Evolución de F/OSS: estudio de caso múltiple sobre el crecimiento

Jorge Ramirez[1], Carina Reyes[2], María Laura Massé[3], Nilsa Sarmiento[4]
[1,2,3]Facultad de Ciencias Exactas, Universidad Nacional de Salta, Centro de Investigación y Desarrollo en Informática Aplicada – Avda. Bolivia 5150 – Salta
[4] Instituto De Investigaciones En Energía No Convencionales (INENCO), Av. Bolivia 5150, Salta, Argentina
[1,2,3] {ramirezj, reyes, mlmassep}@cidia.unsa.edu.ar, [4]nilsamsarmiento@gmail.com

Abstract: la disponibilidad de datos propia de los proyectos de Software Libre y de Código Abierto (F/OSS por sus siglas en inglés) posibilita realizar investigaciones reproducibles, al tiempo que habilitan un tratamiento estadístico poco frecuente en la investigación en ingeniería de Software. En este trabajo analizamos el ritmo de crecimiento del tamaño de un grupo de productos F/OSS a lo largo de sucesivas versiones, e indagamos sobre la posible relación entre ese crecimiento y un indicador de acoplamiento.

Palabras Clave: Software Libre y de Código Abierto, Evolución del Software, Métricas

1 Introducción

El estudio del Software Libre y de Código Abierto (F/OSS por sus siglas en inglés) se ha convertido en un campo de investigación de interés creciente para la ingeniería de Software. La disponibilidad de información públicamente accesible sobre un gran número de proyectos de software ofrece la oportunidad de abordar investigaciones [1] [2] en las que los estudios pueden ser repetidos y comparados por diferentes investigadores [3].

La investigación sobre la evolución de software es uno de los tópicos sobre los que se ha producido mayor cantidad de trabajos a partir del estudio de F/OSS [4] [5] [6].

El conocimiento sobre la forma en que el software evoluciona y los factores que intervienen en ese proceso son relevantes a para orientar distintos aspectos atinentes a la administración de proyectos de software [7]. Los sistemas de software sufren cambios para poder seguir cumpliendo con las funciones para las que fueron creados. El estudio de las distintas versiones de software llevó a esos autores a elaborar una serie de afirmaciones conocidas como Leyes de la Evolución del Software.

Las investigaciones de Lehman y otros[8] , así como diversos trabajos posteriores [5] [9] [10] [11], pusieron de relieve la importancia de conocer las características generales de la evolución del software, así como los factores que inciden en ella. Lehman observó que los sistemas que modelan operaciones humanas deben

modificarse para contemplar nuevos requerimientos, cambios organizacionales y tecnológicos, entre otros; esto implica que sus funciones aumentan y con ello la posible complejidad de los sistemas de software, lo que dificultaría nuevos cambios.

La formulación de las leyes de la evolución fueron modificándose con el tiempo [12]. A medida que los enunciados fueron más precisos, fue posible realizar más investigación empírica sobre estos temass; no obstante, las limitaciones en cuanto al acceso a los datos y a la divulgación de los mismos, como las normas de confidencialidad, dificultan un abordaje estadístico más profundo. [12] [13]

Las leyes que se relacionan con el presente trabajo son: la II, de complejidad creciente, que sostiene que los sistemas evolutivos aumentan progresivamente su complejidad si no se adoptan medidas para mantenerla o reducirla; la VI, que afirma que el software debe aumentar sus funciones a lo largo del tiempo para seguir siendo útil; y las VII, que plantea que la calidad tiende a descender, a menos que se adapte y mantenga de acuerdo con el contexto en el que opera.

Teniendo en cuenta las observaciones de los párrafos anteriores, nos planteamos aquí explorar la evolución de un conjunto de proyectos de F/OSS en base a la información públicamente disponible en cualquier emprendimiento de ese tipo, es decir, el código fuente y la fecha de publicación de cada versión; esos datos posibilitan el análisis del código fuente mediante métricas de producto y la visualización de la evolución cotejando cambios entre una versión y otra.

Específicamente, analizamos en este trabajo la evolución de un conjunto de proyectos F/OSS a fin de comprobar:

- Si se observa que el aumento del tamaño del software tiende a ser menor en las sucesivas versiones, lo que podría relacionarse con el aumento de la complejidad a lo largo del proceso evolutivo (en consonancia con la ley II de la evolución)
- Si existen indicios de que el aumento del tamaño del software de una versión a la siguiente está relacionada con el acoplamiento entre clases.

El presente trabajo se organiza de la siguiente forma: en la sección siguiente se resumen trabajos relacionados con la temática abordada; a continuación, se describe la metodología utilizada, detallando las herramientas utilizadas para la extracción de métricas y el procesamiento de los datos; posteriormente, se presentan resumidamente los datos obtenidos; en la sección siguiente se discuten los resultados; finalmente, se exponen las conclusiones y se proponen trabajos a futuro.

2 Trabajos Relacionados

Como señalamos en la introducción, existe un cuerpo creciente de investigaciones sobre el F/OSS en los que se aborda la evolución del software y en particular las características del aumento del tamaño a lo largo de las diferentes versiones.

Godfrey y Tu [14] publicaron en 2000 una investigación sobre la evolución del núcleo de Linux, cuyos resultados contrastaban con las investigaciones realizadas por Lehman y otros en el ámbito del software privativo o tradicional. Estos autores comprobaron que el tamaño del núcleo del sistema operativo libre crece a tasas supralineales, lo que contradice el enunciado de la ley de Lehman y en particular los modelos propuestos por Turski [15] [16].

El trabajo de Godfrey y Tu fue repetido, con algunas variantes, por Robles y otros [17] , extendiéndose la investigación a un conjunto amplio de proyectos de F/OSS [18] que posibilitan una visión más amplia.

Como señalamos en la introducción, las características del Software Libre han posibilitado un número creciente de investigaciones, muchas de ellas vinculadas con la evolución. Se han realizado numerosos estudios de caso [19] [20] [21] , análisis comparativos de diferentes proyectos [13] [22] , análisis de muestras amplias con procesamiento estadístico [23] [13] , etc.

El trabajo actual retoma investigaciones anteriores, en particular el estudio de caso de la evolución del proyecto Sweet Home 3D [24] y el análisis de la evolución de un conjunto de indicadores de diseño en un grupo de proyectos de F/OSS [25] . Aunque la temática es distinta, el análisis estadístico que nos llevó a proponer umbrales para promedios de métricas en proyectos F/OSS [26] comparte con este trabajo un marco conceptual y algunas de las herramientas utilizadas.

3 Metodología

Nuestro objetivo en este trabajo apunta a conocer con mayor profundidad algunos aspectos de la evolución de un conjunto de productos de F/OSS. En ese sentido, el enfoque que adoptamos es el de realizar una serie de estudios de caso entendidos como un método de investigación de "*una instancia (o un pequeño número de instancias) de un fenómeno contemporáneo de ingeniería de software dentro de su contexto en la vida real"* [27] [28].

Se trata de un estudio de carácter exploratorio, donde buscamos obtener conclusiones preliminares susceptibles de profundizar y contrastar en estudios posteriores.

Para este trabajo analizamos todas las versiones publicadas y disponibles para un conjunto de 13 proyectos de software. La selección se limitó a desarrollos realizados en Java, a fin de descartar que el lenguaje de desarrollo pudiera incidir en los resultados.

Para cada versión publicada de cada proyecto se obtuvieron a partir del código fuente las siguientes métricas: el tamaño medido en líneas de código (LOC) y la dispersión de acoplamiento (DA), definida como la proporción de las llamadas a métodos diferentes (es decir, las llamadas a cada método se cuentan una sola vez)

respecto del total de métodos definidos en toda el código de la versión. Ambas métricas se extrajeron mediante la herramienta iPlasma [29], que puede descargarse de http://loose.upt.ro/reengineering/research/iplasma. La selección de la métrica de acoplamiento tuvo en cuenta su carácter global como atributo de diseño [30] , no dependiente del tamaño, siendo además una de las medidas que analizamos en trabajos anteriores [26].

No existe acuerdo en cuanto a las métricas que conviene utilizar en este tipo de investigaciones; no obstante cabe destacar que Lehman sugiere la utilización del número de secuencia de versión para analizar la evolución y que las métricas de tamaño SLOC (líneas de código fuente), LOC y cantidad de módulos parece exhibir una importante correlación entre ellas[31] [32] .

La información obtenida se organizó en planillas de cálculo mediante LibreOffice Calc, agregando a cada versión la fecha de publicación para poder considerar el intervalo entre cada versión y la siguiente.

La lista de proyectos analizados se detalla en la tabla 1, señalándose además el período de tiempo en que fueron publicadas las versiones estudiadas y la cantidad de versiones.

Tabla 1: Lista de proyectos analizados

Proyecto	Intervalo	Versiones
PDFBox	2010-2014	18
JFreeChart	2004-2009	20
Art Of illusions	2002-2010	28
FreeMind	2000-2014	41
ICEPDF	2009-2014	25
Lwjgl	2002-2013	49
Batik	2001-2008	19
Chemistry Development Kit	2007-2014	36
Vuze	2010-2015	28
Sweet Home 3D	2007-2016	36
HSQLDB	2001-2014	21
Dynamic Reports	2010-2015	48
Jxplorer	2002-2014	42

En la planilla de cálculo se agregó una columna con el número de días entre una versión y la anterior, y el aumento de tamaño en LOC por día; para cada proyecto se

calculó el coeficiente de Pearson a fin de observar la posible existencia de correlación entre el intervalo de tiempo entre la publicación entre versiones y la dispersión de acoplamiento.

A continuación, se realizaron gráficas para visualizar el incremento de LOC por día entre versiones; para explorar la posible existencia de tendencias en cuanto al ritmo en de crecimiento a lo largo de las sucesivas versiones, se agregó la línea de tendencia calculada por LibreOffice Calc[33].

El empleo de herramientas de libre disponibilidad, como iPlasma y LibreOffice, favorece la reproducibilidad y la extensibilidad de la experiencia.

4 Resumen de los Datos Obtenidos

Los datos obtenidos de acuerdo a la metodología expuesta en el apartado anterior se organizaron en gráficas y tablas.

En la Tabla 2 resumimos la información observada en cada proyecto. La clasificación de la tendencia se basa en la observación visual

Tabla 2: Coeficiente de Pearson y caracterización de la tendencia en la variación del incremento del tamaño por día entre versiones, para cada proyecto analizado.

Proyecto	Coeficiente de Pearson entre DA y +LOC/día	Tendencia Observada
PDFBox	0,171	Levemente Ascendente
JFreeChart	-0,143	Descendente
Art Of illusions	-0,015	Descendente
FreeMind	-0,063	Constante
ICEPDF	0,094	Descendente
Lwjgl	-0,05	Constante
Batik	0,274	Descendente
Chemistry Development Kit	0,5	Ascendente
Vuze	0,021	Descendente
Sweet Home 3D	-0,335	Descendente
HSQLDB	0,53	Levemente Ascendente
Dynamic Reports	-0,05	Descendente
Jxplorer	0,11	Constante

En este trabajo se contemplaron 411 versiones de los 13 proyectos analizados

5 Análisis de los datos

Los gráficos elaborados de acuerdo con la metodología descrita en la sección 3 permiten observar que la mayoría de los proyectos analizados (7 sobre 13) muestran una tendencia descendente en cuanto a la incremento de las líneas de código por día entre versiones, según se resume en la tabla 3.

En 3 de los proyectos no se visualizan variaciones que puedan constituir tendencias en cuanto al crecimiento del tamaño del software.

Si bien en otros 3 parece verificarse una tendencia creciente, es decir, de mayor incremento de tamaño por día de una versión a la siguiente, sólo en un proyecto se visualiza esta tendencia con claridad: Chemistry Development Kit (ilustración 1); esta aplicación lleva ya 9 años de desarrollo.

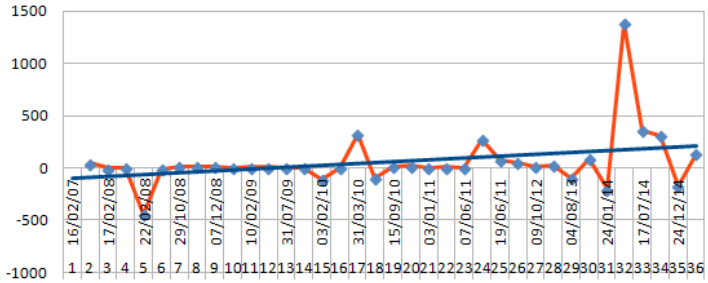

Ilustración 1: Chemistry Development Kit exhibe una tendencia a aumentar el crecimiento en LOC por día

Por razones de brevedad, incluimos sólo un gráfico que muestra claramente una tendencia decreciente (ilustración 2) y el que parece aumentar su crecimiento con el tiempo.

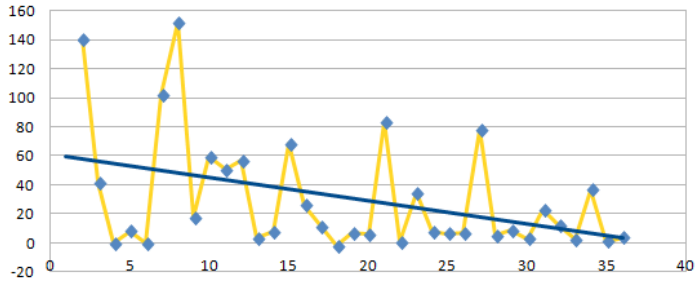

Ilustración 2: Sweet Home 3D muestra una tendencia a disminuir el ritmo de crecimiento en LOC por día

En cuanto a la relación entre el acoplamiento y el ritmo de crecimiento del tamaño, sólo en dos de los proyectos aparece una correlación que podría indicar una vinculación relevante, al menos en término de relaciones lineales: HSQLDB y Chemistry Development Kit. No obstante, cabe destacar que el último ha publicado una cantidad de versiones importante (36), en tanto que en los demás proyectos que superan las 30 versiones los valores del coeficiente son muy cercanos a cero.

6 Conclusiones y trabajos futuros

El trabajo realizado permite observar que, al menos en el conjunto de aplicaciones analizado, se verifica un paulatino descenso en el ritmo de crecimiento de productos F/OSS a lo largo de su evolución. Este resultado es compatible con las afirmaciones condensadas en las llamadas Leyes de la Evolución del Software.

Sin embargo, se observa también que en algunos proyectos no se verifica un descenso en ese ritmo; en un caso, inclusive, se observa un aumento en esa tasa. Al tratarse de una aplicación que lleva muchos años de desarrollo y que ha producido más de una treintena de versiones, ese comportamiento es llamativo; sería de esperar que un programa alcance cierta estabilidad en cuanto a los requerimientos y demás factores que afectan al proceso evolutivo, por lo que el aumento del ritmo del crecimiento amerita indagar sobre los factores que están incidiendo en ese comportamiento.

Los proyectos que exhiben una tendencia constante pueden relacionarse con la realización de trabajos por parte del equipo de desarrollo orientados a reducir las dificultades de modificación del software para futuras versiones; esto sería compatible con la formulación de las leyes II y VII de la Evolución del Software. De todos modos, requeriría una investigación más profunda para verificar esa hipótesis, o para detectar otros factores que podrían estar en juego.

