Panorama del software libre

Vicente Matellán Olivera Jesús M. González Barahona
Pedro de las Heras Quirós José Centeno González
Francisco Ballesteros Cámara
E-mail: gsyc-profes@gsyc.inf.uc3m.es
Departamento de Informática
Universidad Carlos III de Madrid

9 de febrero de 2005

Resumen

La industria del software utiliza actualmente un modelo de generación de beneficios basado en los ingresos por distribución, es decir, por copia vendida. Como en este modelo de desarrollo los ingresos provienen de la venta de copias, las empresas de producción de software están muy incentivadas para dedicar sus esfuerzos a vender más y mas copias, y por lo tanto, a limitar lo más posible las posibilidades de redistribución de las copias que ellas no controlan (como por ejemplo, los préstamos entre usuarios).

Por otra parte, como sólo la empresa que tiene los derechos de un programa concreto puede mejorarlo y adaptarlo a las necesidades de sus clientes, con lo que se corta de raíz la competencia, que queda limitada a la posibilidad de los usuarios para adquirir un programa u otro. Sin embargo, una vez que el usuario ha elegido un programa, no puede acudir a cualquier profesional o empresa para que lo adapte a sus necesidades o lo mejore.

El objetivo de esta ponencia es presentar los fundamentos en los que se basa un modelo alternativo a éste, denominado **software libre**, así como describir algunos de los proyectos más exitosos que se han desarrollado siguiendo este modelo, como GNU/Linux, Apache, BIND, Perl, Gnome, Samba, etc; y explicar por qué compañías como IBM o Netscape empiezan a ver en él una nueva fuente de ingresos y una ventaja competitiva frente a sus rivales.

1. El concepto de *software* libre

Cuando se habla de "*free software*"[1], hay una peligrosa ambigüedad, debido que "*free*" en inglés significa tanto "libre" como "gratis". Afortunadamente, en

[1] Otro término que se está empleando cada vez con más frecuencia es el de Open Source

castellano no tenemos esta ambigüedad [2], y el significado de "software libre" está mucho más claro.

De todas formas es bueno dejar claro que el software libre no tiene porqué ser gratis. Es más, no suele serlo, o al menos no completamente. Las condiciones que tiene que cumplir un determinado producto software para considerarse libre son:

- Disponibilidad del software para adaptarlo a las necesidades del usuario. Naturalmente, esto incluye hacerle mejoras, corregir erratas, aumentar su funcionalidad, etc.
- Permiso de redistribución del software a otros usuarios, que a su vez podrán seguir redistribuyéndolo y modificándolo [3].

Es también importante dejar claro que se trata de libertad y no de obligación. Esto es, un usuario de un programa libre puede modificarlo si lo estima conveniente, pero desde luego no está obligado a ello. De la misma manera, puede redistribuirlo, pero en general tampoco está obligado a ello. Es más, la mayor parte de sus usuarios actuales no hace ninguna de las dos cosas.

Para poder cumplir las condiciones anteriores, hay una tercera que se deriva necesariamente: "Los usuarios del software en cuestión deben tener acceso a su código fuente".

El código fuente de un programa, habitualmente escrito en un lenguaje de programación de alto nivel (C, C++, Java, ...), es absolutamente necesario para poder entender su funcionamiento, modificarlo y mejorarlo. Si un programador dispone del código fuente de un programa, puede estudiarlo, conocer todos sus detalles, y trabajar sobre él como el autor original.

El software propietario se distingue (desde el punto de vista del usuario no técnico) por no permitir su redistribución y además (desde el punto de vista de los programadores) por negar el acceso al código fuente. Para negar ambas cosas se hace uso de la legislación sobre el derecho de autoría (derecho de *copyright*) para limitar las acciones que el usuario del programa puede hacer con él. Por ejemplo, copiarlo, descompilarlo, modificarlo, etc.

Si el precio de la copia no es el origen de los ingresos, ¿cuál es entonces la la motivación de los programadores para seguir escribiendo nuevos y mejores programas? Claramente el mantenimiento y soporte de usuarios. Es más, los usuarios generalmente no quieren comprar un producto, lo que están queriendo comprar es soporte. Igualmente, los compradores de software no necesitan productos genéricos, sino programas adaptados a sus necesidades particulares. Ambas necesidades constituyen el núcleo de los ingresos de los programadores que producen software libre y la necesidad de realizar esas funciones a gran escala está haciendo que surjan empresas que cubren ese segmento, como Cygnus

[2] De hecho, en la comunidad software de habla inglesa se está extendiendo el término "libre software", usando el vocablo castellano, para evitar la ambigüedad de "free'.

