Clasificación y Evaluación de Métricas de Mantenibilidad Aplicables a Productos de Software Libre

José M. Ruiz, Cristian D. Pacifico, Martín M. Pérez

Facultad de Ciencias de la Administración - Universidad Nacional de Entre Ríos

Av. Tavella 1424, (E3202KAC) Concordia, Entre Ríos, Argentina – Tel: (0345) 423-1400

{jruiz, cripac, marper}@fcad.uner.edu.ar

RESUMEN

Un producto software es "***software libre***" si cumple cierta forma de licenciamiento. Esta forma contempla determinadas libertades que, en teoría, puede ejercer el usuario. De estos derechos, uno significativamente importante es la libertad de *estudiar cómo funciona el programa y adaptarlo a sus necesidades*. Para ejercer esta libertad, no basta con que la licencia exija la disponibilidad del código fuente, sino que también se requiere que el software posea características internas que faciliten la legibilidad y modificación del código.

La *"Mantenibilidad"*[1] *(Maintainability)* es la facilidad con la que se modifica, mejora y/o adapta un producto software. Esta característica es identificada y definida por normas de calidad ampliamente aceptadas, que recomiendan establecer métricas para su evaluación.

La línea de I+D presentada aquí, pretende evaluar y clasificar métricas para la "*mantenibilidad*" aplicables a productos software libre y desarrollar herramientas que las implementen, con el objetivo de favorecer la comprensión y modificación del código.

[1]El término original en inglés es *"Maintainability"*. La traducción al español de la norma ISO/IEC 25010 es el término *"mantenibilidad"*, y en la norma ISO/IEC 9126 se utiliza la frase *"facilidad de mantenimiento"*. El presente trabajo utiliza la traducción adoptada por la ISO/IEC 25010.

***Palabras clave*:** Software Libre, métricas, mantenibilidad, calidad.

CONTEXTO

La presente investigación surge en el marco del proyecto de investigación y desarrollo **PID-UNER 7049** "*Guías para aplicación de Normas de Calidad para los procesos de Ingeniería de Software en productos desarrollados con Lenguajes de Programación Open Source*". El Proyecto de I+D es llevado a cabo en la Facultad de Ciencias de la Administración de la Universidad Nacional de Entre Ríos. Esta unidad académica, mediante varias acciones llevadas en conjunto con cámaras empresariales, empresas y desarrolladores, apoya y fomenta el desarrollo e implementación de Software Libre. Este proyecto de I+D se establece como ámbito [illegible] el desarrollo y redacción de guías y normas para certificación de productos y procesos de Software Libre y Tecnologías Abiertas.

1. INTRODUCCIÓN

Software Libre

Un producto software es considerado "***software libre***" si cumple cierta forma de licenciamiento. Esta forma o "*licencia*"

contempla determinadas libertades que, en teoría, puede gozar el usuario. Estas libertades y derechos son establecidas por elección manifiesta de su autor, en tanto que el producto puede ser copiado, estudiado, modificado, utilizado libremente con cualquier fin y redistribuido con o sin cambios o mejoras [1].

Estas libertades son formalizadas en una licencia que adopta un manifiesto de Software Libre. Una licencia es aquella autorización formal con carácter contractual que un autor de un software da a un interesado para ejercer "*actos de explotación legales*" [2].

El manifiesto más utilizado en proyectos de este tipo es la **Licencia Pública General** de proyecto GNU, o **GNU GPL** (General Public License)**.** La licencia GPL estipula los criterios que se tienen que cumplir para que un programa sea considerado libre. En concreto las cuatro libertades que definen al Software Libre son [3]:

- La libertad de ejecutar el programa para cualquier propósito.
- La libertad de estudiar cómo funciona el programa y de adaptarlo a sus necesidades.
- La libertad de redistribuir copias, para que pueda ayudar al prójimo.
- La libertad de mejorar el programa y poner las mejoras a disposición del público, para que toda la comunidad se beneficie.

Con estas libertades, los usuarios (personas, organizaciones, compañías) controlan el programa/producto y lo que éste hace. Estas libertades son derechos, no obligaciones, cada persona puede elegir no usarlas, pero también puede elegir usar todas ellas.

Cabe destacar que aceptar las libertades del Software Libre no excluye de su uso comercial. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial. El desarrollo comercial del Software Libre ha dejado de ser inusual [4] [5].

Si una empresa es productora de productos de software libre (o los patrocina), busca lograr los beneficios inherentes al modelo de ciclo de vida propuesto por la filosofía del *open source*: lograr que su producto sea ampliamente usado, recibir reportes de errores y modificaciones de su producto y establecer líneas de negocios de servicios de personalización, consultarías y capacitación para uso del producto [6] [7].

En este contexto, la libertad de modificar el código fuente no se limita al derecho del usuario, sino a que es ejercicio fundamental para que el proyecto de software libre perdure y se desarrolle. Por eso, no basta con cumplir con la exigencia de disponibilidad del código fuente inferida en la licencia, sino que también hace falta que el producto posea calidades internas que favorezcan y faciliten la legibilidad y modificación del código.

Mantenibilidad y Mantenimiento del Software

El estudio y la definición de los factores o características que afectan la calidad de un software es una de las áreas más desarrolladas en la Ingeniería de Software [8]. De hecho, se han establecido estándares para fijar las normas de calidad.

Pressman [8] describe la calidad del software como un "*proceso eficaz de software que se aplica de manera que crea un producto útil que proporciona valor medible a quienes lo producen y a quienes lo utilizan*".

La publicación de la norma ISO 9126 [9], en el año 1991, constituye un hito en la definición de estándares de calidad de producto software. Luego, en el año 2001,

fue reemplazado por dos estándares relacionados: el ISO/IEC 9126 [10], que describe las particularidades de un modelo de calidad del producto software; y el estándar ISO/IEC 14598 [11], que abordaba el proceso de evaluación de productos software.

El modelo más actual está representado por la familia de normas ISO/IEC 25000 [12], que es el resultado de la evolución de las normas citadas más arriba. La ISO/IEC 25000 conocida como SQuaRE (*System and Software Quality Requirements and Evaluation*), es una familia de normas que tiene por objetivo la creación de un marco de trabajo común para evaluar la calidad del producto software.

El *modelo de calidad de producto* definido por la ISO/IEC 25010 está compuesto por ocho características de calidad: adecuación funcional, eficiencia de desempeño, compatibilidad, usabilidad, fiabilidad, seguridad, **mantenibilidad** y portabilidad [13].

El IEEE[2] define ***mantenibilidad*** como: "*La facilidad con la que un sistema o componente software puede ser modificado para corregir fallos, mejorar su funcionamiento u otros atributos o adaptarse a cambios en el entorno*" [14]. Esta definición está conectada directamente con la definición del IEEE para ***mantenimiento*** del software: "*es el proceso de modificar un componente o sistema software después de su entrega para corregir fallos, mejorar su funcionamiento u otros atributos o adaptarlo a cambios en el entorno*".

Según el modelo de la ISO/IEC 25000, ***"Mantenibilidad" (Maintainability)*** o ***Facilidad de Mantenimiento*** es la capacidad del producto software para ser modificado efectiva y eficientemente, debido a necesidades evolutivas, correctivas o perfectivas. Esta característica se subdivide a su vez en cinco subcaracterísticas aplicables a un producto software: Modularidad, Reusabilidad, Capacidad para analizarse, Capacidad para modificarse, y Capacidad para probarse.

Estudios previos señalan al mantenimiento posterior como la fase que más recursos requiere a lo largo del ciclo de vida del producto, dos veces superior a los costos de desarrollo [8]. A pesar que el modelo SQuaRE no especifica un conjunto de métricas para evaluar la mantenibilidad de productos software, sí establece las características deseables de dichas mediciones. Tienen que ser objetivas, independientes y reproducibles, expresadas por medio de escalas válidas y suficientemente precisas para apoyar comparaciones fiables.

La ***mantenibilidad*** debe establecerse como objetivo tanto en las fases iniciales del ciclo de vida, para reducir las posteriores necesidades de mantenimiento, como durante las fases de mantenimiento, para reducir los efectos laterales [15]. Existen unos pocos factores que afectan directamente la ***mantenibilidad*** de un producto. Los tres más significativos son:

Proceso de desarrollo: la *mantenibilidad* debe formar parte integral del proceso de desarrollo del software y ser uno de sus principios rectores.

Documentación: En múltiples ocasiones, ni la documentación ni las especificaciones de diseño están disponibles y, por tanto, los tiempos y costos de comprensión, corrección y/o modificación se incrementan.

Comprensión de Programas: La causa básica de los altos costos del mantenimiento es la presencia de obstáculos a la

[2]IEEE: Institute of Electrical and Electronics Engineers.

comprensión del funcionamiento del producto.

Para abordar el concepto de métrica en ingeniería de software es necesario mencionar el concepto de medición, según Fenton: "*La medición es el proceso mediante el cual se asignan números o símbolos a los atributos de las entidades en el mundo real, de manera que se les define de acuerdo con reglas claramente determinadas*" [16]. El IEEE define como métrica a "*una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado*" [14].

Existe la necesidad de medir y de controlar la complejidad del software, y si bien un solo valor de esta métrica de calidad es difícil de derivar, es posible desarrollar medidas de diferentes atributos internos de programa (modularidad efectiva, independencia funcional, entre otros). Estas medidas y las métricas derivadas de ellas brindan un mecanismo consistente y objetivo para valorar la calidad [8].

Se han propuesto métricas diseñadas explícitamente para mantenibilidad que deberían afectar las actividades de mantenimiento. El IEEE. 982.1 [17] sugiere un índice de madurez de software (IMS) que proporcione un indicio de la estabilidad de un producto de software (con base en cambios que ocurran para cada liberación del producto).

Otros estudios [18, 19, 20] han puesto de manifiesto la necesidad de contar con mediciones de la mantenibilidad del producto, en especial si éste se trata de software libre.

