Homework 1: Collaborative Text Editor

Hadachi&Lind

November 28, 2016

Must Read:

1. Deadline for doing homework is 4 weeks starting from now “2016.10.26”
   a) due date is: 2016.11.23 23:59:59 EET

2. For any delay in submitting the homework they be will a penalty (-2% of the grade for every day delay)

3. You can work in groups of maximum 4 people

4. Python 2.7 only

5. You should create a public git repository for your project and share it.

6. Be aware that the git should contain a time line progress of your implementation in case there is only one big commit at the end you will be penalized

7. You are allowed to use any fancy open source library that will help you in your design and development.
   a) Except for network handling. This means, you have to provide your own implementation of the application layer protocol based on transport layer of your choice (UDP, TCP, or Both)

8. Be aware that in case you use any ready product or source code of any existing collaborative editor it will be considered as plagiarism.
1 Introduction

One of challenges when you are working in team is to collaborate in an efficient way. Moreover, working from distance with a team is a nightmare especially when you are creating a report or you are brainstorming ideas. From this perspective, the idea of creating a real-time collaborative text editor was born. The collaborative text editor permits multiple users to work on a single document simultaneously via the network. To create such application involves many challenges like handling concurrency, dealing with network connections, and building the GUI. After all it is also a fun homework because you will be building a tool that you can use with your friends and teammates.

2 Motivation

The main motivation behind this homework is give you the opportunity to practice all the knowledge acquired from the course in creating a useful real world application. During, the process of creating your collaborative editor you have to apply some fundamental design and implementation techniques (sequence diagram, object models, etc).

In addition, the Homework will lead you to apply variety of technologies related to socket programming, networking and I/O. Besides, this assignment will deepen your learning about queue based treatments of concurrency, threads programming, shared state aspect, modeling and designing and interface development. Finally, you will have this experience of working in team with all its benefits and challenges.

3 Specification

In this homework, you have significant freedom in designing your editor as it please you. Not only in the implementation but also in the behavioral aspect. However, we suggest you that you implement first the minimal implementation at beginning till you have a stable version then you can add all other functionalities or features.

Your freedom can be resumed to the following decisions that you have to make:

- What set of features the editor includes
  - It is not necessary to make looking like MSOffice of LibreOffice, it can be just simple ASCII based editor like Notepad or even simpler.
  - The most important feature is of course the collaborative editing, therefore all the corresponding features are obligatory (share document, invite editor, remove editor, etc.). The exact features here depend on the strategy you chose for collaborative editing.
  - Some primitive routines like Save File, Open File, Undo, Redo etc. are optional, but nice to have.
• Which structure to adopt for the document
  – Whatever you choose or implement, it must support the concurrent modification and conflict resolution
• How to name the document and where to store them
  – Dependent of your sharing strategy! Possible strategies:
    * Server stores all the documents, the clients do create new documents or modify existing.
      - Clients cannot edit the document off-line
      - All the concurrency and editing conflicts are resolved by server
      - Server maintains the ownership/permissions:
        1.) what document is allowed to be edited by which clients,
        2.) who is the owner of what document.
    * Client (master) stores the document and exposes it for editing using server, other clients (slaves) can only edit the document when the master is connected to server.
      - Server act as a proxy, and never stores any documents, just redirecting the requests/responses from the slave clients to master client.
      - Server maintains the groups of users:
        1.) in each group there is one master client
        2.) and one or many slave clients.
      - All the concurrency and editing conflicts are resolved by master client.
      - Slave clients cannot edit the document off-line
    * Client stores the document, but once it gets shared all the other clients get the copy of it.
      - Clients can modify documents off-line, they synchronize once they get on-line.
      - Server act as a proxy, and never stores any documents, just redirecting the requests/responses between the clients.
      - Client groups are maintained by the clients themselves:
        1.) each client knows which other clients to contact in order to sync the document,
        2.) server just provides the communication channels between the clients.
    * Any other sharing strategies with your own design!
      - Just make sure the requirements are fulfilled
– Irregardless of what you have chosen as a sharing strategy it must be made clear and reflected in your documentation!

