

ESTUDIO SOBRE LOS PROGRAMAS INFORMÁTICOS DE CÓDIGO ABIERTO PARA EMPRESARIOS Y ABOGADOS

Stephen J. Davidson*, Leonard, Street and Deinard, Minneapolis (Minnesota, Estados Unidos de América)

Copyright 2004 Stephen J. Davidson
Todos los derechos reservados

ÍNDICE

<u>Tema</u>	<u>Página</u>
Un elemento del ecosistema	1
Algunas definiciones útiles	2
Implicaciones económicas y prácticas	5
Desarrollo	5
Tipos de licencias	6
Normas	8
El costo total de propiedad	9
Las licencias <i>GPL</i> y <i>LGPL</i>	11
Conclusión	16

* Stephen Davidson es el Presidente del Departamento de Propiedad Intelectual y Tecnología de la Información del estudio de abogados *Leonard, Street and Deinard* de Minneapolis (Minnesota), donde representa a clientes de Estados Unidos en todo lo relacionado con proyectos de creación, transacciones y litigios en el ámbito de la informática y las tecnologías de la información. Fue Presidente de la *Computer Law Association* y es Profesor adjunto de Derecho de Tecnologías de la Información en la Facultad de Derecho de la Universidad de Minnesota. El Sr. Davidson ha participado en numerosas conferencias sobre Derecho informático en los Estados Unidos de América, Canadá, Europa, Asia y América Latina, y fue redactor de importantes publicaciones periódicas sobre Derecho de Tecnologías de la Información. Su dirección electrónica es: steved@leonard.com.

Un elemento del ecosistema

Los programas informáticos de código[†] abierto son la parte del ecosistema de los programas informáticos que ofrece a los programadores y usuarios de programas informáticos una forma alternativa de desarrollar y distribuir programas informáticos. En ese entorno, cohabitan con un amplio abanico de métodos de desarrollo y distribución, a saber, programas informáticos del dominio público, programas informáticos de libre distribución y compartidos, programas informáticos comerciales y privados, e incluso programas informáticos que aún no están disponibles en el mercado pero de los que ya se habla (*vaporware*). Existen programas informáticos de código abierto en forma de herramientas de desarrollo de programas, sistemas operativos y aplicaciones informáticas.[‡]

El principal promotor de los programas informáticos de código abierto es *Free Software Foundation*, desarrollado a mediados de los años 1980 por el informático Richard Stallman, cuyas misiones principales dice ser las siguientes: 1) promover el derecho de los usuarios a utilizar, estudiar, copiar, modificar y redistribuir programas informáticos; 2) fomentar el desarrollo y el uso de programas y documentación libres; 3) dar a conocer los problemas éticos y políticos que conlleva el libre uso de programas informáticos; 4) desarrollar nuevos programas informáticos libres; y 5) enmarcar dichos programas informáticos en un sistema coherente en el que no se necesite utilizar programas informáticos privados.[§]

El objetivo del presente artículo es dar a conocer al lector las diferencias entre los programas de código abierto y los programas comerciales, señalando las implicaciones prácticas de unos y otros, y planteando algunas cuestiones generales que los programadores, los gobiernos y las empresas comerciales deberían tener en cuenta al elaborar sus políticas y actividades de adquisición de programas, para considerar, más allá de sus costos y beneficios inmediatos, de qué manera las decisiones del hoy podrían afectar, a largo plazo, a las posibilidades y elecciones de las que dispondrán en el mercado tecnológico mundial del mañana.

[†] La palabra “código” se refiere al código fuente de los programas informáticos. Los programas informáticos y sistemas operativos suelen escribirlos personas que utilizan un lenguaje de programación determinado. Es lo que se llama el código fuente del programa. Para que el computador pueda utilizar realmente un programa, lo tiene que traducir del código fuente a un idioma informático que el computador pueda entender y ejecutar. Ese proceso de traducción se llama “compilación”. <http://iet.ucdavis.edu/glossary.cfm#s>. El código fuente es el texto que los programadores introducen y modifican en el computador para producir programas que funcionen. Para cambiar un programa, el programador cambia el texto del código fuente y después genera una nueva versión del programa a partir de él. En general, la modificación o reparación de un programa está limitada por el creador original del programa, algo que no se puede evitar a no ser que se disponga del código fuente pertinente. <http://www.google.com/search?hl=en&lr=&oi=defmore&q=define:source+code>.

[‡] Para consultar una lista de programas de código abierto, véase <http://directory.fsf.org/>.

[§] Véase <http://www.fsf.org/fsf/fsf.html>.

Algunas definiciones útiles

En principio, los programas informáticos comerciales o privados se distribuyen únicamente en formatos binarios ejecutables, y sus autores se reservan el código fuente para modificar y distribuir el programa, o para autorizar a terceros a hacerlo. No pocas veces consideran los programadores de programas informáticos comerciales que los códigos fuente de sus programas son “las joyas de la corona” de la empresa, y velan por lo tanto celosamente por mantenerlos secretos. En el nivel más básico, los “programas informáticos de código abierto” se diferencian de los programas privados por el hecho de que su código fuente está al alcance de todas las personas que obtengan una licencia para usar el programa y, en muchos casos, porque suele autorizarse su modificación y redistribución en formato binario y con código abierto indistintamente.

En los últimos años, las fuerzas del mercado han dado lugar a numerosas variantes de programas informáticos comerciales y de código abierto, con características específicas en cada caso. Dentro de ese abanico, en la actualidad existen: 1) los programas comerciales “de código cerrado” cuyo código fuente sólo es conocido por el programador original; 2) los programas comerciales “de código cerrado” cuyo código fuente está sujeto a licencias que se otorgan a los usuarios autorizados, que podrán usarlo con carácter privado para mantener y modificar el programa en condiciones confidenciales muy estrictas; 3) los programas “de código compartido”, cuyo código fuente se pone a disposición de los titulares de licencias con fines limitados y restricciones relativas a su uso y a su divulgación; 4) los programas informáticos “de código comunitario”, cuyo código fuente está a disposición de una comunidad limitada de usuarios con muchos fines, aunque con ciertas limitaciones en cuanto a su uso, modificación y distribución; y 5) los verdaderos programas “de código abierto”, cuyo código fuente está disponible a los fines de su uso, modificación y distribución “libre”, aunque la licencia correspondiente puede quedar sujeta a ciertas condiciones u obligaciones que obstaculizan su uso comercial (véase más adelante). Las fuerzas del mercado hacen que esos casos se superpongan y las diferencias entre los distintos tipos de programas queden algo difuminadas, aunque todavía siguen generando muchos debates. Gran parte de esos debates han sido provocados por el “movimiento de programas libres”, cuyos miembros defienden los programas informáticos libres con un fervor cuasi-religioso.

A pesar de las diferencias evidentes que presentan con respecto a los de código cerrado, los de código compartido y los de código comunitario, es difícil definir con precisión los programas “de código abierto”. Una búsqueda reciente en *Google*® arrojó no menos de 10 definiciones procedentes de distintas fuentes, aunque todas adoptan perspectivas y enfoques algo diferentes.** Ésta podría ser una definición útil y concisa:

Un programa informático de código abierto es un programa a cuyo código de programación pueden acceder los usuarios con el fin de leerlo, modificarlo, o elaborar nuevas versiones del programa incorporándole

** Véase <http://www.google.com/search?hl=en&lr=&oi=defmore&q=define:Open+source+software>.

ciertos cambios. Existen muchos tipos de programas de código abierto, que se diferencian principalmente por las condiciones de las licencias en virtud de las cuales se pueden (o se deben) redistribuir los ejemplares (modificados) del código fuente.^{††}

Implicaciones económicas y prácticas

Las diferencias prácticas más significativas entre los programas comerciales y los de código abierto se relacionan con la forma de desarrollarlos, el régimen de licencias utilizado para su distribución, su relación con las normas, y el llamado “costo total de propiedad”. En este documento no cabe tratar de forma pormenorizada estas cuestiones que ya han sido objeto de conferencias de varios días en los últimos años, pero pueden resultar de utilidad algunas observaciones generales.

Desarrollo

Los programas informáticos de código abierto suelen ser obra de personas con mucho talento o de grupos informales o “comunidades” de programadores que desean resolver un problema técnico y compartir sus resultados con el resto del mundo. Algunos de sus defensores declaran que los modelos de código abierto generan más innovación, mientras que otros afirman que permiten producir modelos iguales o incluso superiores a los productos comerciales de la competencia. Debido a su forma de distribución, puede existir una gran variedad de “productos” resultantes de un programa de código abierto determinado, e incluso pueden haberse derivado de él muchos otros programas poco conocidos con comportamientos algo o muy diferentes.

Los programas comerciales o privados suelen diseñarse y desarrollarse para responder a la demanda del mercado, o al menos a una necesidad percibida en el mercado. Sus características suelen determinarlas el mercado y los usuarios. Su desarrollo se realiza de forma relativamente estructurada y disciplinada, y los productos resultantes suelen estar bastante bien catalogados, ser de calidad comprobada y contar con buen apoyo. Las características de los programas comerciales suelen ser el resultado de una larga evolución basada en el mercado. Evidentemente, el usuario es quien paga todo esto, lo quiera o no.

Tipos de licencias

Los desarrolladores de programas informáticos comerciales utilizan regímenes de licencias que: 1) permiten explotar su propiedad intelectual de forma que generen suficientes ingresos para pagar la investigación, el desarrollo, la comercialización y los costos complementarios de los programas, y obtener alguna que otra ganancia (en algunos casos los beneficios son escasos, y en otros monumentales); y 2) limitar el uso y las posibilidades de transferencia del programa, prohibir el uso de la ingeniería inversa, y

^{††} Véase www.cross-web.com/information/glossary.htm.

fijar limitaciones de garantía y de responsabilidad. En el caso de los programas informáticos comerciales, los usuarios pagan por los beneficios de los que disfrutan, y posiblemente también por parte de los que no necesitan. No obstante, la teoría económica fundamental en la que se basan las licencias de los programas informáticos comerciales supone que cada parte disfruta por adelantado de una parte de los beneficios que va a obtener, a cambio de un pago que también realizará por adelantado.

Lo mismo sucede con los programas informáticos de código abierto, aunque el modelo de concesión de licencias es muy distinto. Que el código fuente sea abierto no significa necesariamente que cualquiera puede hacer lo que le plazca con el programa informático, aunque en algunos casos así sea. Existen dos modelos principales. Uno es el modelo *BSD* (sigla inglesa de *Berkeley Software Distribution*), muy permisivo puesto que se autoriza su utilización con cualquier fin, e incluso se puede modificar y ser distribuido en forma de producto derivado, gratuito o de pago, siempre y cuando los ejemplares incluyan un tipo determinado de licencia, así como un aviso sobre el derecho de autor, un descargo de garantía y una limitación de responsabilidades. El sistema operativo *UNIX* es un ejemplo de programa informático distribuido a través de una licencia *BSD* (entre otras vías de concesión de licencia), y el sistema operativo *Solaris®* de la empresa *Sun Microsystems* es un programa informático privado derivado de *BSD UNIX*.

El otro modelo principal permite usar, modificar y redistribuir libremente el programa informático, pero es muy restrictivo. Los ejemplos más comunes son la *Licencia pública general (GPL o General Public License)* de la *Free Software Foundation* y la *Licencia pública general reducida (Lesser General Public License o LGPL)*, que permite modificar y distribuir los programas libres derivados pero prohíbe desarrollar productos derivados privados. El sistema operativo *Linux* es un ejemplo de producto de código abierto derivado de *UNIX*, y muchas herramientas de desarrollo de programas conocidas como el *GNU C compiler* también se distribuyen mediante la licencia *GPL*.

Otro tipo de licencia de código abierto es la licencia “de código comunitario”, que puede variar mucho en función de la “comunidad” de usuarios que la use. Por ejemplo, la licencia de una comunidad académica puede ser muy permisiva, mientras que la licencia de la comunidad de *Sun Microsystems* para la plataforma *Java®* presenta importantes restricciones. Las personas interesadas en este tema, pueden obtener más detalles en el sitio Web de *Open Source Initiative*, donde figuran más de 50 ejemplos de distintas licencias de código abierto usadas en la actualidad.^{‡‡}

Normas

Entre las características típicas de los programas informáticos privados o de pago se incluye la posibilidad de que sus propietarios los protejan con derechos de propiedad intelectual y controlen sus especificaciones. Éstas incluyen, por ejemplo, las interfaces de

^{‡‡} <http://www.opensource.org/licenses/bsd-license.php>.

programas de aplicación (*API* por sus siglas en inglés) y otros aspectos relacionados con la interoperabilidad. Por esa razón, los defensores del código abierto dicen a menudo que los códigos abiertos equivalen a las normas abiertas. Se podría afirmar, sin embargo, que las “normas” nacen de dos formas. Por un lado, pueden ser adoptadas de manera generalizada en el mercado por parte de los usuarios. Y por otro lado, pueden elaborarse las organizaciones de normalización.

Lo cierto es que prácticamente todas las “normas abiertas” han sido desarrolladas por consorcios de representantes de la industria privada. Otra realidad es que los programas informáticos de código abierto, por su propia naturaleza, tienden a convertirse en programas sin normas por la relativa facilidad y libertad con que se pueden modificar. De hecho, los distribuidores comerciales de los programas informáticos de código abierto a menudo los modifican deliberadamente para diferenciarse de los demás. Por ejemplo, si bien las empresas *IBM*, *SCO Group*, *Red Hat* y *Sun Microsystems* distribuyen programas derivados del sistema operativo *UNIX*, no todas son iguales ni representan una misma “norma”.

El costo total de propiedad

A menudo se suele establecer una equivalencia entre los conceptos de “código abierto” y “programa informático gratuito”, y muchos defensores de los programas de código abierto señalan que una de las principales ventajas de las soluciones de código abierto es que permiten adquirir programas informáticos sin pagar una tasa en concepto de licencia. No obstante, como señala la *Free Software Foundation*, “el *software* Libre es un asunto de libertad, no de precio. Para entender el concepto, debes pensar en “libre” como en “libertad de expresión”, no como en “cerveza gratis” [N. del T.: en inglés una misma palabra (*free*) significa tanto libre como gratis, lo que ha dado lugar a cierta confusión].”^{§§} Los programas que se suelen considerar como programas informáticos de código abierto son gratuitos, en mayor o menor medida, en el momento de hacerse con ellos – es decir, se pueden adquirir, copiar y usar sin pagar nada a cambio. Sin embargo, cuando uno considera el “precio” del programa en cuestión, es importante tener en cuenta el *costo total* de su adquisición y uso, esto es, el “costo total de propiedad”.

Por ejemplo, imagine el costo de la adaptación de un programa informático “gratuito” a las necesidades de un usuario particular. Las características de los programas comerciales suelen basarse en estudios de mercado y en inversiones para orientar su desarrollo a la comunidad de usuarios o al mercado. Aunque la concordancia no sea siempre perfecta, los programas informáticos comerciales tienden a plantearse como verdaderas soluciones para la mayoría de usuarios. Por lo contrario, los de código abierto a menudo se desarrollan sin prestar mucha atención a los usuarios finales y se distribuyen en un formato que acaba modificándose. Para satisfacer las necesidades de un usuario determinado, puede ser necesario modificarlos, en algunos casos por un precio considerable. Además, los creadores de los programas informáticos comerciales suelen

^{§§} <http://www.GNU.org/philosophy/free-sw.es.html>

encargarse de su mantenimiento y del soporte técnico, mientras que raras veces sucede lo mismo en el caso de los programas informáticos de código abierto.

Al considerar el costo de los programas informáticos libres, es importante tener en cuenta cuestiones tales como el costo de las modificaciones, el mantenimiento y el soporte técnico, así como el costo de la documentación destinada al usuario y la documentación técnica (los programas “libres” a menudo no incluyen ninguna de las dos), el costo de los exámenes de garantía de la calidad (los programas “libres” no van casi nunca acompañados de garantías y en la mayoría de los casos se producen y se distribuyen sin haberse llevado a cabo las pruebas rigurosas a las que se someten los programas comerciales), y los costos de la adaptación, la puesta en funcionamiento, la correcciones de errores, el desarrollo continuo, y las cuestiones de seguridad que se deberá solucionar. El costo de la formación debe tenerse en cuenta doblemente, pues necesitarán formación el personal técnico y los usuarios finales, dado que la mano de obra disponible posiblemente sólo disponga de conocimientos y capacidades adquiridos en relación con el desarrollo de programas comerciales, y quizá sea necesario formarla específicamente para que adquiera la experiencia adicional imprescindible a la hora de trabajar con productos “libres”.

En última instancia, el análisis de la rentabilidad debe basarse en la siguiente pregunta: ¿una solución basada en programas privados obligará al usuario a pagar demasiado por cosas que no necesita, o le ofrece servicios y ventajas que le permitirán obtener mejores resultados que con las ofertas de la competencia disponibles en forma de paquetes “libres” que al final superan el costo total de la solución comercial?

Las licencias *GPL* y *LGPL*

La variante de código abierto que más atención ha llamado – y que más ha llegado a quitarles el sueño a los creadores de programas informáticos comerciales, inversores, y socios de fusiones y adquisiciones de empresas de programas informáticos – es un programa distribuido a través de la *Licencia pública general (GPL)* y la *Licencia pública general reducida (LGPL)* de la *Free Software Foundation*. En su sitio Web, la iniciativa sobre programas informáticos libres de la *Free Software Foundation* (el *Proyecto “GNU”*) explica cómo surgieron la fundación y su proyecto:

El *Proyecto GNU* ha desarrollado un sistema completo de *software* libre llamado *GNU* (*GNU* No es Unix) que es compatible con Unix. El documento inicial de [Richard Stallman](#) sobre el proyecto *GNU* se llama *Manifiesto GNU*, y ha sido traducido a [otros idiomas](#). Se escogió como nombre *GNU* porque cumplía algunos requisitos; primero, era un acrónimo recursivo de “*GNU* No es Unix”; segundo, ya existía esa palabra (N. del T.: en inglés *GNU* significa Ñu), y tercero, porque era divertido decirla (o [cantarla](#)). También tenemos el [Anuncio Inicial del Proyecto GNU](#), escrito en 1983.

La palabra “libre” se refiere a **libertad** no a precio (N. del T.: en inglés se usa la misma palabra para libre y gratuito). Puedes o no pagar un precio por obtener *software GNU*. De cualquier manera, una vez que obtienes el *software*, tienes tres libertades específicas para usarlo. Primera, la libertad de copiar el programa y darlo a tus amigos o compañeros de trabajo; segunda, la libertad de cambiar el programa como desees, por tener acceso completo al código fuente; tercera, la libertad de distribuir una versión mejorada ayudando así a construir la comunidad (si redistribuyes *software GNU*, puedes cobrar una cuota por el acto físico de efectuar la copia, o bien puedes regalarla).

El *Proyecto GNU* fue concebido en 1983 como una forma de devolver el espíritu cooperativo que prevalecía en la comunidad computacional en días pasados – hacer la cooperación posible al eliminar los obstáculos impuestos por los dueños de *software* privativo.

En 1971, cuando Richard Stallman comenzó su carrera en el MIT (Instituto de Tecnología de Massachusetts), trabajó en un grupo que usaba **software libre** exclusivamente. Incluso compañías informáticas frecuentemente distribuían *software* libre. Los programadores eran libres de cooperar unos con otros, y frecuentemente lo hacían.

