GNU
Software Libre

Software Libre

GNU

Utopía Pirata

http://utopia.partidopirata.com.ar

Índice general

1

¿Qué es el software libre?[1]

Definición de software libre

La definición de software libre estipula los criterios que se tienen que cumplir para que un programa sea considerado libre. De vez en cuan-

[1] http://www.gnu.org/philosophy/free-sw.es.html

> do modificamos esta definición para clarificarla o para resolver problemas sobre cuestiones delicadas. Más abajo en esta página, en la sección Historial[2], se puede consultar la lista de modificaciones que afectan la definición de software libre.

"Software libre" es el software que respeta la libertad de los usuarios y la comunidad. En grandes líneas, significa que **los usuarios tienen la libertad para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software**. Es decir, el "software libre" es una cuestión de libertad, no de precio. Para entender el concepto, piense en "libre" como en "libre expresión", no como en "barra libre".

Promovemos estas libertades porque todos merecen tenerlas. Con estas libertades, los usuarios (tanto individualmente como en forma colectiva) controlan el programa y lo que este hace. Cuando los usuarios no controlan

[2] http://www.gnu.org/philosophy/free-sw.es.html#History

el programa, decimos que dicho programa "no es libre", o que es "privativo". Un programa que no es libre controla a los usuarios, y el programador controla el programa, con lo cual el programa resulta ser un instrumento de poder injusto[3].

Un programa es software libre si los usuarios tienen las cuatro libertades esenciales:

- La libertad de ejecutar el programa para cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera (libertad 1). El acceso al código fuente es una condición necesaria para ello.
- La libertad de redistribuir copias para ayudar a su prójimo (libertad 2).
- La libertad de distribuir copias de sus versiones modificadas a terceros (libertad 3). Esto le permite ofrecer a toda

[3]https://gnu.org/philosophy/free-software-even-more-important.html

> la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

Un programa es software libre si otorga a los usuarios todas estas libertades de manera adecuada. De lo contrario no es libre. Existen diversos esquemas de distribución que no son libres, y si bien podemos distinguirlos en base a cuánto les falta para llegar a ser libres, nosotros los consideramos contrarios a la ética a todos por igual.

En el resto de esta página tratamos algunos puntos que aclaran qué es lo que hace que las libertades específicas sean adecuadas o no.

La libertad para distribuir (libertades 2 y 3) significa que usted tiene la libertad para redistribuir copias con o sin modificaciones, ya sea gratuitamente o cobrando una tarifa por la distribución, a cualquiera en cualquier parte[4]. Ser libre de hacer esto significa, entre otras

[4] http://www.gnu.org/philosophy/free-sw.es.html #exportcontrol

cosas, que no tiene que pedir ni pagar ningún permiso para hacerlo.

También debe tener la libertad de hacer modificaciones y usarlas en privado para su propio trabajo o pasatiempo, sin siquiera mencionar que existen. Si publica sus cambios, no debe estar obligado a notificarlo a nadie en particular, ni de ninguna manera en particular.

La libertad de ejecutar el programa significa que cualquier tipo de persona u organización es libre de usarlo en cualquier tipo de sistema de computación, para cualquier tipo de trabajo y finalidad, sin que exista obligación alguna de comunicarlo al programador ni a ninguna otra entidad específica. En esta libertad, lo que importa es el propósito del *usuario*, no el del *programador*. Usted como usuario es libre de ejecutar el programa para alcanzar sus propósitos, y si lo distribuye a otra persona, también esa persona será libre de ejecutarlo para lo que necesite; usted no tiene el derecho de imponerle sus propios objetivos a la otra persona.

La libertad de redistribuir copias debe incluir las formas binarias o ejecutables del programa, así como el código fuente, tanto para las versiones modificadas como para las que no lo estén. (Distribuir programas en forma de ejecutables es necesario para que los sistemas operativos libres se puedan instalar fácilmente). Resulta aceptable si no existe un modo de producir un formato binario o ejecutable para un programa específico, dado que algunos lenguajes no incorporan esa característica, pero debe tener la libertad de redistribuir dichos formatos si encontrara o programara una forma de hacerlo.

