Sofia University “St. Kliment Ohridski”
Faculty of Mathematics and Informatics

IM Chat Server/Client

Contemporary Java Technologies Coursework

Stanislav Peshterliev, 61096

10 September 2010

Table of Contents

1. Introduction

According to Wikipedia[1] “client–server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers that are known to each other only through a network”. The most popular example of client-server architecture is Internet, where in order to view a web page you send a request (url address) to a web server and then receive response (HTML) which is processed by the client (web browser). Other popular client-server application is for instant messaging programs and protocols, like IRC, ICQ, Jabber and etc. These allow users to exchange text or binary messages over a network almost instantaneously. Clients connect to a server and then they can send messages to other on-line users connected to the same server. Usually all messages between users are forwarded by the server, but there are also peer to peer chat programs like Skype where messages are directly exchanged. In my coursework I am implementing the first case, because it better illustrates client-server architecture.

2. Architecture

2.1. Server

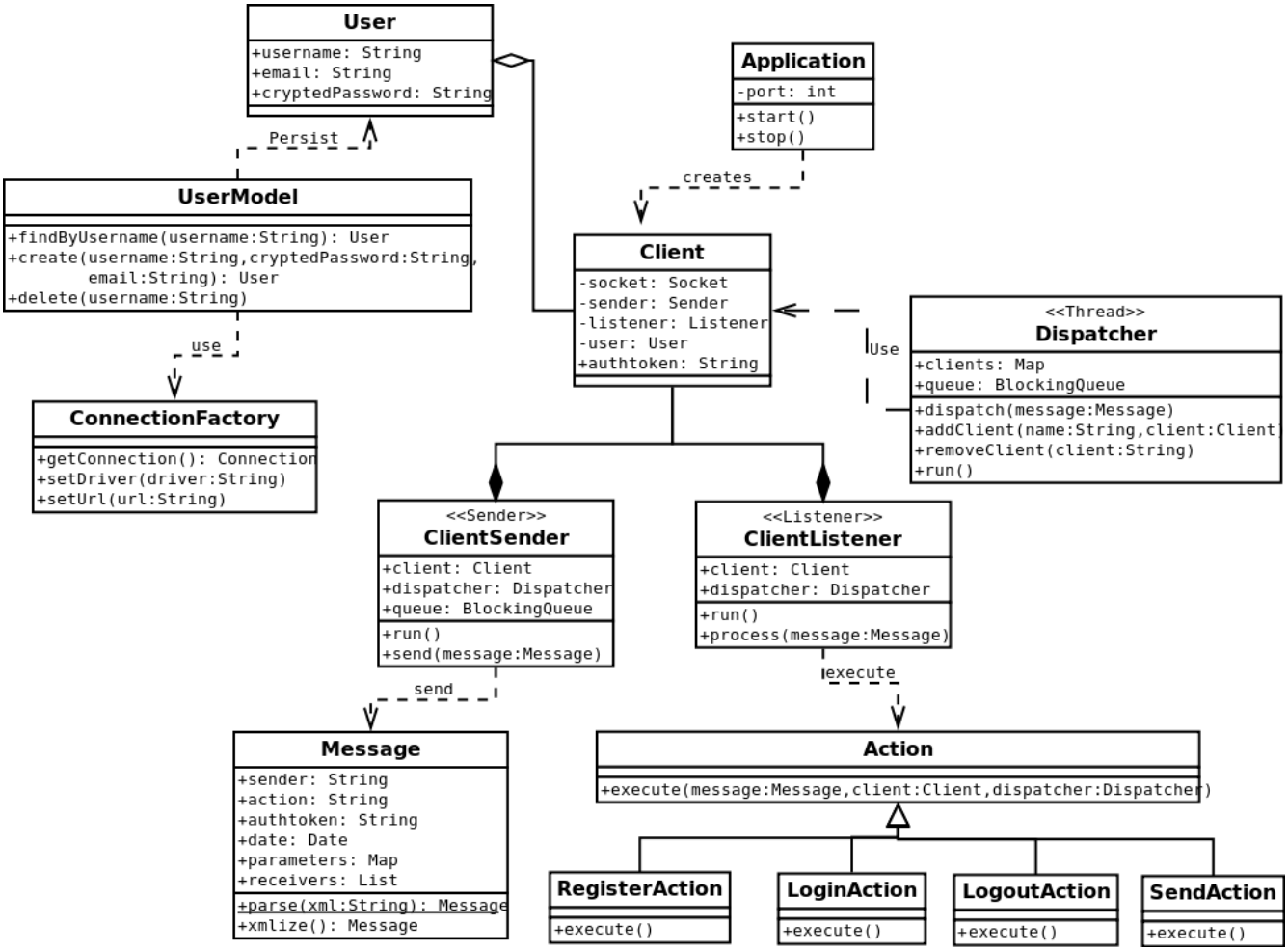
IM Server is started as a process. In the main thread it listens on a specific port for incoming client connections. Additional dispatcher thread is started, responsible for distributing messages to clients. For each connected client two threads are started - one for handling incoming messages and one for sending messages. Dispatcher's and client's sending message threads are use queuing to avoid the case when there are more messages than server machine is able to send. The server is designed to respond to predefined set of actions: Login, Logout, Register, Send. In order for a user to send message to another user they both must have registration and be authenticated with user name and password.