En los 80, casi todo el *software* era privativo, lo cual significa que tenía dueños que prohibían e impedían la cooperación entre usuarios. Esto hizo necesario el *Proyecto GNU*.***

En el artículo “El sistema operativo *GNU* y el movimiento del *software* libre”, publicado inicialmente en el libro *Open Sources*, Richard Stallman, el fundador de la *Free Software Foundation*, escribió:

Poco después de comenzar en el *Proyecto GNU*, escuché acerca del *Free University Compiler Kit* [Kit de Compilador de la Universidad Libre], también conocido como VUCK. (La palabra alemana para *free* comienza con una V.) Se trataba de un compilador diseñado para manejar múltiples lenguajes, C y Pascal entre ellos, y para admitir múltiples máquinas destino. Le escribí a su autor para consultarle si *GNU* lo podría usar.

Él me respondió burlonamente, dejando en claro que la universidad era libre, pero el compilador no.

La meta de *GNU* era dar libertad a los usuarios, no sólo ser popular. Por lo tanto, debíamos usar términos de distribución que impidieran que el

*** <http://www.GNU.org/GNU/GNU-history.es.html>

software GNU se transformara en *software* privativo. El método que utilizamos se denomina *copyleft*.

El *copyleft* usa la ley de copyright, pero la da vuelta para servir a lo opuesto de su propósito usual: en lugar de ser un medio de privatizar el *software*, se transforma en un medio de mantener libre al *software*.

La idea central del *copyleft* es que le damos a cualquiera el permiso para correr el programa, copiar el programa, modificar el programa y redistribuir versiones modificadas--pero no le damos permiso para agregar restricciones propias. De esta manera, las libertades cruciales que definen al “*software* libre” quedan garantizadas para cualquiera que tenga una copia; se transforman en derechos inalienables.

Para que el *copyleft* sea efectivo, las versiones modificadas deben ser también libres. Esto asegura que todo trabajo basado en el nuestro quedará disponible para nuestra comunidad si se publica. Cuando los programadores que tienen trabajo como programadores se ofrecen como voluntarios para mejorar un *software GNU*, es el *copyleft* lo que impide que sus empleadores digan: no puede compartir esos cambios, porque los queremos usar para hacer nuestra versión propietaria del programa.

El requerimiento de que los cambios deben ser libres es esencial si queremos asegurar la libertad para cada usuario del programa. Las compañías que privatizaron el *X Window System* en general realizaron algunos cambios para transportarlo a sus sistemas y hardware. Estos cambios fueron pequeños comparados con el gran tamaño de *X*, pero no fueron triviales. Si el hacer cambios fuera una excusa para negar libertad a los usuarios, sería fácil para cualquiera tomar ventaja de la excusa.

Un tema relacionado trata la combinación de un programa libre con código no libre. Tal combinación será inevitablemente no-libre; cualesquiera libertades que falten a la parte no-libre, le faltarán también al todo. Si se permiten tales combinaciones se abriría un agujero lo suficientemente grande como para hundir el barco. Por ello, un requerimiento crucial para el *copyleft* es que se tape este hoyo: cualquier cosa agregada a o combinada con un programa bajo *copyleft* debe ser tal que la versión combinada total sea también libre y bajo *copyleft*.

La implementación específica de *copyleft* que usamos para la mayoría del *software GNU* es la *Licencia Pública General* de *GNU* (*GNU General Public License*) o *LPG GNU* para abreviar. Tenemos otras clases de *copyleft* que se usan en circunstancias específicas. Los manuales *GNU* también están bajo *copyleft*, pero utilizamos un *copyleft* mucho más simple, porque no es necesaria la complejidad de la *LPG GNU* para los manuales. †††

Lo que les quita el sueño a muchos es el hecho de que un supuesto programa informático de pago pueda quedar sujeto a una licencia de *GNU* si se considera que “está

††† <http://www.gnu.org/gnu/thegnuproject.es.html> (Se ha eliminado una nota a pie de página en el texto.)