Para que las libertades 1 y 3 (realizar cambios y publicar las versiones modificadas) tengan sentido, usted debe tener acceso al código fuente del programa. Por consiguiente, el acceso al código fuente es una condición necesaria para el software libre. El "código fuente" ofuscado no es código fuente real y no cuenta como código fuente.

La libertad 1 incluye la libertad de usar su versión modificada en lugar de la original. Si

el programa se entrega unido a un producto diseñado para ejecutar versiones modificadas por terceros, pero rechaza ejecutar las suyas –práctica conocida como "*tivoización*" o "bloqueo", o (según la terminología perversa de quienes lo practican) "arranque seguro"–, la libertad 1 se convierte en una ficción teórica más que una libertad práctica. Esto no es suficiente. En otras palabras, estos binarios no son software libre, aun cuando se hayan compilado a partir de un código fuente libre.

Una manera importante de modificar el programa es agregándole subrutinas y módulos libres ya disponibles. Si la licencia del programa especifica que no se pueden añadir módulos que ya existen y que están bajo una licencia apropiada, por ejemplo si requiere que usted sea el titular del copyright del código que desea añadir, entonces se trata de una licencia demasiado restrictiva como para considerarla libre.

La libertad 3 incluye la libertad de publicar sus versiones modificadas como software libre. Una licencia libre también puede autori-

zar otras formas de publicación; en otras palabras, no tiene que ser una licencia con copyleft[5]. No obstante, una licencia que requiera que las versiones modificadas no sean libres, no se puede considerar libre.

Para que estas libertades sean reales, deben ser permanentes e irrevocables siempre que usted no cometa ningún error; si el programador del software tiene el poder de revocar la licencia, o de añadir restricciones a las condiciones de uso en forma retroactiva, sin que haya habido ninguna acción de parte del usuario que lo justifique, el software no es libre.

Sin embargo, ciertos tipos de reglas sobre la manera de distribuir software libre son aceptables, cuando no entran en conflicto con las libertades principales. Por ejemplo, el copyleft (definido muy resumidamente) es la regla en base a la cual, cuando redistribuye el programa, no puede agregar restricciones para denegar a los demás las libertades principales. Esta regla no entra en conflicto con las libertades

[5] https://gnu.org/copyleft/copyleft.html

principales, más bien las protege.

En el proyecto GNU usamos el copyleft para proteger legalmente las cuatro libertades para todos. Creemos que existen razones importantes por las que es mejor usar el copyleft[6]. De todos modos, el software libre sin copyleft[7] también es ético. Véase en categorías del software libre[8] una descripción de la relación que existe entre el "software libre", "software con copyleft" y otros tipos de software.

"Software libre" no significa que "no es comercial". Un programa libre debe estar disponible para el uso comercial, la programación comercial y la distribución comercial. La programación comercial de software libre ya no es inusual; el software libre comercial es muy importante. Puede haber pagado dinero para obtener copias de software libre, o puede haber obtenido copias sin costo. Pero sin tener

[6] https://gnu.org/philosophy/pragmatic.html

[7] https://gnu.org/philosophy/categories.html#Non-CopyleftedFreeSoftware

[8] https://gnu.org/philosophy/categories.html

en cuenta cómo obtuvo sus copias, siempre tiene la libertad de copiar y modificar el software, incluso de vender copias[9].

Si una modificación constituye o no una mejora, es un asunto subjetivo. Si su derecho a modificar un programa se limita, básicamente, a modificaciones que alguna otra persona considera una mejora, el programa no es libre.

No obstante, eventuales reglas sobre cómo empaquetar una versión modificada son aceptables si no limitan substancialmente su libertad para publicar versiones modificadas, o su libertad para hacer y usar versiones modificadas en privado. Así, es aceptable que una licencia le obligue a cambiar el nombre de la version modificada, eliminar el logotipo o identificar sus modificaciones como suyas. Son aceptables siempre y cuando esas obligaciones no sean tan agobiantes que le dificulten la publicación de las modificaciones. Como ya está realizando otras modificaciones al programa, no le supondrá un problema hacer algunas más.

[9] https://gnu.org/philosophy/selling.html

Las reglas del tipo “si pone a disposición su versión de este modo, también debe hacerlo de este otro modo” también pueden ser, bajo la misma condición, admisibles. Un ejemplo de una regla admisible sería alguna que requiera que, si usted ha distribuido una versión modificada y uno de los programadores anteriores le solicita una copia, usted deba enviársela (tenga en cuenta que tal regla le sigue permitiendo optar por distribuir o no distribuir su versión). Las reglas que obligan a suministrar el código fuente a los usuarios de las versiones publicadas también son admisibles.

Un problema particular se presenta cuando la licencia requiere que a un programa se le cambie el nombre con el cual será invocado por otros programas. De hecho este requisito dificulta la publicación de la versión modificada para reemplazar al original cuando sea invocado por esos otros programas. Este tipo de requisitos es aceptable únicamente cuando exista un instrumento adecuado para la asignación de alias que permita especificar el nombre del programa original como un alias de la

versión modificada.

En algunos casos las normas de control de exportación y las sanciones comerciales impuestas por el Gobierno pueden limitar la libertad de distribuir copias de los programas a nivel internacional. Los desarrolladores de software no tienen el poder de eliminar o pasar por alto estas restricciones, pero lo que sí pueden y deben hacer es rehusar imponerlas como condiciones para el uso del programa. De este modo, las restricciones no afectarán las actividades ni a las personas fuera de las jurisdicciones de tales Gobiernos. Por tanto, las licencias de software libre no deben requerir la obediencia a ninguna norma de exportación que no sea trivial como condición para ejercer cualquiera de las libertades esenciales.

La mera mención de la existencia de normas de exportación, sin ponerlas como condición de la licencia misma, es aceptable ya que esto no restringe a los usuarios. Si una norma de exportación es de hecho trivial para el software libre, ponerla como condición no constituye un problema real; sin embargo, es un

problema potencial ya que un futuro cambio en la ley de exportación podría hacer que el requisito dejara de ser trivial y que el software dejara de ser libre.

Una licencia libre no puede exigir la conformidad con la licencia de un programa que no es libre. Así, por ejemplo, si una licencia requiere que se cumpla con las licencias de "todos los programas que se usan", en el caso de un usuario que ejecuta programas que no son libres este requisito implicaría cumplir con las licencias de esos programas privativos, lo cual hace que la licencia no sea libre.

Es aceptable que una licencia especifique la jurisdicción de competencia o la sede para la resolución de conflictos, o ambas cosas.

La mayoría de las licencias de software libre están basadas en el copyright, y existen límites en los tipos de requisitos que se pueden imponer a través del copyright. Si una licencia basada en el copyright respeta la libertad en las formas antes mencionadas, es poco probable que surja otro tipo de problema que no ha-

yamos anticipado (a pesar de que esto ocurre ocasionalmente). Sin embargo, algunas licencias de software libre están basadas en contratos, y los contratos pueden imponer un rango mucho más grande de restricciones. Esto significa que existen muchas maneras posibles de que tal licencia sea inaceptablemente restrictiva y que no sea libre.

Nos resulta imposible enumerar todas las formas en las que eso puede suceder. Si una licencia basada en un contrato restringe al usuario de un modo que no se puede hacer con las licencias basadas en el copyright, y que no está mencionado aquí como legítimo, tendremos que analizar el caso, y probablemente concluyamos que no es libre.

Cuando se habla de software libre, es mejor evitar usar términos como “regalar” o “gratuito”, porque dichos términos implican que el asunto es el precio, no la libertad. Algunos términos comunes como “piratería” implican opiniones con las que esperamos no concuerde. Véase un análisis sobre el uso de esos términos en nuestro artículo palabras y frases confusas

que vale la pena evitar[10].

Por último, tenga en cuenta que para interpretar criterios tales como los que se establecen en esta definición de software libre, se hace necesario un cuidadoso análisis. Para decidir si una licencia de software específica es una licencia de software libre, la evaluamos en base a estos criterios para determinar si concuerda tanto con el espíritu de los mismos como con la terminología precisa. Si una licencia incluye restricciones inaceptables, la rechazamos, aun cuando no hubiéramos anticipado el problema en estos criterios. A veces los requisitos de una licencia revelan una cuestión que hace necesaria una reflexión más profunda, incluyendo la discusión con un abogado, antes de que podamos decidir si el requisito es aceptable. Cuando llegamos a una conclusión sobre una nueva cuestión, solemos actualizar estos criterios para que resulte más fácil ver por qué una cierta licencia puede o no ser calificada como libre.