• How and what guarantees are made regarding the concurrent edits effect.

– Synchronous strategies or “change-on-write”
  * Locking strategies
    - Each modification locks entire document
    - Each modification locks the portion of the document
    - ...
  * Change submission strategies:
    - Each time keyboard key is pressed a change gets submitted
    - Each time a full word is entered a change is submitted
    - ...

– Asynchronous or lock-free strategies (using document versions)
  * Version control is required

Nonetheless, your implementation has to meet some requirements and they are as follows:

1. **Real-time.** The collaborative editor should be capable of providing the real-time response due to editing and viewing the changes.

2. **Tenacity.** User should be able to edit a document over several sessions:
   a) user should be able to continue editing from where he left the document
   b) in case there were changes introduced by other users the document must reflect them once a users is editing the document again

3. **Multiple users.** The collaborative editor should allow at least three users to work simultaneously.

4. **Textual format.** Allowing ASCII text and handling the line breaks.

### 4 TASKS

The task can be resumed as follows:

1. **Individual preparation:** We advice you to complete all the tasks before starting the homework. *(Optional)*

2. **Putting up the Team:** Meet up and creating you own team and define your strategy.
3. **Modeling and designing:** You can choose any tool that can help you in creating your object models, sequence diagrams, UML diagram, etc. Make sure you define clearly and carefully what the collaborative text editor is. Please document all your process in modeling and making decision and why; in all the case you will need it for deliverables. *(include in the report)*

4. **Architecture:** Design the system architecture with all its component. *(Include in the report)*

5. **Protocol:** Design a protocol for the system’s components. Express your protocol as a state machine. *(Include in the report)*

6. **GUI:** Sketch your user interface and its different screens and dialogs. This step will also allow you to discover the structure and the flow of your interface. *(Include in the report)*

7. **Code Design:** at this level you may use object model and model dependency diagram to help you define the pattern, how to handle the concurrency design, and make it easy during testing step. *(Include in the report)*

8. **Testing Strategy:** you should adopt a testing strategy (by determining in which order you will test your components, how you will derive test cases, what level of coverage you plan to achieve, how you will test the user interface, etc). Please justify your strategy in the report. *(Include in the report)*

9. **Implementation:** write your code following your design. be-careful to update your report in case you made any change in the design during the implementation stage, which is highly probable to happened.

10. **Testing:** Run your testing strategy and document all the results. *(Include in the report)*

11. **Reflection:** Write a brief comment about you experience as a Team by answering the following question,
   a) what was hard?
   b) what was confusing?
   c) what was easy?
   d) what would you do differently if you had the chance to do it again?
   *(Include in the report)*

5 **DELIVERABLES AND GRADING**

5.1 **Deliverables:**

They will Three major deliverable for this Homework:

**First:** You have to submit your Team members’ names including you and also the public git repository link via the moodle latest 2nd of November at 12h00.
Second: A report that includes the following:

- Modeling and designs
- The network architecture and protocol;
- The code designs and discussion;
- The GUI design;
- The testing strategy;
- Testing results
- Reflection

Third: The source code of your implementation.

The implementation, we will consider only the source code from your public git repo provided. Note, any modification done after the due date to the git won't be considered.

Concerning the report, it should be compressed and submitted in the course website by one member of the team.

Link: https://courses.cs.ut.ee/2016/ds/fall/Main/Kodutööd

5.2 Grading:

Total points for this Homework is 20 pts, and its breakdowns is as follows:

- 6 pts: Modeling and designs, the network architecture and protocol, the GUI design;
- 5pts: The code designs and discussion, the testing strategy,
- 7pts: The Implementation and testing results
- 2 pts: Reflection