CONCORDIA UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6231, Winter 2022 Instructor: R. Jayakumar

ASSIGNMENT 2

Issued: Feb. 9, 2022 Due: Feb. 23, 2022

Note: *The assignments must be done individually and submitted electronically.*

Distributed Appointment Management System (DAMS) using Java IDL (CORBA)

In this assignment, you are going to implement the distributed appointment management system (DAMS) from Assignment 1 in CORBA using Java IDL. In addition to the 3 patient operations introduced in Assignment 1, the following operation needs to be implemented:

swapAppointment (*patientID, oldAppointmentID, oldAppointmentType, newAppointmentID, newAppointmentType*):

When a patient invokes this operation to change an appointment (of the specified appointment type) he/she has already booked. In this case, the current city server (which receives the request from the patient) first checks whether the patient has booked the old appointment, then checks with the new city server (on which the new appointment has to be booked) whether there is available capacity for the new appointment, and if both checks are successful then atomically book the patient for the new appointment and cancel the old appointment for the patient. That is, the book and cancel operations should both be successful or none of them should be done. Note that all these checks, book and cancel operations should be done using UDP/ IP messages as they are server-to-server communications.

In this assignment you are going to develop this modified application in CORBA using Java IDL. Specifically, do the following:

- Write the Java IDL interface definition for the modified DAMS with all the 7 specified operations.
- Implement the modified DAMS. You should design a server that maximizes concurrency. In other words, use proper synchronization that allows multiple users to correctly perform operations on the same or different records at the same time.
- Test your application by running multiple clients with the 3 servers. Your test cases should check correct concurrent access of shared data, and the atomicity of *swapAppointment* operation.

Your submission will be graded for correct and efficient implementation of all the operations in addition to correct use and implementation of mutual exclusion in accessing shared data and proper exploitation of concurrency to achieve high performance.

MARKING SCHEME

[30%] *Design Documentation*: Describe the techniques you use and your architecture, including the data structures. Design proper and sufficient test scenarios and explain what you want to test. Describe the most important/difficult part in this assignment. You can use UML and text description, but limit the document to 10 pages. Submit the documentation and code electronically by the due date; print the documentation and bring it to your DEMO.

[70%] *DEMO*: You have to register for a 5–10 minutes demo. You cannot demo without registering, so if you did not register before the demo week, you will lose 40% of the marks. The demo should focus on the following:

- [50%] *The correctness of code:* Demo your designed test scenarios to illustrate the correctness of your design. If your test scenarios do not cover all possible issues, you will lose part of marks up to 40%.
- [20%] *Questions:* You need to answer some simple questions (like what we have discussed during lab tutorials) during the demo. They can be theoretical related directly to your implementation of the assignment.

QUESTIONS

If you are having difficulties understanding any aspect of this assignment, feel free to contact your teaching assistants (Lab FI: Rajkumar Rakoli rokalirajkumar@gmail.com, Lab FJ: Brijesh Lakkad brijeshlakkad22@gmail.com, Lab FK: Stallone Macwan stallonemacwan@gmail.com). It is strongly recommended that you attend the lab sessions, as various aspects of the assignment will be covered there.