Zeng Xia

40139418

Distributed Event Management System (DEMS) using CORBA

In this project, I used CORBA to make a Distributed Event Management System that can manage three cities' events. The CORBA DEMS achieves language independence and platform independence. The ORB core makes the platform independence. It can help a client to call a method of an object. Its function same as communication module in JAVA RMI. Meanwhile, I make an IDL file to set all CORBA interfaces and using object adapter and implementation repository make the language independence. The object adapter as a bridge to connect between the IDL interfaces and programming language interfaces. Also, the implementation repository can activate registered servers and locate running servers. The platform independence and language independence are of great significance to CORBA. The platform and language independence enabled CORBA to achieve more inclusive distributed system which can run on any platform and use any language.

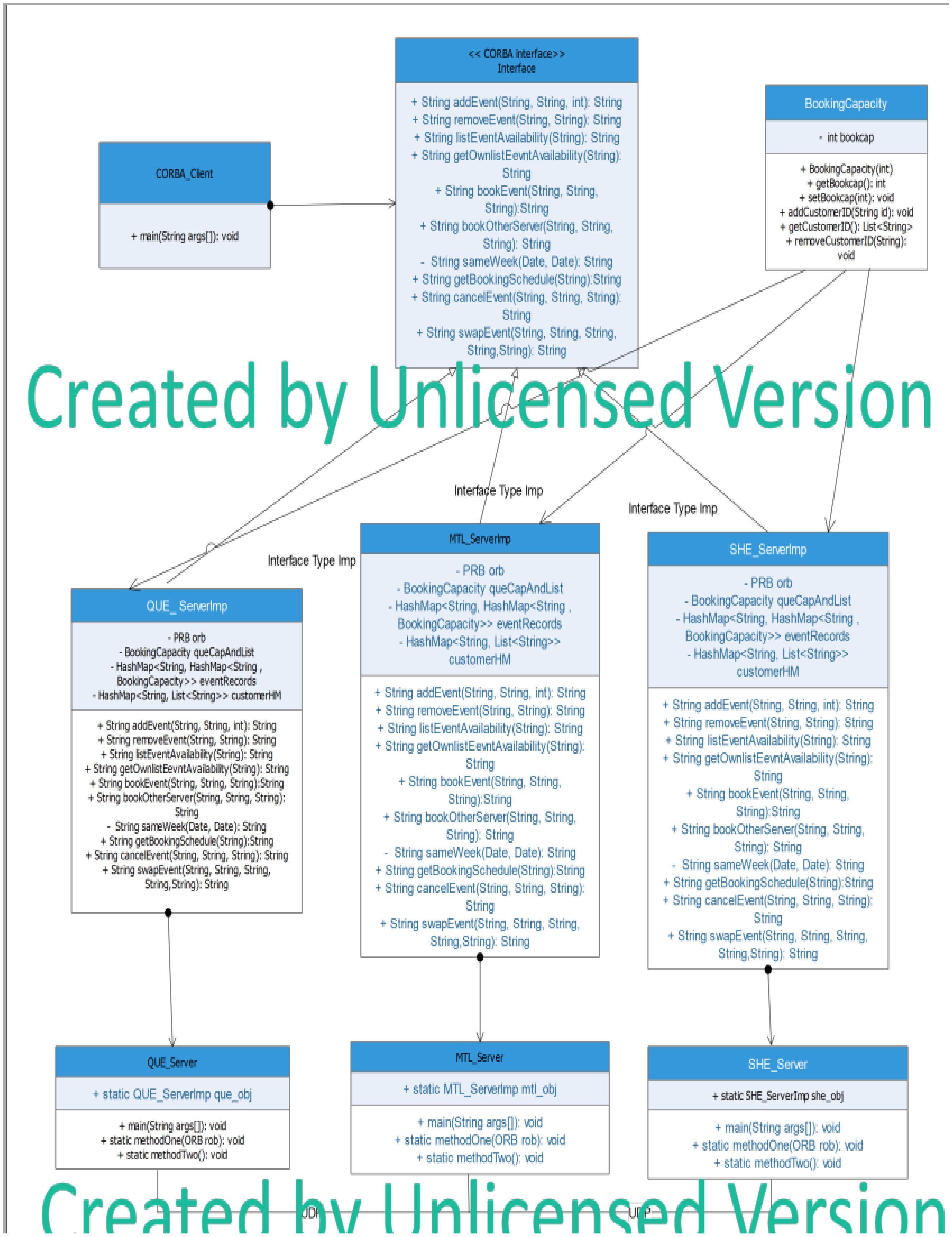
Therefore, this project uses CORBA to do the RPC between client and three servers and uses UDP socket to do the IPC between servers. Also, in this project each server uses a nested HashMap to store all events information. In this nested HashMap that uses three city's name (Conferences, Seminars and Trade Shows) as the keys. All eventIDs and events' details as the value. And using eventIDs as the subkeys. Then, using events' details as the sub-values. The CORBA can invoke the needed methods. Some of the methods can use UDP socket to get or send information to other servers.

