ARTÍCULOS REFACTORING

Miguel González Medina

Kayla DePalma, Izabel Miminoshvili, Chiara Henselder, Kate Moss, Eman Abdullah AlOmar,
Exploring ChatGPT's code refactoring capabilities: An empirical study,
Expert Systems with Applications,
Volume 249, Part B,
2024,
123602,
ISSN 0957-4174,
https://doi.org/10.1016/j.eswa.2024.123602

El artículo de investigación "Explorando las capacidades de refactorización de código de ChatGPT: un estudio empírico" profundiza en un estudio empírico realizado para evaluar las capacidades y limitaciones de ChatGPT en la refactorización de código, enfocándose en ocho atributos de calidad, siendo estos: complejidad, desempeño, compresibilidad, acoplamiento, reutilización, legibilidad, cohesión y tamaño del diseño. El estudio dividió su investigación en tres partes: evaluar la capacidad de ChatGPT para refactorizar código, preservar el comportamiento y generar documentación. Los investigadores utilizaron ChatGPT para refactorizar un conjunto de datos de 40 segmentos de código Java, y luego el código refactorizado se evaluó utilizando el Detector de errores de programación (PMD) para identificar fallas de programación.

El estudio encontró que ChatGPT ofrecía consistentemente soluciones precisas y rápidas cuando se le solicitaba refactorizar los segmentos de código, con mejoras observadas en 39 de 40 casos. Sin embargo, se observó que las respuestas de ChatGPT eran inconsistentes y su capacidad para comprender errores y operaciones complejas era limitada. A pesar de estas limitaciones, ChatGPT demostró el potencial de brindar sugerencias ventajosas para mejorar la calidad del código. Los resultados del análisis de PMD identificaron violaciones en varias categorías, siendo las más comunes las relacionadas con el estilo del código. Además, el estudio exploró la efectividad de ChatGPT para preservar el comportamiento del código refactorizado y proporcionar documentación para los segmentos de código.

Lo que más interesante me ha resultado es que el estudio reveló que ChatGPT puede proporcionar valiosos cambios de refactorización, particularmente con modificaciones menores como cambiar el nombre de métodos y variables, y mejorar la estructura y limpieza del código. Sin embargo, la confiabilidad de la plataforma fue cuestionada debido a su inconsistencia, comprensión limitada de errores complejos y variaciones en las respuestas al mismo mensaje. A pesar de estas limitaciones, se descubrió que ChatGPT es una herramienta beneficiosa para la programación, especialmente cuando es supervisada por programadores humanos para determinar la importancia de los cambios sugeridos.

Álvaro Moreno Martín

Título: “Rubbing salt in the wound? A large-scale investigation into the effects of refactoring on security”
Revista/Congreso: Empirical Software Engineering
Dirección Web: https://link.springer.com/article/10.1007/s10664-023-10287-x
Autores: Emanuele Iannone, Zadia Codabux, Valentina Lenarduzzi, Andrea De Lucia, Fabio Palomba
Fecha de publicación: 24 de mayo de 2023

El artículo investigado aborda la relación entre la refactorización del software y la seguridad de las aplicaciones. Se realiza una extensa investigación empírica sobre cómo la refactorización, aunque se supone que mejora la calidad del código sin alterar su comportamiento externo, puede influir en el perfil de seguridad del software. Esto se examina a través de un estudio de minería de repositorios de software en tres niveles, considerando 14 tipos de refactorizaciones en 39 proyectos, con un análisis detallado de cómo ciertas refactorizaciones pueden afectar las métricas de seguridad, la deuda técnica relacionada con la seguridad y la introducción de vulnerabilidades conocidas.

Considero este estudio relevante porque profundiza en un área poco explorada pero crítica de la ingeniería de software: el impacto de las refactorizaciones en la seguridad. Las conclusiones del estudio sugieren que, aunque la mayoría de las refactorizaciones tienen un impacto limitado en la seguridad, ciertos tipos como "Inline Method" y "Extract Interface" pueden mejorar aspectos de la seguridad, mientras que otros como "Extract Superclass" y "Pull Up Attribute" tienden a violar prácticas recomendadas de seguridad. Este conocimiento es crucial para desarrolladores y organizaciones que buscan mantener y mejorar la seguridad de sus aplicaciones mientras optimizan su código. Además, el estudio ofrece recomendaciones y una hoja de ruta para investigaciones futuras, contribuyendo a un desarrollo de software más seguro y eficiente.

Pablo Garaulet Tovar

Software Code Refactoring: A Comprehensive Review
Journal of Education and Science (ISSN 1812-125X), Vol: 32, No: 01, 2023 (71-80)
Hiba Muneer Yahya1* , Dujan B. Taha2
(Received November 29, 2022; Accepted February 15, 2022; Available online March 01, 2023)
https://www.iasj.net/iasj/download/dfc9ea6567d57e1a

El artículo “Software Code Refactoring: A Comprehensive Review “ aborda la complejidad del software que ha ido en aumento con el paso de los años y el avance de la tecnología. Esta complejidad, ha provocado un software con peor calidad a nivel de código. Para afrontar esta situación, diversos investigadores han llevado a cabo técnicas de refactoring para hacer el código más comprensible y quitar los “code smells”, para así mejorar la estructura interna del código sin que afecte a la función principal del código.

Esta refactorización también se ha aplicado en la literatura, mediante métodos automáticos basados en reglas, aprendizaje automático o análisis del espacio de software para optimizar métricas.
También menciona que se han desarrollado técnicas y herramientas para facilitar la refactorización automática, que incluyen análisis estáticos y diversas técnicas basadas en reglas, que tienen ventajas y desventajas, y habrá que usar en función de los requisitos del proyecto.

La refactorización automática aunque se encuentra en continuo desarrollo, es un ámbito en el que se requiere más investigaciones para hacerla más eficiente y más automatizada, para reducir el esfuerzo y aumentar la precisión del código.

Me parece un artículo muy interesante, que trata un problema crítico como es la complejidad del código, que va en aumento conforme avanza la tecnología. Desde mi punto de vista, la mayoría de código es muy difícil de comprender y al presentar mucho código se hace muy pesado tratar de entender el comportamiento de éste. Es por ello que la refactorización es de gran ayuda para solventar esta situación, sin afectar a la funcionalidad externa del código, reduciendo el coste, facilitando el mantenimiento del código y aumentando la productividad y la calidad del software. Además, cada vez está más presente el software en todos los ámbitos, trabajos y actividades, por ello como menciona en el artículo habría que invertir en desarrollo de técnicas de refactoring, al igual que se invierte en el avance del desarrollo de software