DOCUMENTAR CÓDIGO: ¿UN TRABAJO O UNA NECESIDAD?

Santiago Alasia, Nicolás Cargnelutti, Diego Hernandez, María Julieta Morellato, Fabian Polanco

Abstract

¿Realmente vale la pena invertir tiempo para documentar el código? Si bien esto se trata de una controversia tan antigua como lo son los lenguajes de programación, el tema no ha perdido vigencia. Más aún, en la modernidad, hoy en día tendemos a relativizar la importancia de la documentación, o como mínimo a simplificar la forma de documentar o a explorar distintas variantes.
Por experiencia propia todo programador sabe que, aquellos proyectos que requieren trazabilidad o bien una continuación de algún desarrollo anterior, sin documentación son imposibles de retomar, a veces son inentendibles o inexplicables por el mismo desarrollador del proyecto. En este contexto se ideó y llevó a cabo una experiencia que, a pesar de su sencillez, permitió obtener resultados muy interesantes.

Elementos del Trabajo y metodología

Cuando se está programando es necesario comentar convenientemente cada una de las clases que brindan la funcionalidad al software. Estos comentarios se incluyen en el código fuente con el objeto de clarificar y explicar el comportamiento de la clase en cuestión: se deben comentar las clases, los paquetes y en definitiva todo elemento que se considere importante.
Esta documentación tiene como objeto hacer más comprensible el código fuente a otros programadores.

Experiencia y resultados

Con fines de probar las ventajas del código documentado sobre el no documentado se ha decidido llevar a cabo la siguiente experiencia.
En primer lugar se seleccionarán alumnos con conocimientos acerca de la tecnología a utilizar (Java SE, EE). Luego se les dará, a la mitad de los alumnos, una porción de código de un programa desarrollado en Java. Al otro grupo se les ofrecerá la misma porción de código pero sin documentar.
Cada uno de los encuestados de ambos grupos tendrán un cierto tiempo para leer el código y tratar de entenderlo.
Finalmente, se realizan las mismas preguntas a cada uno de los participantes de la experiencia. El objetivo de las mismas es poder determinar si los encuestados son capaces de dar una explicación razonable y acertada sobre el funcionamiento del software.

Resultados obtenidos

PROGRA-MADOR	MODO	PORCENTAJE DE COMPRENSIÓN	
		Cantidad	%
1	Documentado	13/16	81,25
2	Documentado	14/16	87,5
3	No documentado	3/16	18,75
4	No documentado	2/16	12,5
5	No documentado	4/16	25%
6	Documentado	12/16	75
7	No documentado	5/16	31.25
8	Documentado	12/16	75
9	No documentado	3/16	18.75
10	Documentado	13/16	81.25

	PORCENTAJE DE COMPRENSIÓN
PROMEDIO DOCUMENTADO	80 %
PROMEDIO NO DOCUMENTADO	21 %

Discusión de resultados

Analizando los resultados se encontró que los desarrolladores que trabajaron con clases no documentadas, lograron comprender el código en menor tiempo, pero no pudieron reconocer la funcionalidad de dicha clase. En cambio, aquellos que trabajaron con clases documentadas, tardaron en entender el código línea por línea pero comprendieron mejor la funcionalidad de cada método.
En otras palabras aquellos desarrolladores que contaron con documentación, demoraron más tiempo pero también lograron una comprensión generalizada del proyecto.

Se observó que los desarrolladores acostumbrados a leer la documentación, ponen énfasis en establecer todas las relaciones lógicas entre las clases que existen en un proyecto para interpretar su funcionalidad, dejando de lado cuestiones de implementación.

Conclusión

Se creía que un comentario debía de representar (en pocas palabras) un contenido adicional y relevante para la comprensión del código en sí mismo. Luego de la experiencia, que ciertamente aportó un resultado interesante, podemos decir que más que una creencia, se trata de un hecho tangible, comprobable y cuantificable.
Más que anotaciones independientes, los comentarios deben servir para aclarar la funcionalidad general de un método o clase, su relación con los demás y aportar contenido a la trazabilidad del código.
El próximo paso a seguir sería la implementación de una prueba más rigurosa para poder darle más relevancia al tema. Y así aspirar a adoptar esta práctica de desarrolladores tanto en ámbitos académicos como así también en organizaciones afines.

Bibliografía

- Brian W. Kernighan y P. J. Plauger, “Elementos de Estilo de Programación”, capitulo 7: Documentación, Ed. Diana, 1980.
- Sara Alvarez, desarrolladora web, “Importancia de la documentación”.
- J.F.Diaz Lic. en Ciencias de la Computación, Cómo Documentar Código con Efectividad.
- José A. Mañas, Dept. de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid, “Documentación de código”.
- Lisardo Fernandez Cordeiro, 1o GII - ETSE Universidad de Valencia, “Herramientas de Documentación de Programas”.

Instituto Universitario Aeronáutico, Facultad de Ingeniería, Departamento de Informática