2. LÍNEAS DE INVESTIGACIÓN Y DESARROLLO

Este trabajo pretende formalizar procedimientos y normas de aplicabilidad para realizar mediciones orientadas a la mantenibilidad de productos de software libre, permitiendo que las métricas obtenidas ajusten la calidad interna del código fuente con el objetivo de facilitar y asegurar su libertad de modificar el producto.

Puntualmente, este trabajo propone realizar las siguientes tareas:

1. Evaluar de las principales normas y buenas prácticas referidas a la ***mantenibilidad*** de software, en especial los casos de estudios aplicados a software.
2. Clasificar los patrones de medición estudiados, realizando una valoración individual de su impacto en la aplicación de las libertades de Software Libre.
3. Diseñar un núcleo de mediciones/métricas recomendables para aplicar sobre ***mantenibilidad***, definiendo su alcance y aplicación,
4. Desarrollar herramientas para la recolección de métricas de ***mantenibilidad*** en base al diseño anterior.

3. RESULTADOS OBTENIDOS/ESPERADOS

Los objetivos de este trabajo son:

- Lograr un estudio comparativo de mediciones de mantenibilidad teniendo en cuenta el impacto en productos de software libre.
- Obtener una guía para poder aplicar métricas de mantenibilidad a proyectos de software libre.
- Incorporar la guía y herramientas

desarrolladas como guías de aplicaciones de normas de calidad específicas para software libre.

4. FORMACIÓN DE RECURSOS HUMANOS

Este trabajo forma parte de 1 (un) proyecto de tesis de la Maestría en Sistemas de Información. Adicionalmente se prevé contemplar, para los detalles de implementación de herramientas, la realización de 1 (un) Trabajo Final –tesina de grado– para la Licenciatura en Sistemas, ambas carreras dictadas en de la Facultad de Ciencias de la Administración de la UNER

BIBLIOGRAFÍA

[1] R. M. Stallman, Free Software, Free Society: Selected Essays, Boston, Massachusetts, MA: GNU Press, 2002.

[2] F. y. F. H. da Rosa, Guía práctica sobre software libre: su selección y aplicación local en América Latina y el Caribe, Montevideo - Uruguay: UNESCO, 2007.

[3] Fundación del Software Libre de Europa, [En línea]. Available: https://fsfe.org/about/basics/freesoftware.es.html.

[4] T. Wasserman, Building a Business on Open Source Software, 2009.

[5] P. y. A. N. Cooper, «Free and open source software business and sustainability models. OSS Watch,» 2014. [En línea]. Available: http://oss-watch.ac.uk/resources/businessandsustainability.

[6] M. Ferris, «Open source code and business models: More than just a license. OpenSource.com,» 2014. [En línea]. Available: http://opensource.com/business/13/5/open-source-your-code.

[7] L. Ibañez, «Open source economic model: Sell the license or charge a consulting fee? OpenSource.com,» 2013. [En línea]. Available: http://opensource.com/education/13/2/open-source-economic-model.

[8] R. S. Pressman, Ingenieria del Software. Un enfoque práctico. 7 Ed. ISBN: 978-607-15-0314-5, 2005 ed., MCGRAW-HILL, 2010.

[9] ISO 9126, «Software Quality Model,» International Organization for Standardization, Geneva, Switzerland, 1991.

[10] ISO/IEC 9126, «Software engineering - Product quality» International Organization for Standardization, Geneva, Switzerland, 2001.

[11] ISO/IEC 14598-6, «Software engineering - Product evaluation -,» International Organization for Standardization, Geneva, Switzerland, 2001.

[12] ISO/IEC 25000, «Systems and software engineering – Systems and software product Quality Requirements and Evaluation (SQuaRE). Guide to SQuaRE,» International Organization for Standardization, Geneva, Switzerland, 2014.

[13] ISO/IEC 25010, «Systems and software engineering – Systems and software product Quality Requirements and Evaluation (SQuaRE) – System and software quality models.,» International Organization for Standardization, Geneva, Switzerland, 2014.

[14] Engineers, Institute of Electrical and Electronics, «610.12 - IEEE Standard Glossaries of Software Engineering Terminology,» IEEE Computer Society Press, New York, NY. USA, 1990.

[15] B. W. Boehm, J. R. Brown y M. Lipow, Quantitative evaluation of software quality, Los Alamitos, CA, USA: IEEE Computer Society Press, 1976.

[16] N. E. Fenton y S. L. Pfleeger, Software Metrics: A Rigorous and Practical Approach, Boston, MA, USA: PWS Publishing Co., 1997.

[17] Engineers, Institute of Electrical and Electronics, «982.1 - IEEE Standard Dictionary of Measures of the Software Aspects of Dependability,» IEEE Computer Society Press, New York, NY. USA, 1988.

[18] Y. &. X. B. Zhou, Predicting the maintainability of open source software using design metrics, In Wuhan Univ.: J. Nat Sci., 2008, pp. 13:p 14-20.

[19] S. R. S. a. K. C. L. Yu, «Measuring the maintainability of open-source software,» de *International Symposium on Empirical Software Engineering*, 2005.

[20] B. G. J. H. H. E. S. M. Smit, «Maintainability and Source Code Conventions: An Analysis of Open Source Projects,» University of Alberta, 2011.