Software Development

Bachelor in Computer Science and Engineering

Guided Exercise 4,

Refactoring and Simple Design

Name		Mail	Group
Student 1:	**ENRIQUE ALCOCER DÍAZ (100472118)**	**100472118@alumnos.uc3m.es**	**19**
Student 2:	**DAVID BARROSO MURCIA (100472229)**	**100472229@alumnos.uc3m.es**	**19**

TABLE OF CONTENTS:

1. Introduction

In the proposed guided exercise we were asked to summarize four articles related to refactoring that are relevant for software development.

2. A Live Environment to Improve the Refactoring Experience

Publisher: Association for Computing Machinery, New York, NY, United States

Published: 08 December 2022

Authors: Sara Fernandes, Ademar Aguiar, André Restivo

Document

In this article, a software development tool called Live Refactoring Environment is proposed. This tool is a prototype plug-in for the IDE IntelliJ Idea(although it is mentioned that it could be implemented in other IDEs as well) that assesses in real time the source code quality of the program being developed, looking for potential refactors. A color gradient from green to red is assigned to every line of code, telling the developer which lines need refactoring the most, and multiple function extraction solutions are proposed, letting the developer choose. As of the publication date, Extract Method is the only refactoring methodology included, but the authors of the article mention that they plan to include more refactoring techniques, such as the detection of lazy classes and duplicated code. Moreover, they want to empirically assess in the future the developer experience. This article is related to software development, as it proposes a different approach to refactoring code, enforcing a better code quality in real time that could potentially reduce the amount of time spent doing further refactoring.

3. Migrating a large JavaScript Web UI to TypeScript to improve developer experience

Publisher: Lappeenranta-Lahti University of Technology LUT

Published: 2022

Author: Maija Heiskanen

Document

TypeScript is a statically typed superset of the programming language JavaScript. In recent years, TypeScript has been ranked on Stack Overflow as the most liked programming language alongside Rust, which is also statically typed. Therefore, there is a strong tendency towards static typing, especially in large code bases. This thesis studies and ensembles a plan of migration of JavaScript user interfaces to TypeScript, showing practical examples. Although this migration is time consuming, it is concluded that the developer experience quite improves, as TypeScript enhanced the features of the IDE's autocomplete and enforced a consistent structure, making the code more readable. To sum up, this thesis narrates a refactoring plan that improves code readability and thus developer experience.

4. ChatGPT prompt patterns for improving code quality, refactoring, requirements elicitation, and software design.

Publisher: arXiv, Cornell University.

Published: 2023

Author: Jules White, Sam Hays, Quchen Fu, Jesse Spencer-Smith, Douglas C. Schmidt

Department of Computer Science, Vanderbilt University, Nashville, TN, USA

Document

In the article proposed, the author analyzes the new tools provided by the development of powerful AI models, focusing on LLM's (Large-scale Language Model), in which we can find the most known until now, ChatGPT, and the use that can give in improving different aspects of software development. Taking into account that the approach of many users is directly generate software through these tools, the author realizes the huge improvement that can be done to software that is already developed, and the user wants to change specifications to improve its readability and overall quality. The article referred analyzes how ChatGPT uses prompts, to specify in a better way how to use the tool, and how to apply it to code. The fact that this is an important innovation of our era, I think is highly valuable to know how to use the tools of not the future, but the present, and therefore it should have a relevant part in the course.

5. Refactoring Codes to Improve Software Security Requirements

Conference: International Conference on Industry Sciences and Computer Science Innovation

Published: 2022.

Author: Abdullah Almogaheda, Mazni Omarb, Nur Haryani Zakariac.

Department of Software Engineering, Taiz University, Taiz, Yemen and School of Computing, Universiti Utara Malaysia, Sintok 06010, Malaysia

Document

Refactoring is seen by the majority as a great technique in order to make the code more readable and to be highly related to maintainability, testability and understandability, but what is forgotten is that with the appropriate refactoring method, the security of our code can increase. Techniques such as the *Extract method* or *Encapsulate Field* are possible ways of refactoring our code, and that is why this course is encouraging the learning of this aspect. In my opinion, understanding that refactoring is not only about changing names of variables or just making your code more readable, but also increasing its security by the use of hiding classes or methods, encapsulating them, is a great addition to the course.