ZeroMQ based chat service

Grupo de Sistemas Distribuídos
Universidade do Minho

1 Objectives

Implement a chat service, both client and server(s) in Java using ZeroMQ, exploiting mainly the Pub-Sub pattern.

2 Tasks

1. Design a first version involving a single server (broker). Write also the client program, which interacts with the server through ZeroMQ, being both publisher and subscriber. (Telnet cannot be used as a client here.) Clients should be able to change chat room, through some local command, e.g., `\room uminho`, but there is no authentication. Use room names for subscriptions.

2. Write a more scalable version involving a federation of server nodes. In this version there are two fixed lists of nodes, where clients connect to publish (S subscribing-like nodes) and where they connect to subscribe (P publising-like nodes). When a client starts, it chooses one of each randomly. Plan the number P, for a given S and the average number of clients that each message reaches, so that each server nodes handles approximately the same number of messages (assuming no multicast).

3. Modify the previous version, to allow dynamic addition of publishing server nodes (while having a fixed list of S subscribing nodes). Publishing nodes may be added by contacting a main server, to which clients also connect when they start, so that it tells them which nodes to connect to. (Now, clients only know the address of this main server.) When a client stops, it informs the main server.