T-106.5600 Concurrent Programming Assignment 2 - Plan

by: Hannu Huhtanen 291288

--Goal--
Implement a simple chat system that utilizes a tuple space implemented with Java synchronization mechanisms.

--Style guidelines--
All the code will be written cleanly and according to best practises as far as the assignment requirements allow.

--Implementation plan--

Task A: Tuple space

The tuple space will have a dataholder for the saved tuples and all access to this data through methods defined in the interface will be synchronized. The get and read methods will stay waiting in case the tuple they were looking for doesn't exists at the time of calling the method. After a tuple has been inserted with the put method waiting threads are awakened. I'll probably create a object for a tuple which has a overriden equals and hashcode methods to make finding tuples from the space easier.

Task B: Chat system API

The caht system API consist of two parts, The ChatServer which is the chat program that connects to the network and ChatListeners which each handle a single channel in the chat. The communications between ChatServers are done through a common tuple space. The data structure inside tuple space took some thinking but I ended up with a plan. The common properties of the network are saved in one tuple. For each channel there is a own status tuple. Also each message is saved in it's own tuple. This way it is easy to track the order of the messages and remove those that are outside the defined buffer size. Below are examples for tuples.

Server status: ("chatserver", <RowcountAndChannelnames>)
channel status: ("chatserver", "channelname", <FirstAndLastMsgId>)
message: ("chatserver", "channelname", "MsgId", <Message>)