Respecto de la relación entre la medida de acoplamiento “dispersión de

acoplamiento" y el ritmo de crecimiento del software entre versiones, los datos parecen descartar que exista una correlación -al menos de tipo lineal. Si en algunos casos esta vinculación aparece como significativa, podría deberse a características específicas del proyecto sobre las cuales hace falta mayor investigación.

En cuanto a la validez de estos resultados, cabe considerar que la métrica LOC como medida de tamaño es una de las más utilizadas y se ha verificado en otros estudios una fuerte correlación con otras métricas usadas en la investigación sobre la evolución del software[31]. En cambio, la métrica que elegimos para describir el acoplamiento podría no estar captando aspectos de la complejidad que tienen mayor incidencia en la modificabilidad o evolutividad del software (entendida como un atributo que evalúa la facilidad o dificultad de que un producto evolucione).

En futuros trabajos sería pertinente indagar en las relaciones con otras métricas, y adoptar un enfoque multivariante que permita incorporar factores que tienen incidencia en el proceso evolutivo del F/OSS, como la cantidad de desarrolladores, la popularidad, características propias del dominio al que se aplican, etc.

Referencias

1. Kon, F., Meirelles, P., Lago, N., Terceiro, A.S., Chavez, C., Mendonça, M.G.: Free and Open Source Software Development and Research: Opportunities for Software Engineering. In: SBES. pp. 82–91 (2011).
2. von Krogh, G., von Hippel, E.: The Promise of Research on Open Source Software. Manage Sci. 52, 975–983 (2006).
3. González-Barahona, M., Robles, G.: On the reproducibility of empirical software engineering studies based on data retrieved from development repositories. Empir. Softw. Eng. 17, 75–89 (2012).
4. Bauer, V., Eckhardt, J., Hauptmann, B., Klimek, M.: An Exploratory Study on Reuse at Google. In: Proceedings of the 1st International Workshop on Software Engineering Research and Industrial Practices. pp. 14–23. ACM, New York, NY, USA (2014).
5. Kemerer, C.F., Slaughter, S.: An Empirical Approach to Studying Software Evolution. IEEE Trans. Softw. Eng. 25, 493–509 (1999).
6. Pei-Breivold, H., Chauhan, M.A., Babar, M.A.: A Systematic Review of Studies of Open Source Software Evolution. In: 17th Asia Pacific Software Engineering Conference (APSEC), IEEE (2010).
7. Moazeni, R., Link, D., Boehm, B.: Lehman's Laws and the Productivity of Increments: Implications for Productivity. In: 2013 20th Asia-Pacific Software Engineering Conference (APSEC). pp. 577–582. IEEE (2013).
8. Lehman, M.M., Ramil, J.F.: An approach to a theory of software evolution. In: IWPSE '01: Proceedings of the 4th International Workshop on Principles of Software Evolution. pp. 70–74. ACM, New York, NY, USA (2001).
9. Ciraci, S., Broek, P.V.D.: Evolvability as a Quality Attribute of Software Architectures. In: International ERCIM Workshop on Software Evolution 2006 (2006).
10. Crnkovic, I.: Predictability and evolution in resilient systems. In: Software Engineering for Resilient Systems. pp. 113–114. Springer (2011).
11. Barais, O., Lawall, J., Meur, A.-F.L., Duchien, L.: Software Architecture Evolution. In: Mens, T. and eds, S.D. (eds.) Software Evolution. pp. 233–262. Springer Verlag (2008).
12. Herraiz, I., Rodriguez, D., Robles, G., Gonzalez-Barahona, J.M.: The Evolution of the

Laws of Software Evolution: A Discussion Based on a Systematic Literature Review. ACM Comput Surv. 46, 28:1–28:28 (2013).
13. Herraiz, I.: A statistical examination of the evolution and properties of libre software. In: ICSM. pp. 439–442 (2009).
14. Godfrey, M.W., Tu, Q.: Evolution in Open Source Software: A Case Study. In: In Proceedings of the International Conference on Software Maintenance. pp. 131–142 (2000).
15. Turski, W.M.: Reference Model for Smooth Growth of Software Systems. IEEE Trans Softw Eng. 22, 599–600 (1996).
16. Turski, W.M.: The reference model for smooth growth of software systems revisited. IEEE Trans. Softw. Eng. 28, 814 (2002).
17. Robles-Martínez, G., González-Barahona, J.M., Centeno-González, J., Matellán-Olivera, V., Rodero-Merino, L.: Studying the evolution of libre software projects using publicly available data. In: 3 rd Workshop on Open Source Software Engineering. p. 111 (2003).
18. Robles, G., Amor, J.J., Gonzalez-Barahona, J.M., Herraiz, I.: Evolution and Growth in Large Libre Software Projects. In: IWPSE '05: Proceedings of the Eighth International Workshop on Principles of Software Evolution. pp. 165–174. IEEE Computer Society, Washington, DC, USA (2005).
19. González-Barahora, J., Robles, Gregorio, G., Herraiz, Israel, I., Ortega, Felipe, F.: Studying the laws of software evolution in a long-lived FLOSS project. J. Softw. Evol. Process. 26, 589–612 (2014).
20. Capiluppi, A., Morisio, M., Ramil, J.F.: Structural Evolution of an Open Source System: A Case Study. In: IWPC. pp. 172–182 (2004).
21. Terceiro, A., Chavez, C.: Structural Complexity Evolution in Free Software Projects: A Case Study. In: Quality and Architectural Concerns in Open Source Software (2009).
22. Neamtiu, I., Xie, G., Chen, J.: Towards a better understanding of software evolution: an empirical study on open-source software. J. Softw. Evol. Process. 25, 193–218 (2013).
23. Koch, S.: Software evolution in open source projects—a large-scale investigation. J Softw Maint Evol. 19, 361–382 (2007).
24. Ramirez, J., Gimson, L., Gil, G.: Evaluación de la Evolución del Diseño en F/OSS:un Caso de Estudio. In: VII Workshop Ingeniería de Software - XVI Congreso Argentino de Ciencias de la Computación (2010).
25. Ramirez, J., Reyes, C., Gil, G.: Métricas de Código fuente y Evolución de F/OSS: un estudio exploratorio. In: 42 Jornadas Argentinas de Informática. pp. 198–209. , Córdoba (2013).
26. Ramirez, J., Gil, G., Reyes, C.: Umbrales sugeridos para promedios de métricas de diseño de una aplicación en Java, (2015).
27. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw Engg. 14, 131–164 (2009).
28. Ramirez, J.: F/OSS para el reuso: Métricas, Desarrollo de Herramientas y Marco para su Evaluación, http://hdl.handle.net/10915/48176, (2015).
29. Marinescu, C., Marinescu, R., Mihancea, P.F., Ratiu, D., Wettel, R.: iPlasma: An Integrated Platform for Quality Assessment of Object-Oriented Design. In: ICSM (Industrial and Tool Volume). pp. 77–80 (2005).
30. Lanza, M., Marinescu, R., Ducasse, S.: Object-Oriented Metrics in Practice. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2005).
31. Herraiz, I., Robles, G., Gonzalez-Barahona, J.M.: Comparison between SLOCs and number of files as size metrics for software evolution analysis. 2011 15th Eur. Conf. Softw. Maint. Reengineering. 0, 206–213 (2006).
32. Thomas, L.G., Schach, S.R., Heller, G.Z., Offutt, J.: Impact of release intervals on empirical research into software evolution, with application to the maintainability of Linux. Softw. IET. 3, 58–66 (2009).

33. Document Fundation: Curvas de Regresión en LibreOffice, https://help.libreoffice.org/Chart/Trend_Lines/es#La_ecuaci.C3.B3n_de_regresi.C3.B3n_lineal.