El Software Abierto y su Impacto en la Enseñanza de la Computación

Jorge Aguirre* Nazareno Aguirre†
Marcelo Arroyo† Javier Smaldone†

Resumen

En este trabajo se comenta el impacto que la existencia de software abierto está produciendo en la enseñanza de la Computación, y se comenta la forma en la cual la educación puede beneficiarse a partir de esta nueva y creciente filosofía de desarrollo de software.

1 Introducción

En los últimos años, una nueva forma de desarrollo de software ha tenido un fuerte crecimiento: *el software abierto*[3]. Esta novedosa metodología de desarrollo fue iniciada dentro del ámbito académico, para luego extenderse hasta abarcar algunas áreas de la industria [5].

Uno de los medios principales que ha beneficiado tanto la distribución como el desarrollo de software abierto es *Internet*. Ésta permite que grupos de desarrolladores distribuidos alrededor del mundo puedan trabajar en conjunto; de hecho, esta forma de trabajo ha dado lugar a un estilo de desarrollo verdaderamente revolucionario: el *modelo del bazar*[2]. Este modelo consiste en la producción sucesiva de prototipos abiertos, construidos en paralelo por individuos con diferentes intereses y objetivos; contrariamente a lo que se cree, ésta metodología de trabajo ha demostrado ser especialmente beneficiosa. El ejemplo más sorprendente de su utilización es el desarrollo del conocido sistema operativo *GNU/Linux*.

Varias entidades han promovido la construcción y utilización de software abierto. Entre ellas podemos citar a *GNU (**G**NU's **N**ot **U**nix)* [4] y la *Free Software Foundation*. Estas entidades, lideradas por grandes desarrolladores, como Richard Stallman (fundador de GNU), han impulsado y organizado equipos

*Área de Computación. Facultad de Ciencias Exactas, Físico-Químicas y Naturales. Universidad Nacional de Río Cuarto. Departamento de Computación. Facultad de Ciencias Exactas, Físicas y Naturales. Universidad de Buenos Aires. e-mail: jaguirre@dc.uba.ar

†Área de Computación. Facultad de Ciencias Exactas, Físico-Químicas y Naturales. Universidad Nacional de Río Cuarto. Ruta Nacional Nro. 36, Km. 601. Río Cuarto, Córdoba. Argentina. e-mail: {naguirre,marroyo, javier}@dc.exa.unrc.edu.ar

de programadores e investigadores, lo cual ha permitido que una importante cantidad de software abierto de gran calidad esté disponible en la actualidad.

Desafortunadamente, aún el software abierto no goza de una aceptación o utilización masiva en estas latitudes, a pesar de que en muchos casos supera ampliamente a productos comerciales equivalentes, de amplia aceptación. Un claro ejemplo de esto es el compilador C de GNU (*GCC*), el cual cuenta con ciertas características como son portabilidad, calidad del código generado, compilación cruzada, etc., no encontradas en otros compiladores C comerciales.

En este trabajo pretendemos describir el impacto que el software abierto puede tener en la enseñanza de la Computación, y además aclarar algunos de los mitos que con frecuencia afectan a la aceptación y aprovechamiento del software libre.

2 Qué es el *software abierto*?

Existen distintos tipos de distribución de software, pero, a nuestro entender, sólo algunos de ellos se ajustan a la definición de *software abierto* [3]. Las características que nosotros consideramos esenciales para que un producto sea considerado abierto son las siguientes:

- Disponibilidad del código fuente, es decir, el código fuente del producto es incluído en la distribución del mismo.
- No deben existir restricciones en la modificación del código fuente por parte de los usuarios.
- Debe permitirse la redistribución del producto y su uso ilimitado.

Algunas de las licencias de software libre más populares son las de BSD y FSF/GNU. Esta última (conocida como *General Public Licence*) exige que el código modificado por el usuario sea distribuido en los mismos términos.

3 Mitos sobre el software abierto

Existen numerosos mitos que han inhibido la adquisición y utilización de software abierto. Enunciaremos aquí algunos de ellos, y explicaremos por qué estas críticas no pueden ser consideradas si se analizan seriamente.

3.1 Carencia de Calidad

La creencia de que un producto gratuito no alcance ciertas medidas de calidad es bastante común. Sin embargo, muchos desarrolladores de software libre provienen del ámbito académico, lo cual afecta considerablemente a los productos que desarrollan, y aunque su fin no sea obtener retribuciones económicas, suelen tomarse con total seriedad el proceso de desarrollo de productos de software (al parecer, con mucha mayor seriedad que algunas empresas comerciales).

Podemos mencionar algunos ejemplos de software libre ampliamente utilizado en el mundo (e incluídos en muchas distribuciones de software): Servidor de Web Apache, SendMail, XFree86, Emacs, GCC, GhostScript, Kernel de GNU/Linux y varias implementaciones de TCP/IP.

