

Distributed Reservation Management System  
(DRMS Using CORBA)

1. Mitul Jetani(27671180)
2. Ankur Pandey (26614051)
3. Haiyang Sun(27202652)
4. Eric Watat Lowe(27720270)
5. Sumair Zafar(27099347)

COMP 6231 – Distributed System Design

### GROUP NUMBER: 10

Version [2.0]

Index Page

|  |  |  |
| --- | --- | --- |
| No. | Subject | Page No. |
| 1 | ACKNOWLEDGEMENT | 3 |
| 2 | About System | 4 |
| 3 | Sequence Diagram | 7 |
| 4 | Class Filename And Specification | 8 |
| 5 | References | 8 |

ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We are grateful to our project guide [Serguei A. Mokhov](http://users.encs.concordia.ca/~mokhov)for his guidance, inspiration and constructive suggestions that helped us in the preparation of this project.

We also thank our colleagues and lab assistance who have helped in successful completion of the project.

**All Team-10 Members**

# **Distributed Reservation Management System(D.R.M.S.)**

## About System

Distributed Reservation Management System (DRMS) for libraries: a distributed system to manage a group of library systems used by students and administrators to help them manage book reservations.

## Techniques

**CORBA**Corba is designed to facilitate the communication of systems that are deployed on divers platforms. CORBA enables collaboration between systems on different operating systems, programming languages, and computing hardware. CORBA has many of the same design goals as object-oriented programming: encapsulation and reuse. CORBA uses an object-oriented model although the systems that utilize CORBA do not have to be object-oriented. CORBA is an example of the distributed object paradigm.

**User Datagram Protocol**UDP is a communications protocol that offers a limited amount of service when messages are exchanged between computers in a network that uses the Internet Protocol (IP). UDP is an alternative to the Transmission Control Protocol (TCP) and, together with IP, is sometimes referred to as UDP/IP.

**Most Important /Difficult part**

* **Obsolete Technology:**

The CORBA is actually an obsolete technology because of that the latest version of CORBA eclipse plug-in was released on 2008 which is no longer supported by the latest eclipse release Luna. Later we installed the earlier release Juno which has support of CORBA plug-in.

* **Deadlock Problem**:

A deadlock is a state where two, or more, threads are blocked waiting for the other blocked waiting thread (or threads) to finish and thus none of the threads will ever complete. To prevent the deadlock problem the overlapping locks are avoided.

* **Synchronization issues:**

There can be many server and client running simultaneously on different machines. On adding the information of new student the information for each student has to be unique. If the list is accessed simultaneously then it will result into discrepancies of data. Similarly, while reserving the book by the students since the book can be reserved by one student at one time. Hence, it is need to be accessed by only one user at a time. The lock mechanism is used in our application.

* **High Latency:**

The objects residing in the same [address](https://en.wikipedia.org/wiki/Address_space) s and accessible with a simple function call are treated the same as objects residing elsewhere (different processes on the same machine, or different machines). This is a fundamental design flaw in case CORBA applications as it makes all object access as complex as the most complex case. The local objects are treated as remote objects which increase the latency time.

**Test Scenario**

Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. The testing phase involves the testing of system using various test data; Preparation of test data plays a vital role in the system testing. After preparation the test data, the system under study is tested.

Those test data, errors were found and corrected by following testing steps and corrections are recorded for future references. Thus a series testing is performed on the system before it is ready for implementation.

* **Library Server Testing**

|  |  |  |
| --- | --- | --- |
| **SL.No** | **Test Case** | **Test Expected Result** |
| 1. | Add new Book  Book book = new Book("AAA","BBB",1); | Successful |
| 2. | Valid Server  Student student = new Student("Aaa", "Bbb", "cc@cccc.cc", "51411111111", "aaabbb", "xxxxxx", "Concordia"); | Successful |
| 3. | Compare Server  nameOfServer.equalsIgnoreCase("Concordia").isEqual(“Concordia”); | Successful |

## Sequence Diagram

## Class File Name And specification

| Class File in Program | Specification |  |
| --- | --- | --- |
| AdminClient.java | Admin client interface |
| StudentClient.java | Student Client interface |
| Administrator.java | The model of Administrator |
| Book.java | The model of Book |
| Student.java | The model of Student |
| LibraryServerInfo.java | The model to get library server information UDP port, server name etc. |
| LibraryServer.java | Master Server for Student Book information and student Data. |
| UDPSocket.java | UDP socket which is used to communicate between servers |

**Reference**

1. <https://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture>