Software libre y software privativo

22 de diciembre de 2011

Índice

1. Definiciones

Software libre es aquel que garantiza las libertades del usuario. En contraposición, software privativo es aquel donde su dueño se reserva para sí todos los derechos, dando al usuario tan solo una licencia de uso.

Garantizar las libertades del usuario puede tener muchos matices. Pero, en general, se considera libre todo software que permite acceso a su código fuente. No solo para leerlo, sino para modificarlo e incluso para distribuir las modificaciones.

2. Un poco de historia

2.1. Los orígenes del movimiento software libre

Como otros muchos descubrimientos humanos, el concepto de software libre nació fruto de una preocupación personal. A principios de los años 80, Richard Stallman, miembro de la comunidad del laboratorio de inteligencia artificial del MIT, veía cómo cada vez con mas frecuencia los proyectos universitarios derivaban en el nacimiento de empresas privadas para explotarlos. De esta forma, software al que había tenido pleno acceso hasta ese momento pasaba a ser una mercancía comercial restringida.

En 1983 anunció su proyecto GNU (acrónimo recursivo de "GNU is Not Unix") para desarrollar un sistema operativo y un conjunto de herramientas que fueran totalmente libres. Con ello pretendía sentar las bases para facilitar posteriores desarrollos de software libre. Un par de años mas tarde, a finales de 1985, fundó la FSF (Free Software Foundation) y dio a conocer los artículos de la licencia "copyleft" GPL. Básicamente, esta licencia garantiza cuatro libertades fundamentales:

- Libertad para utilizar el software para cualquier propósito.
- Libertad para estudiar el funcionamiento interno del software y para realizar cualquier cambio en él.
- Libertad para distribuir copias libremente.
- Libertad para adaptarlo o mejorarlo y para distribuir esas mejoras, en beneficio de toda la sociedad[1].

2.2. Internet posibilita la cooperación a gran escala

También fue otra inquietud personal la que inició el desarrollo del que posiblemente es el software libre más conocido hoy en día: el sistema operativo Linux. Este sistema operativo toma su nombre a partir del nombre de quien lo comenzó a escribir: Linus Trovalds. Quien, en 1991, como proyecto personal para satisfacer su curiosidad acerca de las capacidades del procesador 80386, comenzó a desarrollar el núcleo de un sistema operativo tipo Unix.

[1]

- Toda mejora o adicción a un software GPL, obligatoriamente ha de ir también bajo esa misma licencia.

Desde etapas tempranas del proyecto, Linus puso su código fuente a disposición de otros; en el grupo de noticias comp.os.inix primero y en su propio grupo comp.os.linux posteriormente. Poco a poco, el sistema fue haciéndose popular entre la comunidad. Atrayendo cada vez a más personas, algunas de las cuales comenzaron a aportar parches e incluso nuevas funcionalidades. Debido al carácter afable y abierto de Linus, todas las contribuciones fueron bienvenidas. Y, de esta manera, comenzó a gestarse una comunidad de desarrolladores trabajando en el proyecto. Una comunidad auto organizada, con Linus al frente como "dictador benevolente" marcando el rumbo.

Por otro lado, el proyecto GNU de Richard Stallman también había avanzado mucho, sobre todo en la parte de utilidades y herramientas de desarrollo de software. (Dicho sea de paso, algunas de estas herramientas se estaban utilizando en el proyecto Linux). Pero el proyecto GNU estaba a falta de la pieza más compleja: un sistema operativo libre sobre el que ejecutar las herramientas. (Hasta ese momento, se ejecutaban sobre diversas variantes de Unix más o menos privativas). La sinergia era clara y ambos proyectos llegaron a un acuerdo tácito, resultando lo que hoy en día se conoce como sistema GNU/Linux.

A partir de ahí, el efecto multiplicador de Internet hizo el resto. A medida que más avanzaban los proyectos y más funcionales eran, más personas comenzaban a interesarse por ellos. Creándose una realimentación y una dinámica realmente viva, bien asentada, que garantizaba la continuidad del proyecto.

2.3. Una gran variedad de licencias y denominaciones

En un principio, la única licencia libre era GPL (*GNU Public License*). Pero el concepto de software libre ha ido evolucionando, refinándose, adaptándose a necesidades particulares,... De tal forma que han ido surgiendo multitud de matices: restricciones de uso para no permitir fines militares, limitaciones a la distribución con fines comerciales, permiso o no para derivar software propietario a partir del libre,... Algunas de estas variantes han llegado a formular nuevos articulados, surgiendo multitud de licencias libres[2]; similares entre sí, pero con diferencias.