3.2 Carencia de Documentación

Este punto carece de total fundamento. Si comparamos cualquier producto de software abierto con los productos comerciales, encontraremos que el primero por lo general dispone de una gran cantidad de documentación y descripción técnica detallada (además, claro está de la inclusión del código fuente). Más aún, muchas veces los usuarios de software comercial se ven obligados a comprar la documentación de los programas que han adquirido, además de no conocer detalles técnicos sobre su implementación.

3.3 Carencia de Soporte Técnico

Quienes afirman esto seguramente no han tenido que recurrir al soporte técnico de algunas empresas comerciales. Aún en casos en los cuales un usuario adquiere productos de software de precios elevados, puede verse completamente desprovisto de soporte técnico.

En el caso del software abierto, con frecuencia cualquier usuario puede conseguir soporte técnico contactádo al grupo de desarrollo del producto que utiliza, lo cual es en general muy sencillo gracias a Internet. Los desarrolladores de software abierto suelen colaborar con todo interés en los problemas que los usuarios puedan tener en la utilización de sus programas. Esta actitud está motivada por el interés de conservar a sus usuarios, los cuales permiten que los programas sigan evolucionando con sus aportes de sugerencias, detección de errores, pedidos de nuevas funcionlidades, etc. [2]

3.4 Carencia de Confiabilidad

En muchos casos, los desarrolladores de software abierto son más sinceros que algunas empresas comerciales, desde el punto de vista de que si el producto llegase a tener algunas falencias (bugs), éstas son documentadas e informadas a los usuarios (tal vez sea debido a la falta de presión empresarial y/o comercial). Más aún, los desarrolladores piden a sus usuarios que les informen de los bugs detectados. Esto hace que exista una retroalimentación y el software evolucione y se realicen las correcciones en nuevas versiones (y posibles extensiones) de manera verdaderamente ágil. Valga como ejemplo de esto el hecho de que la empresa Digital Domain, encargada de la realización de los efectos visuales de la película Titanic utilizó para dicha producción sistemas Alpha de Digital, corriendo GNU/Linux, como alternativa a los costosos sistemas Silicon Graphics. Además, fruto de esta experiencia, Digital Domain realizó importantísimos aportes a la evolución de GNU/Linux, y en especial a la migración de dicho Sistema Operativo a plataformas Alpha. Además debemos reconocer que muchos

productos que hoy están a la cabeza en el mercado tienen serios problemas de inestabilidad.

3.5 Deficiencias en Seguridad

Uno de los aspectos que más discusiones despierta con respecto al software libre es el relacionado con la seguridad. Este es un factor especialmente importante en el caso de los Sistemas Operativos libres (FreeBSD y GNU/Linux). Muchos hablan de que dichos Sistemas Operativos son desarrollados por "hackers" (confundiendo este término con el de "crackers" o "preackers") y que estos, intencionadamente, dejan agujeros en la seguridad en los programas [6]. Esto no solamente es absurdo, debido a que el código fuente es totalmente accesible, razón por la cual cualquier trampa de este tipo podría ser detectada por cualquier persona, sino que además, debido a la retroalimentación de la que hablabamos anteriormente, los problemas de seguridad encontrados son rápidamente informados a los desarrolladores (muchas veces, inclusive, junto con la posible solución). Esto hace posible mantener permanentemente actualizados los sistemas, simplemente revisando periodicamente los foros de discusión sobre seguridad y aplicando los parches que se encuentran disponibles via Internet. Un ejemplo de un sistema con altos requerimientos de seguridad que utiliza Software Libre es el sistema de distribución de correo postal de los Estados Unidos de América, que utiliza servidores corren GNU/Linux.

La cantidad de entidades que han comenzado a usar sotfware abierto está teniendo un crecimiento sorprendente. Basta con ver la cantidad de usuarios del sistema operativo GNU Linux. En [5] se hacen referencia a empresas que utilizan plataformas de este tipo.

4 Beneficios del Software Abierto para la Enseñanza

La existencia de software abierto puede beneficiar a la enseñanza de la Computación en muchos aspectos. En primer lugar, debe notarse que, en países como el nuestro, la posibilidad de trabajar con herramientas de software gratuitas (de uso ilimitado) permite implementar cursos y talleres a un costo de software nulo. Además, gran parte del software abierto no exige costosas plataformas de hardware, y por lo tanto pueden aprovecharse los recursos de hardware de mejor manera que en el caso del software comercial.