[3] Esta redistribución puede hacerse gratuitamente o mediante pago, independientemente de como se haya obtenido (gratis o pagando).

Solutions (http://www.cygnus.com) compañía fundada en 1989 y que en la actualidad tiene oficinas en USA, Canada, Japón, Reino Unido y Canadá, CRYNWR (http://crynwr.com), Cyclic Software (http://www.cyclic.com) o Ada Core Technologies (http://www.gnat.com) .

2. Qué no es software libre

Aprovechando la publicidad que están consiguiendo los productos de software libre, como por ejemplo GNU/Linux, Perl o Apache, existe el peligro de agrupar bajo la misma denominación ciertos programas producidos bajo otros modelos de desarrollo de software. Sin embargo, sus características están muy alejadas de las del software libre. Entre ellos están el software "a prueba", el *shareware*, el software "regalado" o el distribuido con código fuente pero sin licencia de redistribución. En este apartado se describen brevemente sus características a fin de ubicar correctamente los productos libres que se describen en esta ponencia.

Software a prueba

El modelo de distribución de este tipo de programas está basado en la disponibilidad gratuita del mismo durante cierto tiempo o en ciertas condiciones. Ese tiempo (o esas condiciones) suelen proponerse para que el usuario pueda probar el programa. Si su funcionamiento le convence, puede a continuación comprarlo, de forma similar a como lo haría con cualquier otro software propietario, y adquiriendo en general derechos y obligaciones iguales a las habituales en el software propietario.

La redistribución del programa en cuestión (incluso a veces durante el periodo de prueba) suele estar absolutamente prohibida.

Desde cualquier punto de vista, este software es completamente propietario, con la única diferencia de que hay un tiempo o unas condiciones de prueba. Pero este hecho no cambia su naturaleza, ni permite a este tipo de programas beneficiarse de ninguna de las ventajas que se obtienen con el software libre.

Shareware

Los programas *shareware* habitualmente pueden redistribuirse con libertad. Pero cuando alguien decide utilizarlos más allá de un periodo de prueba (o bajo ciertas condiciones, como por ejemplo uso empresarial) debe pagar una cierta cantidad a su autor. En algunos casos el programa se "protege" para que deje de funcionar, o funcione de forma limitada, si no se ha pagado al autor. En otros, el programa que se distribuye tiene toda la funcionalidad, y la voluntad del autor se protege únicamente por una licencia. Además, normalmente la distribución se hace sólo en versión binaria, sin que el usuario tenga acceso al código fuente.

Por lo tanto, lo que se introduce con esta modalidad es fundamentalmente una forma "diferente" de distribuir y cobrar por el software. La principal diferencia al compararlo con el software a prueba es la libertad de redistribución (que

no de uso). En cualquier caso sigue siendo radicalmente diferente del software libre

Software regalado

Hay veces en que el software propietario se regala. Las motivaciones para esto pueden ser muy variadas, desde promociones para conseguir que un programa se imponga en un determinado mercado hasta intentos de derribar a una empresa competidora. Este software es claramente propietario, y en general las licencias que lo cubren no se diferencian en nada de las "tradicionales". La única diferencia está en las condiciones de adquisición.

Por ejemplo, la guerra establecida entre los navegadores Web llevó a los principales competidores (Microsoft y Netscape) a regalar sus productos (Explorer y Navigator respectivamente), sin que dejasen por ello de ser propietarios, y sin distribuir por supuesto el código fuente.

Software con fuentes

A veces se distribuye el software incluyendo su código fuente, o permitiendo el acceso a él. Pero si no se permite redistribuir el programa y modificar el fuente y redistribuir las modificaciones, no estamos ante software libre.

Por ejemplo, las primeras versiones de Unix, desarrolladas en los Bell Labs, se distribuían como código fuente, con una licencia de ATT que impedía su redistribución libre a terceras partes.

3. Ventajas del modelo de desarrollo basado en software libre

Dicho todo lo anterior, ¿qué ventajas proporciona el software libre frente a los enfoques propietarios? A nuestro entender, se pueden resumir en tres factores: calidad, velocidad de desarrollo y difusión.

