Valid Input	Program runs as per the selected input.	Pass
4	Selection menu	Invalid input	Rejects the option and ask for another input.	Pass
5	Add event	Valid input	Check the username, the event id, event type and event capacity and then add the event.	Pass
6	Add event	Invalid input	Does not add the event.	Pass
7	Add event	Valid input	Does not allow the manager to add away events.	Pass
8	Delete event	Valid input	Allow the event to be deleted if the event exist.	Pass
9	Delete event	Invalid input	If the event does not exists then inform user about it.	Pass

10	View event availability	Valid input	Show the event availability to the manager	Pass
11	Swap Event	Valid Input	Events get Swapped	Pass
12	Exit	Valid input	Logout the manager.	Pass