Universität Zürich
Institut für Informatik

Lecture **Distributed Systems**
Fall 2009

Prof. Abraham Bernstein, Ph.D.

ASSIGNMENT 3 - TUPLE SPACES (40 POINTS)

Due date: Dec. 15, 2009, 10am (CET)

Rules

- Assumed programming language is *Java* in version 1.6
- Code that is handed in and does not compile will NOT be graded. So please make sure to test your implementation properly.
- Assignments have to be solved individually. It is ok and also desired to discuss problems with peers, whereas copying code is not. As a result, plagiarism will lead to 0 points for the particular assignment for both parties.
- The due date is a hard deadline. Emails that arrive after this deadline will be discarded and therefore the contained solution not graded.

All the above rules are final and no matter for further discussions!

The third assignment is about implementing your own chatroom client using TSpaces from IBM.

TSpaces is a set of network communication buffers called tuple spaces and a set of APIs (and classes that implement the API.) for accessing those buffers. TSpaces allows heterogeneous, Java-enabled devices to exchange data with little programming effort. The package includes server software that implements the buffers and client software for accessing the buffers.

Now your tasks:

1) Preparation

1. Install Java, if you haven't yet.

2. Download IBM TSpaces V2 . There is TSpaces version 3 available now. We haven't tested our implementation with V3 - but feel free to have a try. You possibly need to register if this is the first time you download software from IBM AlphaWorks.

3. After installation, make yourself familiar with TSpaces. Read the User's and the Programmer's Guide and have a look at some examples in the Examples section of the documentation. You should at least understand the basic functionality of TSpaces.

2) Chatroom Implementation (**30 Points**)

Your task is to implement a chatroom client using TSpaces. Therefore we provide you with the following five Java classes, whereof one is unfinished.

* ChatLoginGUI.java - Class providing main method to start up the chatroom client.

* ChatClientGUI.java - The chatroom client. (UNFINISHED - complete it!)

* MessageEvent.java - A message to be written/ taken to/from the tuplespace.

* JoinEvent.java - An event indicating that somebody has entered a chatroom.

* LeaveEvent.java - An event indication that somebody has left the chatroom.

Download chat.zip which provides all of these classes and unzip it.

The next step is to have a look at the five classes and to understand what they are used for. Extend the relevant (marked) parts of the classes to get the chatroom client up and running properly.

Read the Programmer's Guide to learn how to set up a TSpaces server on your local machine (the server manages the tuplespaces). Test the server with one of the simple examples you'll find on the TSpaces website.

* If you ever run into this:

"TupleSpaceServerException: ClassNotFoundException: SomeClassName"

please note: If a client writes an object of "SomeClassName" to the server, then the SomeClassName.class definition must be available to the server via the CLASSPATH. Make sure that, when you start the TSpaces server, the directory containing the application classes is available to the server.

Refer to the Programmer's Guide for more information about this problem and how to avoid it.

After setting up the server and having completed the client code, test your chat system locally (use several chat clients on the same machine).

Optional: test your implementation with a remote server/ other remote clients.

3) Create a short documentation in which you briefly describe your implementation, such that somebody who has not seen your code can understand what you did. Also state the advantages of Tspaces for such kind of applications. (**10 Points**)

The whole documentation should not be longer than 2 pages.

Grading:

Grading will be based on

a) the correctness of your code, i.e. does it solve the given task?

b) readability/structure of your code (including appropriate comments) and treating of exceptions

c) clarity of your documentation, i.e. does it really describe what you implemented and how well can it be understood by somebody who has not written or read the code.

What to hand in and how:

- Create a zip file named <your_student_id>_<first name>_<last name>_assignment02.zip (e.g. 1234567_John_Doe_assignment01.zip). This zip file should contain the completed ChatClientGUI.java (Make sure that your code is comprehensive, e.g. inline comments). It should also contain your documentation as *.pdf file.

In case your code requires any special treatment to compile, you have to enclose a README describing the necessary steps.

- Send this zip archive on time via email to Floarea Serban (*serban|at|ifi.uzh.ch*) . The email subject should start with ***[DS 2009]*** .

Helpful reading

http://www.alphaworks.ibm.com/tech/tspaces

http://www.almaden.ibm.com/cs/TSpaces/html/UserGuide.html

http://www.almaden.ibm.com/cs/TSpaces/html/ProgrGuide.html