En cuanto al propio nombre de software libre (*free software* en inglés), ha tenido también algunos movimientos para cambiarlo por otro menos dado a malentendidos. Por un lado, el hecho de que en inglés la palabra *free* tiene tanto el significado de libre como el de gratuito. Y, por otro lado, el hecho de que habitualmente este tipo de software suele estar disponible para su descarga sin cargo alguno. Ha dado lugar al malentendido de que un software libre ha de ser necesariamente gratuito y al mito de la poca viabilidad económica de cualquier proyecto en torno a software libre. Cuando, en realidad, incluso la propia licencia GPL permite cobrar por el software y existen multitud de negocios rentables en torno al software libre.

Entre varias denominaciones alternativas, una ha sido bastante aceptada, sobre todo en entornos empresariales: software de código abierto (*open source* en inglés). Esta denominación es gestionada por la OSI (Open Software Initiative). Organización que ha definido los criterios[3] mínimos a cumplir por una licencia para ser considerado software de código abierto el software lanzado bajo ella. En general, suelen ser licencias más adaptadas a un uso comercial o que satisfacen una determinada necesidad. Pero garantizando en todo momento las liber-

[2] ver, por ejemplo, http://es.wikipedia.org/wiki/Anexo:Comparación_de_licencias_de_software_libre
[3] http://opensource.org/docs/osd

tades básicas del software: libre acceso al código fuente, libertad para modificarlo y libertad para distribuirlo.

3. ¿Por qué funciona bien el software libre?

En principio, resulta sorprendente cómo alguien dona libremente a la comunidad el fruto de su trabajo o de sus inversiones. ¿Qué posibles motivaciones hay para hacerlo?

3.1. El software es una mercancía diferente

Durante miles de años, decenas de generaciones han tratado y comerciado con bienes tangibles, bienes donde cada unidad física de mercancía tenia su valor y donde si una persona daba a otra su mercancía, la primera había perdido valor. Es tan solo en los últimos 50 años cuando ha surgido un bien intangible de gran valor con un funcionamiento distinto: un bien que gana valor cuando se comparte. De ahí la dificultad de comprender realmente el alcance y las implicaciones del movimiento preconizado por el software libre.

Siguiendo pautas heredadas de siglos de comercio con bienes tangibles o intangibles con un mismo tipo de comportamiento: ser limitados, porque cuesta crear cada unidad, y ser fungibles, porque se gastan con el uso. Parece natural que el software haya de ser también igual. Pero el software es una mercancía con otro tipo de comportamiento: es un bien cuyo valor reside exclusivamente en su uso, no en su posesión (la mayor parte de los programas se desarrollan para uso propio, no para comercializarlos) y su coste reside casi totalmente en la elaboración de la primera unidad, no en la creación de cada unidad funcional de uso (dar una copia a otra persona es sencillo y no implica perder nada de la original).

3.2. ¿Cuándo mantenerlo privativo y cuándo liberarlo?

Hay algunas situaciones donde es ventajoso mantener el software privativo:

- Si ha sido desarrollado expresamente para ser vendido a terceros. Es claro que interesa dar solo licencias de uso individual a cada usuario que lo adquiera.
- Si lleva embebido conocimiento de gran valor competitivo. Puede[4] llegar a ser necesario protegerlo, limitando el acceso directo por parte de los usuarios a su funcionamiento interno.

En el resto de casos, pueden obtenerse importantes ventajas al hacer el software libre:

- Cualquier usuario, bien por si mismo o bien mediante encargo a otros, tiene la capacidad de corregir errores o adaptar el programa a sus necesidades. Resultando esta vía mucho más eficaz y eficiente para mejorar el programa que la tradicional vía de reportar errores o solicitar cambios al desarrollador original y esperar a que este atienda las peticiones.

[4]Pero, en muchas ocasiones, también es posible separar el conocimiento del código fuente. Por ejemplo, mediante una adecuada arquitectura que permita poner dicho conocimiento en archivos de configuración externos al propio software en si.