basado” en un programa informático distribuido mediante una licencia de *GNU*, o que “se ha añadido” a uno de esos programas, o “se ha combinado” con él, y también en el caso de que se considere como “una versión modificada” de un programa distribuido mediante una licencia de *GNU*. El problema es que, excepto en casos extremos, resulta difícil determinar con exactitud qué significan esos términos en relación con las licencias de *GNU*; también resulta sumamente fácil que los programadores desleales aprovechen las zonas oscuras existentes en los programas privados de sus empleadores para trasladarlos al régimen del “*copyleft*”. La consecuencia más temible es que, de esa forma, se puedan someter programas supuestamente privados a condiciones u obligaciones que lleven a su distribución gratuita mediante autorizaciones generalizadas en virtud de las cuales se puedan modificar y redistribuir, y que se pueda exigir la divulgación gratuita del código fuente a todos los titulares de una licencia.

Muchos de los programas informáticos distribuidos mediante licencias *GPL* ofrecen funciones necesarias en sistemas más amplios, y los programadores que trabajan en proyectos privados pueden tener la tentación de retomarlos, sencillamente, en lugar de inventar uno desde cero. Además, muchas herramientas útiles y populares de programación y desarrollo se distribuyen con *GPL* o *LGPL*, y algunas de ellas se cuelan en parte en el programa a cuya creación contribuyen. Por otra parte, se ha llegado a decir que tan sólo con crear un programa que pueda funcionar con un componente *GPL* el nuevo programa resultante podría quedar sometido a una licencia *GPL*.^{†††} Si bien en algunos casos resulta bastante claro que el uso del código *GPL* para elaborar un producto nuevo convierte al producto resultante en objeto de una *GPL*, en otros muchos no resulta del todo claro que así sea. La idea de que los programas informáticos privados o de pago pueden ser objeto de estas licencias de código abierto si se incorpora por inadvertencia un pequeño elemento con un código fuente abierto ha llevado a algunos a considerar al código *GPL/LGPL* como un “virus”.

Muchas preguntas sobre las sutiles implicaciones de las licencias de *GNU* y su incidencia en el mundo real del desarrollo y la distribución de programas informáticos quedarán sin respuesta ni aclaración jurídica; por otra parte, en el presente documento no se puede llevar a cabo un estudio pormenorizado de estas cuestiones. Sin embargo, para incitar a la reflexión, ofrecemos a continuación una serie de preguntas abiertas que se han convertido en motivo de desvelo para muchos:

1. Los contratos de *GPL* y *LGPL*, ¿imponen obligaciones positivas a los titulares de licencias, en virtud de las cuales se podría abrir un proceso judicial contra ellos por incumplimiento de contrato? ¿O son simples autorizaciones condicionales, en cuyo caso, si por ejemplo no se divulga el código fuente a los demás licenciarios, se puede acusar a éstos de infracción al derecho de autor?
2. En general, ¿se considera que los empleados de empresas de programas informáticos privados obligan a sus empleadores a respetar las licencias *GPL* y *LGPL* al descargar y usar herramientas de código abierto?

^{†††} Véase <http://kerneltrap.org/node/view/1735>.

