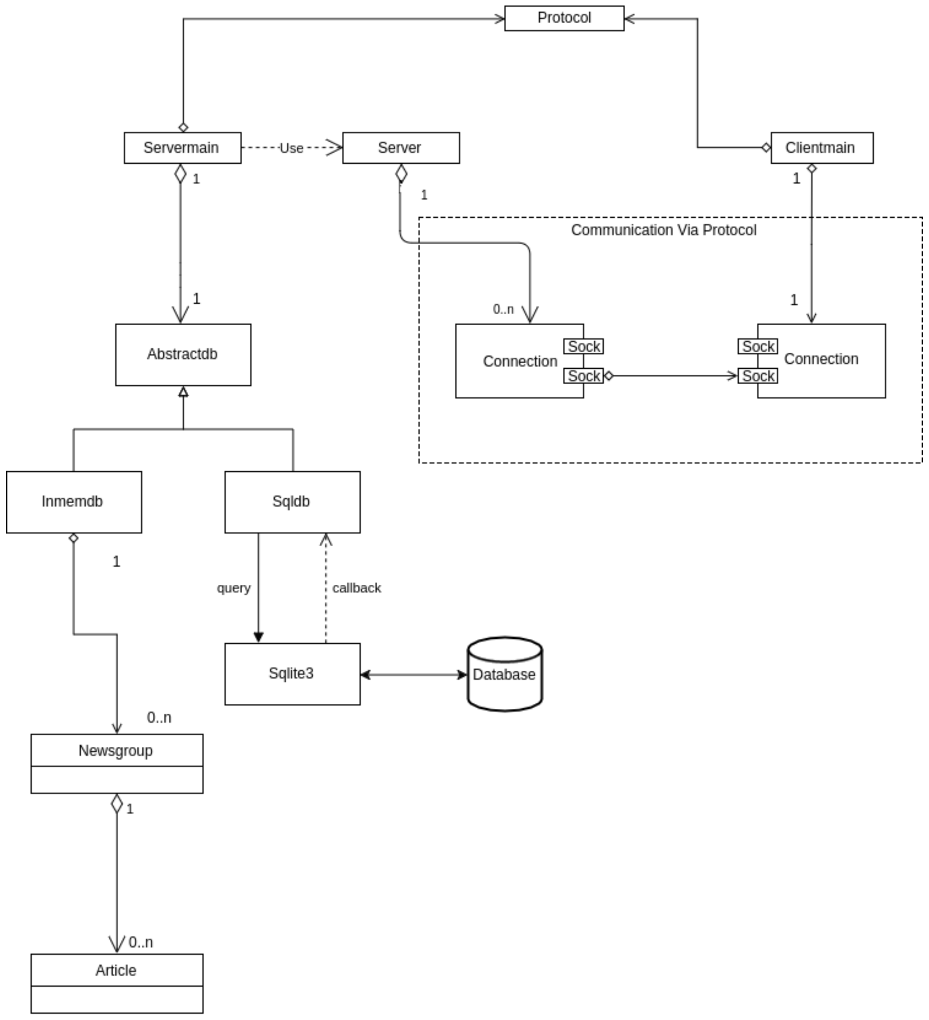
Project - EDA031

Fredrik Danebjer (dat12fda@student.lu.se)
Jonas Danebjer (kin13jda@student.lu.se)
Valthor Halldorsson (fys11sha@student.lu.se)
Simon Johansson (tna11sjo@student.lu.se)

March 30, 2017

1 System Design

1.1 System UML



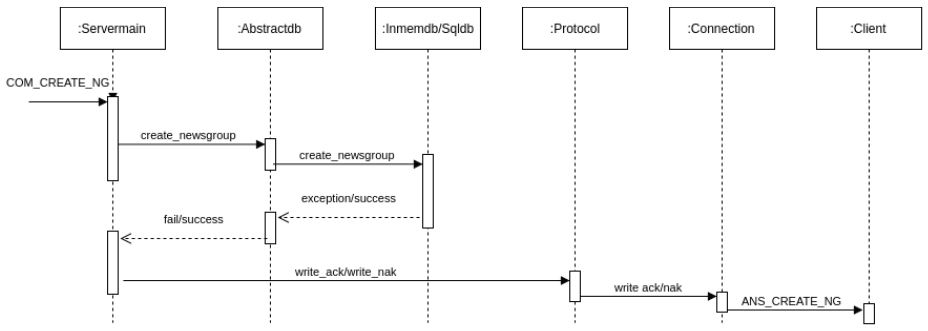
Class responsibilities

- `Abstractdb`: Acts as an abstract superclass so that servermain doesn't have to think about which actual database is instantiated at runtime. It contains the methods used for accessing the database.
- `Inmemdb`: The in-memory version of the database, storing articles in news-

groups objects which themselves contains lists of articles.

- `Article`: Basic article class with the required attributes.
- `Newsgroup`: Basic newsgroup class with a list of articles and it's own identifying members.
- `Sqldb`: A class that interacts with the sqlite3 database API which stores it in a local file (called Database.db).
- `Servermain`: The main class which instantiates the server and database. Can be started with either the inmemory version or sql version of the database. Also makes sure the data recieved is correct and sends error messages to the client if such is not the case.
- `Server`: Handles sockets and waits for activity.
- `Connection`: Handles the basic operations for transferring/receiving data via sockets.
- `Protocol`: The official protocol with which the client and server communicate. Contains functions for writing/receiving data.
- `Clientmain`: The program used by the user to access the newsgroup system.

2 Interaction Diagram



The above sequence diagram depicts the data flow from when the server recieves a message (in this case, Creating a newsgroup) to when the client has received an answer.

3 Conclusion

Our database fullfills all the requirements. One problem that occured was using `sqlite3` C API as an interface to our database, and we should have spent more time looking for better wrapper libraries since a lot of errors were introduced by using that API. For the in-memory database, the concept of lists in lists was quite easy to implement and understand. Most of our problems were with the SQL database. Other than that, the network code required a bit of effort to get working, but mostly because it was repetetitive. In retrospect, a better way of avoiding repeating similar functions should have been found.