For this CORBA DEMS, firstly, I all test all functions that performed in the previous projects to make sure the all functions can also execute on this CORBA DEMS project. To make sure the test cases are enough. I added many events in each server. Then in each server I booked some special cases, such as booking an event until the capacity is 0 and using same customer ID to book 3 events in the same week. Then, in order to prove that I successfully added and scheduled events. I ran the listEventAvailability method and the getBookingSchedule method to list all adding events ID and booking events ID. Then, I called removeEvent method. The result showed remove the event and rebook another event successful. Also, I tested the cancel method that checked the customer ID is correct. If the customer ID is correct, the method canceled the event for the customer. Then, I tried using Visual Studio and C++ to test the CORBA's platform and language independence.

The most important test is the new method swapEvent method. I tested all the possibilities. If canceling old event is succeed or not succeed and booking the new event is succeed or not succeed. Just when both canceling old event is succeed and booking the new event is succeed that swapEvent will be successful, otherwise the result should be unsuccessful.

In this project the most important part is design the CORBA and UDP socket work together. I set the CORBA and UDP in the same server class, but the CORBA will block the process and wait for a command, also the UDP does the same operation. So, I need design a threads concurrent method to make CORBA and UDP do not block each other. Therefore, ensuring thread concurrency is the key to their collaborative work.

Overall, I believe this DEMS CORBA can realize the distributed system. Performing remote process communication and inter process communication. This DEMS can accurately execute customer and management requirements to manage and invoke the database stored on the servers.



<< CORBA interface>>
Interface
+ String addEvent(String, String, int): String
+ String removeEvent(String, String): String
+ String listEventAvailability(String): String
+ String getOwnlistEevntAvailability(String): String
+ String bookEvent(String, String, String):String
+ String bookOtherServer(String, String, String): String
- String sameWeek(Date, Date): String
+ String getBookingSchedule(String):String
+ String cancelEvent(String, String, String): String
+ String swapEvent(String, String, String, String,String): String
CORBA_Client
+ main(String args[]): void
BookingCapacity
- int bookcap
+ BookingCapacity(int)
+ getBookcap(): int
+ setBookcap(int): void
+ addCustomerID(String id): void
+ getCustomerID(): List<String>
+ removeCustomerID(String): void
Interface Type Imp
Interface Type Imp
Interface Type Imp
QUE_ ServerImp
- PRB orb
- BookingCapacity queCapAndList
- HashMap<String, HashMap<String , BookingCapacity>> eventRecords
- HashMap<String, List<String>> customerHM
+ String addEvent(String, String, int): String
+ String removeEvent(String, String): String
+ String listEventAvailability(String): String
+ String getOwnlistEevntAvailability(String): String
+ String bookEvent(String, String, String):String
+ String bookOtherServer(String, String, String): String
- String sameWeek(Date, Date): String
+ String getBookingSchedule(String):String
+ String cancelEvent(String, String, String): String
+ String swapEvent(String, String, String, String,String): String
MTL_ServerImp
- PRB orb
- BookingCapacity queCapAndList
- HashMap<String, HashMap<String , BookingCapacity>> eventRecords
- HashMap<String, List<String>> customerHM
+ String addEvent(String, String, int): String
+ String removeEvent(String, String): String
+ String listEventAvailability(String): String
+ String getOwnlistEevntAvailability(String): String
+ String bookEvent(String, String, String):String
+ String bookOtherServer(String, String, String): String
- String sameWeek(Date, Date): String
+ String getBookingSchedule(String):String
+ String cancelEvent(String, String, String): String
+ String swapEvent(String, String, String, String,String): String
SHE_ServerImp
- PRB orb
- BookingCapacity queCapAndList
- HashMap<String, HashMap<String , BookingCapacity>> eventRecords
- HashMap<String, List<String>> customerHM
+ String addEvent(String, String, int): String
+ String removeEvent(String, String): String
+ String listEventAvailability(String): String
+ String getOwnlistEevntAvailability(String): String
+ String bookEvent(String, String, String):String
+ String bookOtherServer(String, String, String): String
- String sameWeek(Date, Date): String
+ String getBookingSchedule(String):String
+ String cancelEvent(String, String, String): String
+ String swapEvent(String, String, String, String,String): String
QUE_Server
+ static QUE_ServerImp que_obj
+ main(String args[]): void
+ static methodOne(ORB rob): void
+ static methodTwo(): void
MTL_Server
+ static MTL_ServerImp mtl_obj
+ main(String args[]): void
+ static methodOne(ORB rob): void
+ static methodTwo(): void
SHE_Server
+ static SHE_ServerImp she_obj
+ main(String args[]): void
+ static methodOne(ORB rob): void
+ static methodTwo(): void
UDP
UDP