- Cuanto mayor sea la comunidad de personas usando o aportando al proyecto. Más difundido estará su conocimiento y será menor el peligro de obsolescencia tecnológica por desaparición o por decisiones arbitrarias del desarrollador original.
- El mero hecho de saber que otros podrán revisar libremente el código fuente, suele tener un efecto inconsciente de prestar más atención al desarrollo de este. Obteniéndose software de mejor calidad, ya desde el principio.
- En caso de existir previamente algún programa similar o algún componente que pueda resultar útil al desarrollo, se puede aprovechar. Con el consiguiente ahorro de costes y esfuerzo.

Aun en los casos donde el software ha sido encargado a proveedores externos. En lugar de ver su liberación como un despilfarro al regalar el dinero pagado por él. Es más pragmático ver esta liberación como una fuente de posibles beneficios adicionales. Al fin y al cabo, tanto si se mantiene privativo como si se libera, el software va a tener exactamente el mismo valor de uso y el gasto inicial de desarrollarlo se ha de realizar igualmente. Pero, al mantenerlo privativo, se pierden las posibles ganancias derivadas de hacerlo libre.

3.3. La importancia de las comunidades de desarrolladores

Liberar un software no garantiza de forma automática los beneficios citados en la sección anterior. Las comunidades en torno a los proyectos de software libre tienen unas dinámicas características propias, que se han de respetar si se desea fundar un proyecto exitoso.

La mayor parte de las colaboraciones suelen provenir de personas que no reciben ninguna compensación económica por ello. Su compensación suele ser el reconocimiento de sus pares, el prestigio por el trabajo bien hecho. Este es el verdadero motor que mueve los proyectos de software libre. De ahí la importancia de respetar y hacer respetar en todo momento la lista de créditos. La autoría de cada aportación ha de estar bien clara, tanto para lo bueno como para lo malo.

Al ser prácticamente casi todas las colaboraciones voluntarias, es muy difícil dirigir el proyecto en el sentido tradicional del término. Realmente, casi todos los proyectos de software libre son auto organizados y con amplia participación del colectivo en las decisiones. Lo cual no quita para que persista la necesidad de un reducido número de personas con autoridad suficiente para dirimir las controversias. Aunque también es cierto que estas personas suelen obtener su autoridad a través del reconocimiento de sus méritos en el proyecto por parte del resto de la comunidad.

También por esa voluntariedad, cada cual tiende a colaborar en aquellas parcelas que le sean más interesantes o necesarias. Es inútil intentar imponer una lista de funcionalidades a implementar o unas prioridades en los errores a corregir. Más bien, será el saber hacer de la propia comunidad la que determine qué es lo más urgente o lo más importante en cada momento. Pero, con todo, esto no está reñido con la existencia de algún tipo de hoja de ruta orientativa.

Debido a la naturaleza fragmentada de los esfuerzos, es imperativo un adecuado mecanismo para ir incorporando cambios o nuevo código de forma continuada y frecuente. Un sistema ágil para que toda la comunidad activa esté bien informada y sincronizada con respecto a las últimas aportaciones. La tendencia es a funcionar de forma incremental, en pequeños pasos

útiles hacia el fin perseguido. Manteniendo un repositorio común, centralizado y fácilmente accesible, a donde ir incorporando las aportaciones; usualmente tras un filtro (benévolo) por parte de las personas responsables del proyecto.

4. Conclusiones

El movimiento del software libre surge como respuesta a la creciente privatización que se estaba dando en el mundo del software a partir de los años 80. En un principio sus avances fueron lentos y meramente testimoniales en cuanto al impacto sobre la industria del software y los usuarios. Pero despegó fuertemente en cuanto se combinó con la fuerza de Internet para crear y albergar comunidades de interés en torno a proyectos. Hasta llegar a resultar hoy en día un sistema perfectamente válido como modo de desarrollar y mantener software.

Referencias

[1] Moody, Glyn (2000). *Rebel Code*. Basic Books.

[2] Raymond, Eric S. (2008). *The Cathedral & The Bazaar*. O'Reilly Media.

[3] Torvalds, Linus y Diamond, David (2001). *Just for FUN*. Texere.

[4] http://www.fsf.org/

[5] http://www.gnu.org/licenses/gpl.html

[6] http://en.wikipedia.org/wiki/Comparison_of_free_software_licences

[7] http://www.opensource.org/

[8] http://www.opensource.org/licenses

[9] http://en.wikipedia.org/wiki/List_of_free_software_project_directories

[10] http://en.wikipedia.org/wiki/Software_forge

[11] https://github.com/

[12] http://www.kernel.org/