CS 5523: Operating Systems
Fall 2014

Project 2: Online Chat Server

(Due: **Dec. 9, 2014**, Tuesday before class)

- **Objectives**
  - Enhance the understanding of distributed applications and systems
  - Enhance the knowledge of middleware to support remote objects and RMI
  - Practice programming using middleware (e.g., Java RMI)

- **Description [100 points]** (this project is designed for a **group of two students);**

  General information: In this project, you are asked to implement a prototype of a online chat application that consists of a **Server** program and a **Client** program using Java RMI or other middleware.
  Here, the **Server** program creates/maintains a **ChatServerServant** remote object.

  The **ChatServerServant** object provides the following basic online chat functions:
  1. *createUserAccount*, which creates a **UserAccount** object based on the provided information and returns its remote object reference (ROR) to client;
  2. *loginAccount,* which return a user's ROR based on the login information;
  3. *creatGroup*, which creates a **UserGroup** object that keeps track of a list of interested users; a user can call this function to create a group with a given name, and by default the user becomes the first user in the group;
  4. *findUser,* a user uses this function to find another user with a given name, return the ROR of the user object;
  5. *findGroup ,* a user uses this function to find a group with a given name, return the ROR of the group object;

  The ***UserAccount*** objects should maintain a *list* of friends' RORs, as well as a list of Group's RORs the user has joined. For each friend/group, keeps a chatting history *(that is, a list of messages with timestamps)*. The user objects should have the following functions:
  1. *viewProfile/updateProfile*, which allow users to view and update their profile information. The profile can contain: user name, profession, living city, company, college name and graduate year etc;
  2. *inviteFriend*, which allows a user X to invite another user Y to be a friend;
  3. *joinGroup/leaveGroup,* which allows a user to join/leave a group for a given group's ROR;
  4. *sendUserMessage,* which allows a user X to send user Y a message;
  5. *sendGroupMessage,* which allows a user X to send a message to all users in the given group;
  6. *readMessage,* which allows a user X to read all 'new' messages from a given user/group;

The ***UserGroup*** objects should maintain a *list* of RORs of the users that joined the group. This object should also have functions to facilitate group chatting within the users of the group.

For the **Client** program, it should provide an interactive user interface (text or GUI based) that allows users to interact with the Server program (such as accept user's input, invoke corresponding RMIs and display the message).

Specific requirements: Since more than one clients may simultaneously connect to the Server program, proper synchronizations should be included. Moreover, not all users will be online all the time. A user should be able to check out all 'new' messages sent to her/him from her/his last login time.

Work partition:

- It is recommended that the group members work on the interfaces together; then one implement the Server program and another one implement Client program.

Extra credits:

- **[10 points]** Implement **ChatServerServant** , **UserAccount** and **UserGroup** as **persistent objects**, which allow the server to start and read their information from disk with existing user/group information;
- **[10 points]** Implement a **callback** object on Client side that allow the server to push new messages send to the client side of a user;
- **[10 points]** Other significant expansions for the functionality of the program;

Report: Write a project report that should include the following materials and discussions.

- The status (completion or not) of your project;
- The work partition of your project: who completes what?
- The design of your project, such as interfaces, class/method relations etc.
- Where are the synchronization problems in this project and how do you solve it?
- What are the challenges and difficulties that you encountered for this project and how do you overcome them?
- List the references (such as books and links) that help you finish this project.
- Comments and suggestions for this project: which part you like or dislike; what's is your opinion to make it easier/interesting/useful etc.

- **Project submission (what and how):**

  - For the program codes and electronic copy of your report: zip them to a single file and name them as LastName-Proj-2.zip (for example, Zhu-proj-2.zip) and upload the file on Black Board. **No hardcopy is needed for the program codes**;
  - **For the report**: print a **hardcopy** and hand in before the class **on the due day.**