The server is implemented using the standard Java threading and networking APIs. I am using Google Guava library because of its convenient APIs for working with collections and strings. SQLite database is used for persistence. JUnit 4.5 for unit testing.

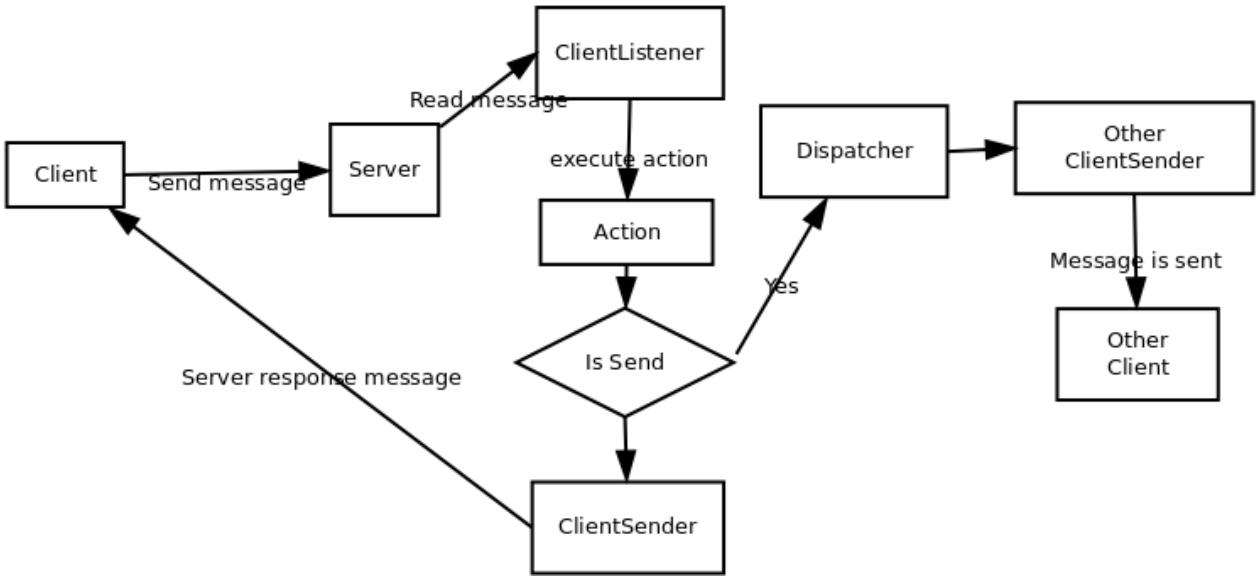
1. Package Structure

- im.server – main server classes for handling low level server details like networking communication and threading
- im.server.actions – action classes are responsible for handling user request; when message from user is received it is processed by one of the classes in this package
- im.server.actions.helpers – helper classes used by actions; helpers are used to bring a higher level of abstraction in implementation of action classes.
- im.server.model – classes connecting to database and manipulating persistent object
- im.common – contains classes shared between client and server application

- im.util – utility classes which can be used in any application



2. Process diagram



2.2 Client

IM Client is an application with graphical user interface. It is used to register, login and send/receive messages. When started two threads are created one that listens for incoming message and one that sends messages to the server. The thread that sends messages to the server maintains a queue in case that the user sends more messages than the client machine or network can handle.

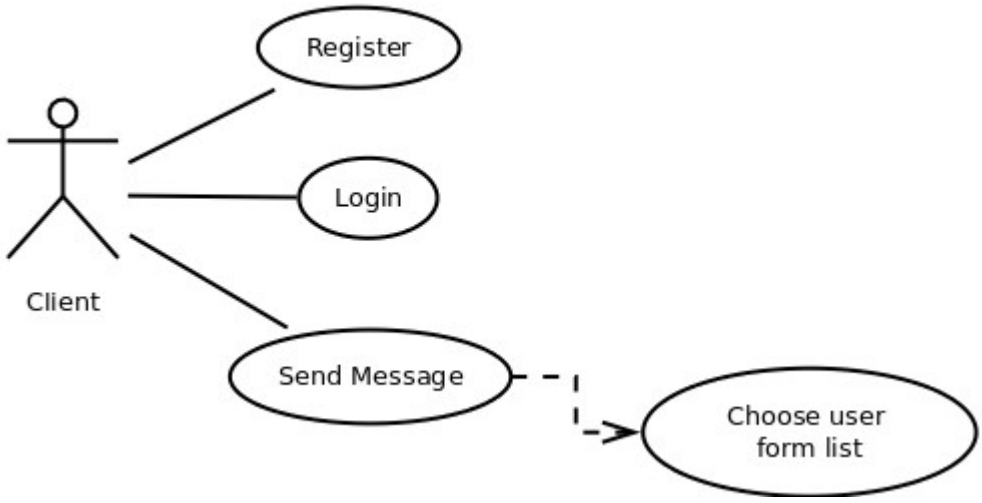
Requests to the server are initiated in the event handler of JFrame objects located in im.client.ui package. These object are created using the utility factory object Frames which graduate that there is only one object with given name supplied. This is convenient for keeping only one window per user that we exchange message with. Server response is handled by action object located in im.client.ui.actions package.

The client is also built with the standard Java threading and networking APIs. Graphical user interface is implemented with Swing GUI toolkit.