3.1. Calidad del Software

En primer lugar, el hecho de disponer del código fuente de un programa permite verificar realmente la calidad del mismo. ¿Alguna vez no se tiene la sensación al usar algún programa propietario de que algo no está bien programado? Cualquiera, con los conocimientos adecuados, puede comprobarlo en el caso de programas desarrollados según la filosofía del software libre. Es más, se pueden realizar sugerencias a los desarrolladores del producto sobre mejoras de diseño, de optimización, etc.

Los usuarios sin conocimientos informáticos profundos probablemente encontraran la ventaja anterior poco práctica, aunque en el fondo les está garantizando que el producto que él/ella usa están bien realizados (igual que nos fiamos de las garantías de calidad de las empresas de normalización ISO-9000).

De la misma forma las erratas, (*bugs*), de los programas pueden detectarse mucho antes e incluso los usuarios no informáticos pueden colaborar en su detección (simplemente usando el programa), involucrándose de esta forma en el desarrollo de los programas que usan.

Existen otro tipo de ventajas relacionadas con la calidad del software, como por ejemplo la garantía de seguridad. Con programas propietarios, en los que no se puede acceder al código del programa, un programa podría estar enviando robando información personal (que programas tiene instalados un usuario, a que lugares se conecta, etc.) o confidenciales (datos económicos, comerciales o de cualquier tipo) del ordenador en el que esté instalado, dañando el sistema (por ejemplo, borrando otros programas), etc. Es decir, a la hora de cuestionarse la seguridad de un programa propietario hay que fiarse de la buena fe del suministrador. Con software libre miles de ojos (muchos de ellos con amplios conocimientos) comprueban que el software sea realmente seguro.

3.2. Velocidad de desarrollo

Con el modelo del software propietario, cuando una empresa decide crear cualquier producto software se ve obligada a partir de cero. No puede aprovechar el código resultado de las experiencias similares anteriores (porque no puede acceder a él). Esto ha limitado y limita gravemente el avance del software.

Con el modelo del software libre se consigue un desarrollo incremental del software, donde cualquier modificación realizada a un programa se realiza sobre modificaciones anteriores. Esto, unido a la cantidad de programadores que pueden llegar a involucrarse en determinados proyectos de software libre, hace que la velocidad de crecimiento de las aplicaciones libres sea mucho más alta que la de las propietarias.

El desarrollo incremental del software es la ventaja competitiva más importante que tiene el software libre. Ninguna empresa, por grande que sea, puede llegar a competir con los grandes grupos de programadores voluntarios (de universidades, cedidos por empresas, profesionales en sus ratos libres, etc.) que se llegan a reunir alrededor de un proyecto común. Programas como GIMP o Apache son buena muestra de ello.

3.3. Difusión de los productos

El software libre, por la facilidad de copia que aporta, se difunde con rapidez, penetrando con fuerza en los segmentos de mercado en los que van existiendo herramientas competitivas. El posible *dumping* [4] de las multinacionales del software no le afecta en absoluto.

Además, el hecho de disponer del código fuente de los programas hace que la traducción de los programas libres de una arquitectura hardware a otra sea mucho más rápida (cualquier programador, en cualquier lugar podría hacerlo).

[4] Es una técnica de marketing (muy habitual últimamente entre las multinacionales del software) regalar productos (véase el caso de los navegadores web) para ganar cuota de mercado o para expulsar a los competidores.

De hecho, el software libre suele ser multi-plataforma, por la gran facilidad de migración, lo que probablemente incline a los fabricantes de plataformas poco extendidas a involucrarse en los proyectos de generación de software libre.

4. Proyectos de *software* libre

Una vez descrita la filosofía del software libre hay que demostrar que esta forma de generar y distribuir el software es viable. La demostración más elocuente son los varios millones de usuarios de GNU/Linux. De hecho se puede considerar que hoy en día GNU/Linux es casi la única alternativa a los productos de Microsoft en el mercado de los sistemas operativos para ordenadores personales.

Pero GNU/Linux, aún siendo el proyecto más conocido, no es ni mucho menos el único proyecto de software libre capaz de competir con los productos del software comercial. Herramientas como el compilador GNAT para Ada constituyen la primera opción para los grandes suministradores del Pentágono; el servidor de HTTP elegido por multinacionales como IBM o Corel es Apache; Internet funciona básicamente porque BIND se encarga de "resolver" direcciones, es decir, de convertir por ejemplo www.elpais.es en 194.224.55.46; el correo electrónico funciona básicamente gracias al programa Sendmail, y así con muchos otros. De entre ellos se comenta a continuación una pequeña muestra:

4.1. Apache

`http://www.apache.org`

Apache es uno de los proyectos de más éxito a nivel empresarial realizado según los principios del software libre. A pesar de la batalla comercial entre Microsoft y Netscape por imponer su servidor web, el más utilizado por los administradores es Apache (como se muestra en `http://www.netcraft.com`).

