Universidad Carlos III

Desarrollo de Software

2022-23

Ejercicio guiado IV

Publicaciones Relacionadas con el Refactoring

Curso 2022-23

Ingeniería Informática, Segundo Curso

Adrián Fernández Galán(NIA: 100472182, e-mail: 100472182@alumnos.uc3m.es)

César López Mantecón (NIA: 100472092, e-mail: 100472092@alumnos.uc3m.es)

Prof. María Natividad Carrero de las Peñas

Índice

1. Refactoring for reuse: an empirical study

Alomar, E.A., Wang, T., Raut, V. *et al.* Refactoring for reuse: an empirical study. *Innovations Syst Softw Eng* 18, 105–135 (2022). https://doi.org/10.1007/s11334-021-00422-6

1.1. Resumen

La publicación analiza numerosos proyectos de código abierto con el fin de analizar las estrategias de *refactoring* que siguen los programadores con el fin de aumentar la reusabilidad de su código. Para ello compara el uso de estas prácticas con otros tipos de *refactoring* más utilizados. Destaca además que el *refactoring* con fines de reutilización afecta al código a alto nivel mientras que los demás tipos afectan al código a más bajo nivel.
El estudio concluye que el *refactoring* enfocado a la reutilización afecta sobre todo a las medidas realizadas sobre métodos (e.g. unitest), mientras que apenas tienen efecto sobre otra clase de medidas. También, destaca que esta clase de prácticas apenas son utilizadas frente a otros tipos de refactoring y otros cambios.

1.2. Conclusiones

La publicación nos presenta una visión más real sobre el uso del *refactoring* en proyectos de software. Nos llama especialmente la atención el bajo porcentaje de uso de las prácticas de reutilización, mientras que en la última práctica ha sido uno de los principales pilares en los que se ha basado la extracción de métodos.

2. A tool to test refactoring detection Tools

Leandro, O., Gheyl, R., Teixeira, L., Ribeiro, M., & Garcia, A. (2022, October 5). A Technique to Test Refactoring Detection Tools. *ACMDL*. Retrieved April 27, 2023, from https://dl.acm.org/doi/pdf/10.1145/3555228.3555246.

2.1. Resumen

Esta publicación nos expone una técnica que nos permite analizar y evaluar las distintas herramientas de detección de refactorizaciones, independientemente del IDE empleado. Al comienzo del estudio nos expone la técnica para luego evaluar distintas herramientas de *refactoring*. El estudio concluye con una serie de resultados de los distintos softwares analizados y lo que han mejorado dichas herramientas con respecto al comienzo de la investigación.

2.2. Conclusiones

La publicación nos presenta un punto de vista interesante sobre las herramientas de *refactoring* que no habíamos pensado. Como herramientas, deben ser evaluadas, y la

existencia de esta técnica y los resultados de este estudio nos permiten depositar nuestra confianza en los destinos IDE's a la hora de detectar posibles refactorizaciones.

3. LiveRef: a tool for Live Refactoring Java Code

Fernandes, S., Agular, A., & Restivo, A. (2023, January 5). LiveRef: a Tool for Live Refactoring Java Code. *ACMDL*. Retrieved April 27, 2023, from https://dl.acm.org/doi/abs/10.1145/3551349.3559532.

3.1. Resumen

Esta publicación expone una aplicación para detección de *refactoring* en tiempo real para Java. Su funcionamiento se basa en la identificación, sugerencia y aplicación de extracciones de métodos. Además, realiza un experimento empírico para mostrar su eficacia.

3.2. Conclusiones

La publicación llama especialmente la atención por la cantidad de tiempo que podría ahorrar a los programadores. Después de habernos enfrentado al cuarto ejercicio guiado, hemos invertido grandes cantidades de tiempo en identificación y valoración de extracción de métodos, algo que podría haberse reducido de usar una herramienta como esta. Creemos que el desarrollo de esta clase de aplicaciones es útil para cualquier programador.

4. Green software: Refactoring approach

Mehrotra, D., Nagpal, R., Sharma, R., & Shgal, R. (2022). Green Software: Refactoring approach. *Journal of King Saud University*, *34*(7). Retrieved April 27, 2023, from https://www.sciencedirect.com/science/article/pii/S1319157820305164.

4.1. Resumen

Este artículo expone el impacto del software en el consumo de energía y cómo el *refactoring* puede reducir en gran medida ese consumo. Para ello se utilizan una serie de aplicaciones de Java, midiendo su consumo antes y después de aplicar diversas estrategias de *refactoring*. El estudio concluye que el *refactoring*, además de mejorar la legibilidad y mantenibilidad del código, ayuda a crear software más eficiente.

4.2. Conclusiones

El artículo presenta un punto de vista interesante sobre el *refactoring*, con una conclusión más que sorprendente. Normalmente la eficiencia y legibilidad son cosas que se entienden como inversamente proporcionales. No obstante, este artículo prueba empíricamente que ambos objetivos pueden ir de la mano. Además, siendo el consumo energético una de las principales preocupaciones sociales a día de hoy, como ingenieros tenemos la responsabilidad

de realizar nuestro trabajo de la manera más eficiente posible. A través de este artículo, el *refactoring* queda probado como una manera de contribuir a la causa.