Algunas aŕeas de la Computación se ven especialmente beneficiadas por la existencia de numerosos productos de software abierto. Enumeraremos algunas de estas áreas, y nombraremos algunos productos de software que las apoyan:

- Lenguajes de Programación: La enseñanza de lenguajes de programación se ve beneficiada por la gran cantidad de compiladores e intérpretes, para casi todos los paradigmas existentes.

- Sistemas Operativos: La aparición de *Minix*[7] despertó un enorme interés , ya que permitía realizar prácticas reales sobre los distintos aspectos de

desarrollo de un sistema operativo. En la actualidad, esta área encuentra diferentes productos que pueden ser utilizados como herramientas de apoyo en el dictado de cursos, y realizar trabajos prácticos con sistemas operativos reales como FreeBSD y GNU/Linux, gracias a la disponibilidad del código fuente y documentación. Además de poder ofrecer y mostrar herramientas para implementar computación paralela como son PVM (Parallel Virtual Machine), MPI, La Laguna C, etc.

- Bases de Datos: Productos como Posgres, Ingres, MySQL, CORAL, etc. permiten que la enseñanza de esta asignatura se vea fortalecida, pues ayudan a abarcar tanto el modelo relacional como las extensiones más comunes de éste (bases de datos deductivas, orientadas a objetos).

- Compiladores: La mayoría de las herramientas más utilizadas para el desarrollo de compiladores cuentan con versiones de distribución libre. Entre ellas podemos nombrar GNU Bison lex-YACC, Eli, etc.

- Ingeniería de Software: En esta área existen herramientas de soporte en el proceso de desarrollo de software para distintas metodologías formales e informales, como por ejemplo, FreeCASE, ErgoUML, Z, Object-Z, etc.

Otra ventaja considerable del uso de software abierto en la educación de computación es independizar la formación de los estudiantes de determinada plataforma de hardware o de software. Por ejemplo, GNU/Linux es el sistema operativo más portable de la actualidad (existen versiones para plataformas Intel, PowerPC, Motorola, Sparc, Alpha, MIPS, etc.) y en el caso del servidor HTTP Apache, se encuentra disponible para Windows NT, Solaris, SCO Unix, GNU/Linux, FreeBSD, entre otros sistemas operativos. Esto subraya la importancia de un concepto fundamental de todo proyecto de software: la *portabilidad*, que parece haberse dejado un poco de lado en ciertos productos de desarrollo de software comerciales.

Además se pueden realizar interesantes estudios y experimentos sobre productos reales, al disponerse de su código fuente y su documentación.

5 Conclusiones

Creemos que el software abierto permite ofrecer un mayor soporte para la enseñanza de la computación, ya que estamos convencidos que los prácticos de laboratorio y los talleres son un complemento muy importante para afianzar los conceptos teóricos desarrollados en las diferentes asignaturas. Si estimamos el costo del software necesario para la implementación de una carrera de computación, seguramente ascendería a una suma muy importante. Además deberíamos contar con las actualizaciones que son cada vez mas necesarias en el mundo informatizado actual. En muchos casos, en nuestras latitudes estos costos se descartan porque la piratería de software es algo muy común.

En la actualidad muchas Universidades han considerado el uso de sotfware libre (un claro ejemplo es la decición del gobierno mejicano en utilizar GNU/Linux

en todos sus niveles educativos), pero esta tendencia también está alcanzando a la industria (como en el caso mencionado anteriormente de los efectos de la película Titanic). Los *clusters* de máquinas Linux operando con interfaces de red de 100Mbps han permitido obtener una potencia computacional comparable con una supercomputadora cuyo costo ascendería a varios millones de dólares.

Es tiempo de que desde las Universidades se impulse la utilización y el desarrollo de este tipo de software y comenzar a mentalizar a los alumnos que la informática no se reduce a una única plataforma de software y hardware. Es nuestra responsabilidad como docentes hacer notar la gran variedad de productos y evitar que el mundo de la computación se torne en un monopolio tecnológico [1].

Referencias

[1] R. Di Cosmo, *Trampa en el Cyberespacio*, URL:www.dmi.ens.efr/~dicosmo.

[2] Eric. S. Raymond, *La Catedral y el Bazar*, URL:http://labpar.fcfm.buap.mx/

[3] Eric. S. Raymond, Trad. José Soto Pérez, *Software Abierto, una metodología (nueva) de desarrollo?* URL:http://labpar.fcfm.buap.mx/

[4] Sitio Web de GNU: URL:http://www.gnu.org.

[5] Listado de Empresas que utilizan Linux. URL:http://www.m-tech.ab.ca/linux-biz/.

[6] Eric Raymond, *Como transformarse en un hacker* URL:http://labpar.fcfm.buap.mx/

[7] A. S. Tanenbaum, *Operating Systems: Design and Implementation*, Prentice Hall Software Serie , 1988.