Su éxito se debe a la calidad y funcionalidad del producto y al convencimiento de que el conocimiento del código fuente es crucial para afrontar los desafíos que presenta un entorno tan cambiante como el del Web. Tanto es así, que hoy otras compañías, como Netscape, han visto que el modelo del software libre es un modelo válido para crear software y han comenzado a distribuir sus productos (en el caso de Netscape su navegador) con licencias basadas en los principios del software libre.

La historia de Apache comenzó en Febrero de 1995. Entonces el servidor Web más popular en Internet era el demonio de HTTP desarrollado por Rob McCool en el *National Center for Supercomputing Applications (NCSA)*, en la Universidad de Illinois, (Urbana-Champaign).

Sin embargo, su desarrollo se había paralizado por lo que un grupo de administradores (utilizando el correo electrónico) se pusieron de acuerdo para coordinar los "parches" que iban realizándose. Utilizando como base NCSA httpd 1.3 añadieron algunos de los parches más habituales, los probaron en sus servidores y realizaron la primera versión oficial de Apache (la 0.6.2) en Abril de 1995. Después de un completo rediseño, la migración a numerosas plataformas

y una serie de pruebas extensivas e intensivas, en Diciembre de 1995 vio la luz la versión 1.0 de Apache. En menos de un año se convirtió en el servidor más utilizado en Internet.

4.2. BIND

`http://www.isc.org/bind.html`

Las siglas BIND (*Berkeley Internet Name Daemon*) no son muy conocidas fuera del ámbito informático más técnico, sin embargo, todo el mundo está muy familiarizado con el servicio que proporciona, el DNS (*Domain Naming System*), es decir, la traducción de las direcciones simbólicas tipo ibm.com, elpais.es, etc. a las nativas de Internet 207.25.98.191. Su utilización es masiva con más de 30 millones de nombres registrados.

Su origen se remonta a 1984 y es mantenido actualmente por la Internet Software Consortium (ISC). Antes de la existencia de este software, el *Stanford Research Institute's (SRI) Network Information Center* (conocido como "el NIC") distribuía la lista de los nombres de las máquinas conectadas a ARPANET en un fichero llamado HOSTS.TXT. El formato del famoso /etc/hosts de las máquinas Unix es una versión reducida de ese fichero. Sin embargo, el mecanismo de actualizar periódicamente ese fichero a todos los ordenadores de la red no escalaba con el crecimiento exponencial [5] de la red. El encargado de resolver el problema fue Paul Mockapetris, que diseñó su arquitectura y construyo la primera versión (JEEVES).

4.3. *GIMP*: GNU Image Manipulation Program

`http://gimp.org/the_gimp.html`

No todo el software libre consiste en extraños programas que utilizan unos cuantos "gurús" informáticos. Muchas de las aplicaciones, y cada vez en mayor medida, son programas de uso general. Como ejemplo de este tipo de aplicaciones se describe a continuación GIMP.

En esencia es un programa de retoque fotográfico, extensible y con una interfaz gráfica que sirve tanto para editar imágenes como para crearlas. Es decir, correspondería a una versión libre de un programa similar al *Photoshop*.

El modelo de desarrollo incremental que proporciona el software libre está haciendo que esta aplicación pueda considerarse un ejemplo paradigmático de su potencia. Continuamente surgen módulos, implementando filtros, analizadores, etc., que pueden enlazarse con los ya existentes. De esta forma es previsible que a muy corto plazo, si no lo es ya, se convierta en la herramienta de edición gráfica más completa.

GNOME: GNU Network Object Model Environment

`http://www.gnome.org`

[5] cada nuevo ordenador no sólo significaba una nueva línea, sino uno más que debía conectarse periódicamente para actualizarse, además de problemas de consistencia, etc.

El proyecto GNOME intenta construir una interfaz gráfica de usuario completa basada en software libre. Como su nombre indica es parte del proyecto GNU (`http://www.gnu.org`) y también forma parte del movimiento Open-Source (`http://www.opensource.org`).

La idea de GNOME es proporcionar todo el conjunto de aplicaciones que utiliza cualquier usuario de ordenadores personales, desde una hoja de cálculo a un procesador de texto, pasando por juegos o herramientas multimedia.

