Refactorización
umh2818-TADS

La **refactorización** (del inglés *refactoring*) es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

1 Refactorización de código

En ingeniería del software, el término *refactorización* se usa a menudo para describir la modificación del código fuente sin cambiar su comportamiento, lo que se conoce informalmente por *limpiar el código*. La refactorización se realiza a menudo como parte del proceso de desarrollo del software: los desarrolladores alternan la inserción de nuevas funcionalidades y casos de prueba con la refactorización del código para mejorar su consistencia interna y su claridad. Los tests aseguran que la refactorización no cambia el comportamiento del código.

La refactorización es la parte del mantenimiento del código que no arregla errores ni añade funcionalidad. El objetivo, por el contrario, es mejorar la facilidad de comprensión del código o cambiar su estructura y diseño y eliminar código muerto, para facilitar el mantenimiento en el futuro. Añadir nuevo comportamiento a un programa puede ser difícil con la estructura dada del programa, así que un desarrollador puede refactorizarlo primero para facilitar esta tarea y luego añadir el nuevo comportamiento.

El término se creó como analogía con la factorización de números y polinomios. Por ejemplo, $x^2 - 1$ puede ser factorizado como $(x + 1)(x - 1)$, revelando una estructura interna que no era visible previamente (como las dos raíces en -1 y +1). De manera similar, en la refactorización del software, el cambio en la estructura visible puede frecuentemente revelar la estructura interna "oculta" del código original.

La refactorización debe ser realizada como un paso separado, para poder comprobar con mayor facilidad que no se han introducido errores al llevarla a cabo. Al final de la refactorización, cualquier cambio en el comportamiento es claramente un *bug* y puede ser arreglado de manera separada a la depuración de la nueva funcionalidad.

Un ejemplo de una refactorización trivial es cambiar el nombre de una variable para que sea más significativo, como una sola letra 't' a 'tiempo'. Una refactorización más compleja es transformar el trozo dentro de un bloque en una subrutina. Una refactorización todavía más compleja es remplazar una sentencia condicional if por polimorfismo.

Aunque la limpieza de código se lleva realizando desde hace décadas, el factor clave de la refactorización es realizar de manera intencionada la limpieza separándola de la adición de funcionalidad nueva, usando un catálogo conocido de métodos útiles de refactorización, para después comprobar el código ejecutando las pruebas unitarias, sabiendo que cualquier cambio en el comportamiento significa que se ha introducido un error.

La refactorización es un aspecto importante de la programación extrema.

El libro de Martin Fowler *Refactoring* es la referencia clásica. Aunque la refactorización de código se ha llevado a cabo de manera informal durante años, la tesis doctoral de William F. Opdyke (1993) es el primer trabajo conocido que examina específicamente esta técnica. Todos estos recursos proporcionan un catálogo de métodos habituales de refactorización. Un método de refactorización tiene una descripción de cómo aplicar el método e indicaciones sobre cuándo debería (o no debería) aplicarse.

La refactorización es un concepto tan importante que ha sido identificado por David A. Wheeler como «una de las más importantes innovaciones en el campo del software».[1]

2 Refactorización de otros textos

El término *refactorización* se originó en el ámbito de la programación de ordenadores, pero el concepto se ha aplicado a la modificación de cualquier texto.

En los sitios Wiki, la *refactorización* se refiere al proceso de reescribir y reorganizar el texto para abreviarlo preservando su contenido. Se aplica en especial en las discusiones, que de esta manera pueden ser hechas accesibles para personas que están interesadas en los argumentos ofrecidos en la discusión y en la información que se obtiene de ella, más que en la propia historia de la discusión. Puede ser difícil refactorizar de tal manera que estén de acuerdo todos los participantes de la discusión.

3 Etimología

El primer uso conocido del término *refactorización* en la literatura publicada se encuentra en el artículo *Refactoring: An Aid in Designing Application Frameworks and Evolving Object-Oriented Systems*, *Proceedings of the Symposium on Object Oriented Programming Emphasizing Practical Applications (SOOPPA)* September, 1990, ACM por William F. Opdyke y Ralph E. Johnson.[2] La tesis doctoral de William Opdyke titulada "Refactoring Object-Oriented Framework" (Universidad de Illinois) se publicó en 1992.[3] Tanto el término *refactorización* como la técnica que define se usaban ya con toda seguridad antes.

4 Referencias

[1]

[2]

[3]

5 Enlaces externos

- Libro de Martin Fowler' sobre refactorización

6 Origen del texto y las imágenes, colaboradores y licencias

6.1 Texto

- **Refactorización** *Fuente:* https://es.wikipedia.org/wiki/Refactorizaci%C3%B3n?oldid=89072002 *Colaboradores:* Sabbut, Pilaf, Rsg, Wikier~eswiki, LeonardoRob0t, Guille.hoardings, Taichi, RobotQuistnix, Platonides, Yrbot, YurikBot, Wiki-Bot, GermanX, Eskimbot, Calsbert, JAnDbot, TXiKiBoT, Rei-bot, Idioma-bot, Aibot, VolkovBot, Muro Bot, BotMultichill, El bot de la dieta, PipepBot, Amischol, Alecs.bot, UA31, MastiBot, ArthurBot, SuperBraulio13, Xqbot, Kismalac, EmausBot, ZéroBot, ChuispastonBot, KLBot2, Jarould y Anónimos: 5

6.2 Imágenes

6.3 Licencia del contenido