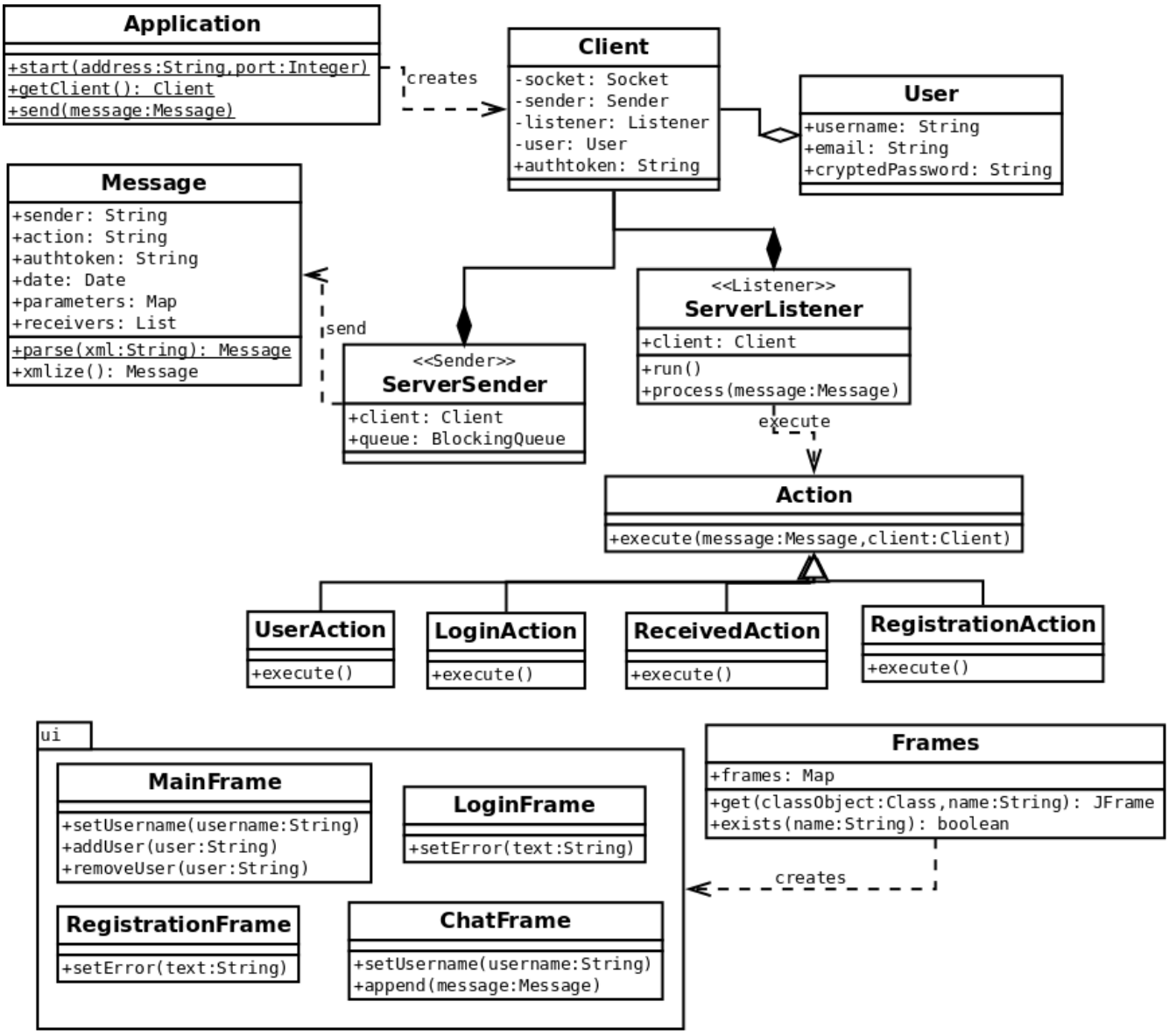
1. Package Structure

- im.client - main client classes for handling low level server details like networking communication and threading
- im.client.ui – classes related to graphical user interface and user interaction.
- im.client.ui.actions - action classes are responsible for handling server response responses and messages from other users; when message from server is received it is processed by one of the classess in this package and user interface is updated
- im.common - contains class shared between client and server application
- im.util - utility classes which can be used in any application

2. Use case diagram



3. Class diagram



3. Development Environment Setup

The project is developed using Netbeans IDE 6.9.1. In order to run the application open Netbeans, import server, client, common. Install sqlite3 database.

Server can be started from server/test/im.server.StartTestServer. When server is started make sure that all unit tests from server/test are passing.

Client can be started from client/test/im.client.TestApplication. Start several instances of the client in order to test chat functionality.

4. References

[1] http://wikipedia.org

[2] Sventlin Nakov, Internet Programming with Java, Faber, 2004