

University College of Northern Denmark

IT-programme

AP Degree in Computer Science

Dmaj0919

**Multi user and concurrency problems**

Mini Report

(Sebastian Holesz, Martin Hotka, Maros Cuninka, Samuel Horacek, Ioan-Sebastian Voinea)

Aalborg, 25. 10. 2020

## **Introduction**

For this semester project we decided to develop a C2C platform where students could post their school homework/assignments and others could help them solve it or directly solve it for them. The system would contain two clients (UIs), one for the end user in the form of a web application developed using ASP.NET, and one for moderators in the form of a dedicated desktop application developed using WPF.

## **Multi user and concurrency problems**

In our platform we are dealing with 4 different types of users:

1. a poster - a person who posts an assignment;
2. a solver - a person who provides a solution to the assignment;
3. a basic user - a person who just browses the website;
4. moderator - the only person who would use the dedicated client to moderate the posted assignments/solutions and the forum

The most common multi user/concurrency issue we are anticipating are:

1. multiple users trying to post an answer to a solution at the same time
2. poster deleting a post while the solver is answering it, leaving the solver’s solution “hanging up in the air” as there is no post to associate it with
3. poster confirming an answer and paying for a solution while the solver is deleting the solution leaving the poster’s payment “hanging up in the air” as there is no solution to associate it with
4. two users registering with the same credentials at the same exact moment
5. moderator deleting someone’s post for abusing platform rules while the solver is answering that post, leaving the solver’s solution “hanging up in the air” as there is no post to associate it with
6. moderator banning a user while some poster is confirming an answer and paying for a solution which was posted by the banned user, leaving the poster’s payment “hanging up in the air” as there is no solution to associate it with
7. user is adding credits to his account and at the same time moderator accepts the credit return request and adds the credits to users account

### **Multiple users trying to post an answer to a solution at the same time**

The solution we have come up with, is a queueing mechanism that would use timestamps from the client side for determining the queue order. Whenever a solution is posted, there would be a timestamp attached to the server request, which would then be taken into consideration by the server which would decide on the queue order based on that.   
This optimistic approach would prevent wrong queue order because of the differences in internet connections or any other reasons. Like this, there would always be a queue based on the real time the solver posted his solution.

### **Poster deleting a post while the solver is answering it, leaving the solver’s solution “hanging up in the air” as there is no post to associate it with**

The solver posts a request to the server. The solution has the ID of the assignment attached to it, so the server starts looking for the same ID among the posted assignments. If it finds it, but it is disabled, deleted (not from the DB but from the web application UI) or inactive, the solver gets a response from the server, that the assignment has been deleted in the meantime and his solution was therefore not accepted.

If, on the other hand, the server finds an active assignment with a matching assignment ID, it attaches the solution to it, using a transaction for that process), so if there is a delete request from the poster in the meantime, it first lets the solution to be attached and then the post is deleted or disabled. If in fact the post is to be deleted after the transaction that attaches the solution to the post is finished, another transaction is started (for deleting the post) and the post is deleted.

### **Poster confirming an answer and paying for a solution while the solver is deleting the solution leaving the poster’s payment “hanging up in the air” as there is no solution to associate it with**

When the poster is going through the answers and decides on one that he likes and chooses to pay for it, a transaction begins, that blocks any deletion attempts from the solver, giving him an error message saying he cannot delete the answer anymore.  
On the other hand, if the solver deletes the answer the transaction begins, giving any potential “solution buyers” an error message that the solution was deleted already. This may create a situation, where a wrong solution was accepted, before the solver was able to take it down. We have decided to solve this problem, by connecting the two parties right after the payment happens, where there will be a chance for them to discuss the solution and solve the possible problems, by agreeing on a refund. During this period (of probably 24 hours) the money will be held by us so if any refund requests happen, we can just return the money to the poster.

### **Two users registering with the same credentials at the same exact moment**

We were deciding between optimistic and pessimistic approach. We decided to handle this issue with pessimistic locking by applying Try/Catch construct, it prevents users and applications from editing data that is being or has been changed. Processes know immediately when a locking violation occurs, rather than after the transaction is complete.

### **Moderator deleting someone’s post for abusing platform rules while the solver is answering that post, leaving the solver’s solution “hanging up in the air” as there is no post to associate it with**

The solution for this problem could be the same solution we use for the problem number 2, as the logic behind it is the same, just the user who is deleting the post is different (poster vs moderator).

### **Moderator banning a user while some poster is confirming an answer and paying for a solution which was posted by the banned user, leaving the poster’s payment “hanging up in the air” as there is no solution to associate it with**

The solution for this problem could be the same solution we use for the problem number 3, as the logic behind it is the same, just that in this case, the solution is not deleted, but the solver is banned by a moderator, which means none of his assignments or solutions can be used.

### **User is adding credits to his account and at the same time moderator accepts the credit return request and adds the credits to users account**

We decided to solve this problem with a possible lost update by applying a transaction, which will block the resources and wait for them to finish the initial task before starting the next one. The next process will read the updated value from the database.