UNIVERSITY OF CONCORDIA
DEPARTMENT OF SOFTWARE ENGINEERING

Open Problem #1

Student name: *Karamveer Kaur*

Course: *Software Engineering Process* – Professor: *Dr. Pankaj Kamthan*
Due date: *August 6th,2019*

PROBLEM T2-OP1

SOFTWARE WASTE.

Give TWO examples from the educational and/or professional projects that you have worked on which led to notable (major and irreversible) waste. Described the type(s) of waste occurred, suggest plausible ways in which waste could have been avoided, and the lessons you learned.

Following are the two examples of the problems I faced while working on projects and assignments during the coursework. First, while making a concurrent client-server application for enrolling the students in a particular course through an online system. I faced following challenges

1. **Low readability:** where there were multiple nested conditions and loops. Some logics became so complicated when I tried to add more functionality/features to the application. Debugging that application took hours and even a day. Some of the conditions cannot be changed as it became so complicated to refactor it.

2.**Dead code:** Some of the conditions in loops never reached. Maintaining it required redundant efforts which could have been avoided otherwise.

3. **Scalability:** When designing the application, I didn't give a thought about, how can I make it scalable for large number of users. Later, when I was adding additional functionality towards the application, enabling atomic transactions and high availability (of server) was not easy task. Second, when working on advanced database project which has the requirement to count pairs in a very large text file. Selection of pair of integers according to their occurrence in the large text file. I faced some issues/challenges are hereby enlisted:

4.**Time taken for execution:** Some of loops has logic to generate the pairs. Some pairs which are counted already according to the given requirement, are counted again which lead to more runtime of the application. This increased the execution time of the program more than expected.

5. **Robustness:** Test cases were not fully covering all the possible conditions of the code. Hence, code was not robust at the time and the lack of time made to skip some test cases for some issues.

6.**Traceability:** Some lines of code and logic was not traceable which made it harder to understand later.

SOLUTIONS

Solutions for resolving issues:

1. Spare more time brainstorming the given problem before actual writing of the code.

2. Design level question should be tackled before providing to writing of the code.

3. Traceability matrix should be made which list the rationale behind the given piece of code.

4. Test cases should be covering the edge cases of the application to make it more reliable.

5. To increase readability function/method, variable names should be chosen wisely which will make sense later on.

6. Always keeping in mind about vertical or horizontal scalability of the applications before making big decisions will help to make application more scalable for future upgradations or addition of features.

7. Removal of dead code at all cost as later it will lead to poor quality of application as well as requires more efforts to maintain the application. "The best code is no code at all".

8. Always design application according to the SOLID principles which will make application easy to maintain and understand.

1. REFERENCES

1. https://codepunk.io/the-seven-wastes-of-software-development/