En cuanto a su construcción utiliza GTK (`http://www.gtk.org`) como herramienta para el desarrollo de las interfaces de usuario de todas las aplicaciones.

Como ejemplo de la potencia y usabilidad de GNOME, el gobierno de Méjico lo ha elegido, junto con el sistema operativo GNU/Linux, para equipar las 140.000 aulas con las que piensan introducir a todos sus estudiantes en el mundo de la informática.

4.4. Tcl/Tk

`http://www.wwinfo.com/tcl/`

El desarrollo de Tcl/Tk comenzó en la Universidad de California. Como Perl y Python, Tcl es un lenguaje de tipo *script* (corresponde a las iniciales de *Tool Command Language*), pero lo que realmente le ha convertido en un éxito es el soporte que ofreció para el desarrollo de interfaces gráficas casi desde su origen, gracias a la extensión Tk. Éste era originalmente era un *toolkit* para entornos basados en X-Window, aunque ahora existen interfaces para Windows y Mac.

Actualmente existen entre 500.000 y 1.000.000 de desarrolladores para Tcl/Tk. Se producen 12,000 descargas de Tcl cada semana desde el FTP de Sun Microsystems [6], con una distribución según el sistema operativo de 65 % para Windows, 30 % para las distintas variantes de Unix, y 5 % Macintosh.

4.5. Perl

`http://www.perl.org`

Perl es un lenguaje interpretado, de tipo *script*, multi-plataforma y utilizado fundamentalmente para la manipulación de textos y para la administración de sistemas.

Perl es otro de los productos estrellas de la factoría virtual y distribuida que forman los desarrolladores de software libre. Su creador fue Larry Wall allá por el año 1987, y su invento tiene hoy más de un millón de usuarios, se usa en multitud de empresas, universidades, etc. y existen numerosas empresas que dan soporte y desarrollan productos basados en él.

4.6. Samba

`http://samba.anu.edu.au/samba/`

[6] Sin contar los muchos otros sitios de descarga por todo el mundo.

El conjunto de herramientas Samba es uno de las aplicaciones más populares del software libre en entornos empresariales. Proporciona un conjunto de programas que implementan el protocolo SMB (también conocido como LanManager o Netbios) para los sistemas UNIX. De esta forma permite

El desarrollo de Samba está en manos del Samba Team, que apoya al autor original, Andrew Tridgell en el desarrollo de nuevas versiones y en la corrección de erratas. Dicho equipo trabaja, como es habitual en el software libre, de forma distribuida por todo el planeta, coordinándose a través de las listas de correo y grupos de discusión (grupos *news*) creados a tal efecto.

4.7. PGP

`http://www.nai.com/default_pgp.asp`

PGP (*Pretty Good Privacy*) es el estándar "de facto" en seguridad software. El autor de PGP, Phil Zimmerman, utilizó el modelo del software libre para ayudar a la distribución de su software considerando crucial este mecanismo para evitar el control gubernamental [7] y depurar lo antes posible las erratas que apareciesen.

5. Conclusiones

La base del modelo libre de creación de software del es una concepción diferente del mercado del software. Propone uno con mayor competencia (haciendo que cualquier programa pueda ser adaptado a un usuario por cualquier programador) y a la vez con mayor colaboración entre los programadores (el desarrollo incremental).

Esta forma de entender el mercado existe desde hace años (prácticamente tantos como los ordenadores electrónicos). Durante la mayor parte de este tiempo ha estado confinado en ambientes universitarios, de investigación, grupos reducidos de usuarios de aplicaciones muy concretas, etc. Sin embargo, en los últimos años se ha producido un crecimiento exponencial en el número de usuarios de este tipo de aplicaciones. Ello ha generado a su vez, y como propone el modelo de software libre, una aceleración en el desarrollo de software libre, cada vez más amigable y robusto.

De hecho, ya existe un amplio mercado a explotar: los usuarios de GNU/Linux (venta de ordenadores pre-instalados, soporte *on-line*, desarrollo de drivers, etc.), administraciones públicas como la mejicana (asesoramiento, venta e instalación, mantenimiento, ...), empresas (servidores web, aplicaciones, ...)

Además, el desarrollo de las redes electrónicas de comunicaciones, la mejora en las interfaces de usuario de los programas de software libre y la publicidad que está consiguiendo, permiten aventurar una excelente salud para el software libre.

[7] Lo que le ha costado varias investigaciones en USA por exportar tecnología considerada como clasificada