3. Aunque las licencias *GPL* o *LGPL* se acepten y sean aplicables en determinados casos, ¿pueden no ser obligatorias algunas de las disposiciones de la licencia debido a su carácter desmedido, si la legislación federal las prohíbe por ser contrarias a las políticas públicas o por otra razón?
4. ¿Tiene la noción de “producto derivado” el mismo sentido jurídico en la legislación tradicional sobre derecho de autor que en *GPL* y *LGPL*, o significa algo diferente en el contexto de una licencia de “copyleft”?
5. ¿Qué significa “combinar” programas informáticos privados con componentes de un programa *GPL* o *LGPL*? ¿Se refiere sólo a su incorporación literal o a su “vinculación estática” a la hora de crear el programa, o puede también referirse a la “vinculación dinámica” cuando el programa se aplica hasta el computador del usuario final? ¿Puede también incluir programas que interactúan con programas *GPL* o *LGPL* si están lo suficientemente sincronizados?
6. ¿Puede “subsanarse” un uso involuntario o no autorizado de un código de *GPL* o *LGPL* en un producto privado eliminando o sustituyendo ese elemento con una nueva componente privada, incluso después de distribuir la versión inicial “infectada”?
7. El hecho de suministrar copias de un programas informático a los socios de una empresa, ¿constituye una “distribución” en el sentido que se contempla en las licencias de *GPL* y *LGPL*?
8. ¿Constituye “distribución” en el sentido que se entiende en las licencias de *GPL* y *LGPL* el hecho de poner a disposición en Internet, de forma pasiva, programas informáticos que se pueden descargar?
9. ¿Constituye “distribución” en el sentido que se entiende en las licencias de *GPL* y *LGPL* el hecho de poner a disposición en Internet una funcionalidad determinada de un programa informático (por ejemplo, en el caso de un proveedor de servicios de aplicación)?
10. ¿Cuáles son las implicaciones de *GPL* y *LGPL* en materia de patentes?
11. En el supuesto de que surjan divergencias con respecto al significado de una disposición de la licencia *GPL* o *LGPL*, ¿qué intención se considera? En otras palabras, ¿se tiene en cuenta la intención de las partes en la transacción concreta a la cual se ha aplicado esa la licencia, o la intención de la persona u organización (o sea, la *Free Software Foundation*) autora del documento en el que figura la licencia?^{§§§}
12. ¿Qué políticas y prácticas deberían aplicar los programadores que crean programas informáticos privados para que sus programas no se vean afectados por *GPL* ni *LGPL*?
13. ¿Qué medidas de diligencia debida convendría que aplique una persona que tiene previsto adquirir o invertir en una empresa cuyo valor se basa en el carácter privado de sus programas informáticos?
14. ¿Cuáles son las obligaciones en materia de auditoría y de divulgación que vinculan a una empresa con participación pública cuyo valor se basa en el carácter privado de sus programas informáticos?

Conclusión

^{§§§} Véanse las respuestas de la *Free Software Foundation* a estas preguntas sobre *GPL* y *LGPL* en <http://www.fsf.org/licenses/gpl-faq.html>.

Los miembros de la industria de programas informáticos independiente han introducido cambios que no se habían visto en nuestro mundo desde la revolución industrial. Jamás hubieran podido invertir tantos miles de millones en el desarrollo de sus tecnologías sin proteger el fruto de su intelecto. Al mismo tiempo, los programas informáticos de código abierto forman parte del ecosistema de los programas informáticos y ofrecen a los programadores y usuarios una forma alternativa de desarrollar y distribuir programas informáticos. Es una necesidad, y a la vez un objetivo.

Para decidir si se adopta una solución basada en programas informáticos de código abierto o si se paga por usar un programa comercial, es importante comparar ambos tipos de programas, y no sólo en lo que se refiere a sus características técnicas, sino que también se debe evaluar la inversión global y el costo total de propiedad que implica. Asimismo, es importante saber si una solución basada en programas de código abierto puede satisfacer las necesidades de los usuarios en lo que se refiere a su compatibilidad operativa con otros programas y a la necesidad de que los usuarios puedan colaborar e interactuar con personas procedentes de sectores y mercados donde los productos comerciales son, de hecho, la norma.

Por último, es importante recordar que el movimiento de los programas de código abierto no sólo desea beneficia a los programadores y a las comunidades de usuarios de los programas a través de la distribución de programas “libres”, sino que también pretende acabar con la necesidad de usar programas privados o de pago. Este movimiento, ayundándose de grupos de presión y de otros medios, ha convencido a algunas empresas e incluso a ciertos gobiernos para que se inclinaran por los productos de código abierto, y las licencias *GPL* y *LGPL* del *Proyecto GNU* han suscitado muchas y muy serias preguntas acerca de la integridad de los programas informáticos de pago o privados desarrollados por empresas privadas. Ojalá el presente documento haya ofrecido datos de antecedentes útiles para los lectores sembrando en ellos la semilla de la reflexión.