GatherChain Usability Evaluation

João Almeida (42490) — 2021 — FCT-NOVA, DI

Figure 1: *Gatherchain logo*

Abstract

In this paper, it is proposed a blockchain-based solution for version control of model-driven engineering artefacts. The goal is to facilitate collaboration in a multi-user area, like the education field, and track changes in a trusted and secure manner. This solution is based on using the **Hyperledger Fabric Network** to govern and regulate file version control functions among students and teachers. A thorough state-of-the-art is done in project management tools, modelling tools and authenticity technology. Research highlights many related work who try to solve similar issues. The risks and disadvantages of existing collaboration solutions are considered, as well as possible ways of their elimination.

This document provides the usability testing script to be performed with the participants of said test. It acts as a guide to follow up with the users. The script was written in Portuguese and translated to English after the tests were completed.

Keywords: blockchain, version-control, GUI, usability testing, electron

1 Introduction

First of all thank the participant for coming in and taking the time to test my project. Explain that the session will take approximately 45 minutes. Secondly, and the most important thing, assure the user that there are no wrong answers. We're here to test this solution an try to fix it and improve it.

2 Screening Usability Questions

In this section we try to get more relevant data about the participants, with their consent of course. These questions were:

- What's your age and gender?
- From 1-5 rate your proficiency using a computer?
- From 1-5 rate your exposure to programming?
- From 1-5 rate your exposure to version-control systems (*eg*: git)?
- Do you have a GitHub account? (If not, the participant can use one created for testing purposes).

3 Tasks

In this section we provide some scenarios to be followed by the participants in GatherChain. Following the *Customer Effort Score (CES)*, after a completed task we ask how difficult or easy it was to complete it.

1. *During class your professor asks you to create groups for the next project. You join forces with your buddies Sméagol and Gollum, with student numbers, **1937** and **1954**, respectively. Your professor also shares with class an application to be used during this project. He says it's an application that tracks all the changes made in a project. Your buddies already registered in the application during class and are now waiting for you. You wait to get home and use the app. They ask you, as you're the last one to register, to also create the group. Create a directory in the computer to host the professor's project, and using the GatherChain App, show me how you would proceed with the registration, creation and initialization of the solution. Just don't forget that your student number is **1992**.*
2. *The group is now registered in the GatherChain solution, great work! Your professor asks you to create a text file named "Hobbit.txt" in the directory previously created to host the project. He asks you to add a line with the word "Precious".*
3. *You created the file and updated in your computer, but your colleagues can't see it. Using the GatherChain solution show me how would you do share the file with your group.*
4. *Your colleagues can now see your changes, and they have already made changes of their own, but you can't see them. Using the GatherChain solution show me what would you do. You should see in the Hobbit.txt the added lines.*
5. *Your colleagues made more changes, check them again. Oh no! Gollum made a mistake in the last change and you're the only one who saw it. Using the GatherChain solution revert to the last good change, the change before the Gollum (1954) one.*
6. *You want to check what was written in the file in the second change counting from the beginning, but you don't want to revert all the changes made. Using the GatherChain solution show me what would you do.*

4 Post-Testing Questions

After the users completed the usability test we ask questions about their general experiences:

1. Overall, what's your experience been with the app?
2. What did you like about the app?
3. What didn't you like about the app?

And also use the *System Usability Scale (SUS)* to let the participants score from 1-5, ranging from *Strongly Agree* to *Strongly Disagree*, 10 items:

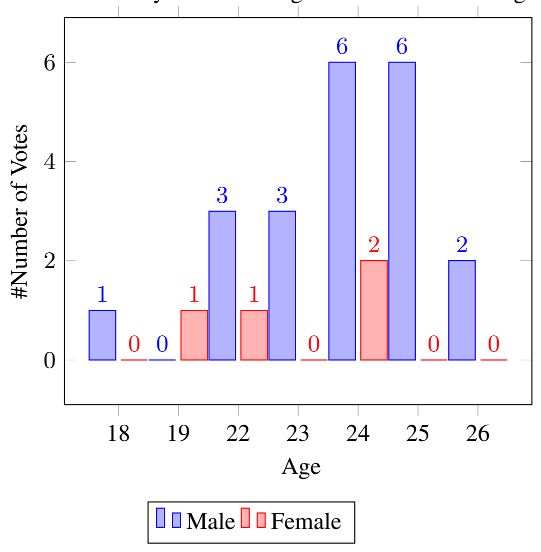
1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.

3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

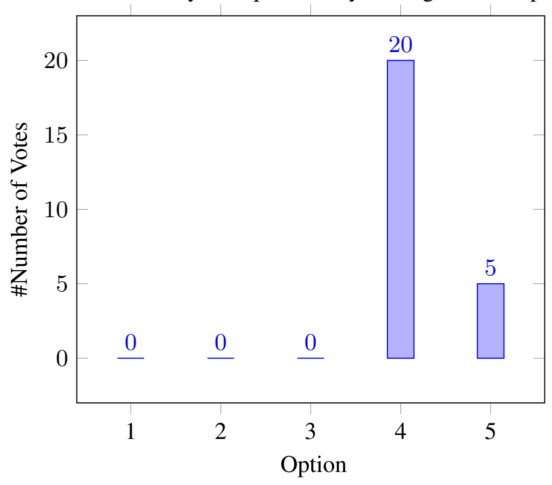
5 Results

In the end we got, 25 users to test the system, which we appreciate very much the liberty to take the time to help us. The participant's demographic distribution was the following:

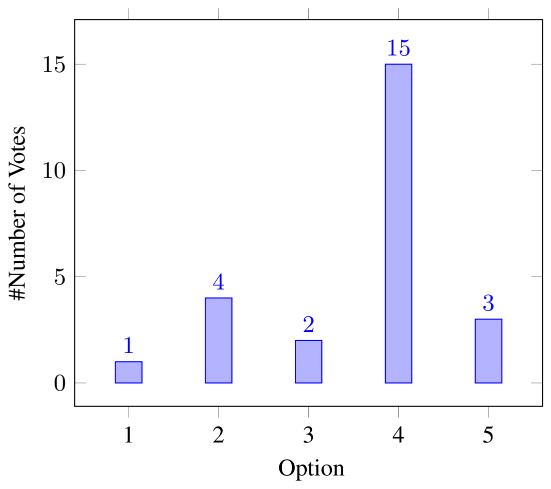
- What's your age and gender?



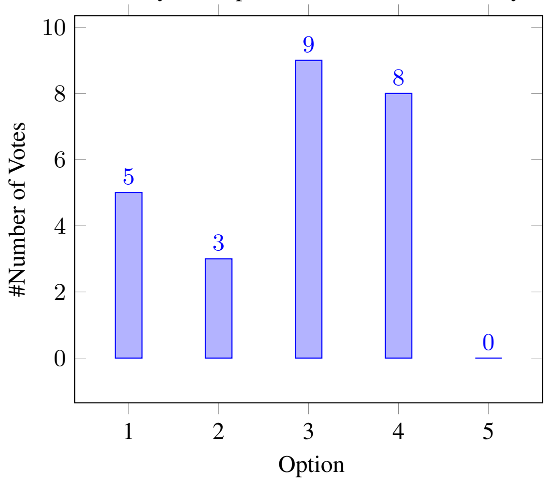
- From 1-5 rate your proficiency using a computer?



- From 1-5 rate your exposure to programming?

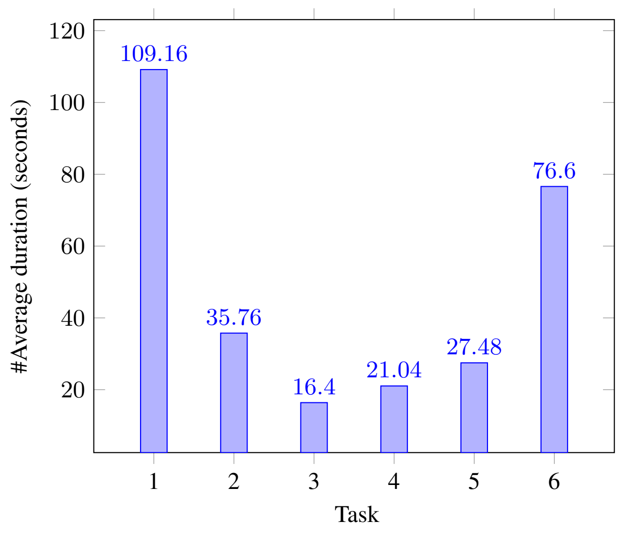


- From 1-5 rate your exposure to version-control systems?



During the testing process the task that took the most time for the participants to complete was the 1st task (which was the task that had more for the user to do) with an average of *1min49sec*. Although the task that the participants had more problems, and an average of *20%(5)* of the participants couldn't even finish it, was the last task, number 6. All of the participants had difficulty finding where to press in the app to check the artefacts of given commits.

An overview of the tasks duration's average is shown in the next graphic. The times are only for the duration of the task not including the duration of execution of the backend (eg: After the user registers the group the timer is stopped. The timer doesn't record the time it takes the app to register the group.). The recorded times for task number 6 are only for those who finished said task successfully.



The SUS scoring system works in a different way. The participant's scores for each question are converted to a new number, added together and then multiplied by 2.5 to convert the original scores of 0-40 to 0-100. Though the scores are 0-100, these are not percentages and should be considered only in terms of their percentile ranking. Based on research, a SUS score above a 68 would be considered above average and anything below 68 is below average.

The SUS scoring came as an average of *31* points. Multiplied by 2.5 it gives *77*. The SUS score is considered above average, but very close to the lower limit. A SUS score of *75* has higher perceived usability than *70%* of all products tested. It can be interpreted as a grade of a B-. The maximum score given by a user was an *82.5*, an A (the top 10% scores), which is also the point where users are more likely to be recommending the solution to a friend. The lowest score was a *72*. Here, there was no correlation between scoring and demography. The question with the highest score in average was question number 3 *"I thought the system was easy to use."* with an average of *4.6* points (normalized). The lowest scoring question was number 6 *"I thought there was too much inconsistency in this system."* with an average of *3.3* points (normalized).

Most users answered that the application was simple and easy to use. The most experienced with version control systems liked the simplicity of use and the integration of GitHub. The only negative that they thought of, was the pruning of branching abilities. The less experienced liked the ability to see all the transactions in the home screen, and didn't like the duration that some commands took to execute.