Si está interesado en saber si una licencia

[10] https://gnu.org/philosophy/words-to-avoid.html

específica está calificada como licencia de software libre, consulte nuestra lista de licencias[11]. Si la licencia que busca no está en la lista, puede consultarnos enviándonos un correo electrónico a <licensing@gnu.org>.

Si está considerando escribir una nueva licencia, por favor contacte a la FSF escribiendo a esa dirección. La proliferación de distintas licencias de software libre significa mayor esfuerzo por parte de los usuarios para entenderlas; podemos ayudarle a encontrar una licencia de software libre que ya exista y que satisfaga sus necesidades.

Si eso no fuera posible, si realmente necesita una nueva licencia, con nuestra ayuda puede asegurarse de que la licencia sea realmente una licencia de software libre y evitar varios problemas en la práctica.

[11] https://gnu.org/licenses/license-list.es.html

Más allá del software

Los manuales de software deben ser libres[12] por las mismas razones que el software debe ser libre, y porque de hecho los manuales son parte del software.

También tiene sentido aplicar los mismos argumentos a otros tipos de obras de uso práctico; es decir, obras que incorporen conocimiento útil, tal como publicaciones educativas y de referencia. La Wikipedia[13] es el ejemplo más conocido.

Cualquier tipo de obra *puede* ser libre, y la definición de software libre se ha extendido a una definición de obras culturales libres[14] aplicable a cualquier tipo de publicación.

[12] https://gnu.org/philosophy/free-doc.html
[13] http://wikipedia.org
[14] http://freedomdefined.org

¿Código abierto?

Otro grupo ha comenzado a usar el término "código abierto" (del inglés "open source") que significa algo parecido (pero no idéntico) a "software libre". Preferimos el término "software libre" porque una vez que ya se sabe que se refiere a la libertad y no al precio, evoca la idea de libertad. La palabra "abierto" nunca se refiere a la libertad[15].

Traducción: Luis Miguel Arteaga Mejía, 2001.

[15] https://gnu.org/philosophy/open-source-misses-the-point.html

[16] http://creativecommons.org/licenses/by-nd/3.0/us/deed.es

Revisiones: Hernán Giovagnoli, Daniel (lluvia).

2

¿Qué es el copyleft?[1]

El copyleft es un método general para hacer un programa (u otro tipo de trabajo) libre, exigiendo que todas las versiones modificadas y extendidas del mismo sean también libres.

La forma más simple de hacer que un programa sea libre es ponerlo bajo dominio público[2], sin derechos de autor. Esto permite a la

[1] http://www.gnu.org/copyleft/copyleft.es.html

[2] https://gnu.org/philosophy/categories.es.html#PublicDomainSoftware

gente compartir el programa y sus mejoras si así lo desean. Pero también permite que gente no tan cooperativa convierta el programa en software privativo[3]. Pueden realizarse tantos cambios como se quiera y distribuir el resultado como un producto privativo. Las personas que reciben el programa con esas modificaciones no tienen la libertad que el autor original les dio, ya que han sido eliminadas por el intermediario.

El objetivo del Proyecto GNU[4] es dar a *todos* los usuarios la libertad de redistribuir y cambiar software GNU. Si los intermediarios pudiesen quitar la libertad, tendríamos muchos usuarios, pero no tendrían las anteriores libertades. Por eso, en lugar de poner el software GNU bajo dominio público, lo protegemos con "Copyleft". Con copyleft cualquiera que redistribuya el software, con o sin cambios, deberá de otorgar al usuario la libertad de copiarlo y modificarlo, garantizando que se man-

[3] https://gnu.org/philosophy/categories.es.html#ProprietarySoftware

[4] https://gnu.org/gnu/thegnuproject.es.html

tendrán estas libertades para todos los usuarios.

El copyleft también provee un incentivo[5] para que otros programadores se sumen al software libre. Algunos programas libres importantes, como el compilador GNU para C++, existen sólo por este motivo.

El copyleft también ayuda a los programadores que quieran contribuir con mejoras[6] al software libre[7] obteniendo permiso para hacerlo. Estos programadores a menudo trabajan para compañías o universidades que harían casi cualquier cosa para conseguir más dinero. Un programador puede querer contribuir con sus cambios a la comunidad, pero su superior puede querer convertir sus cambios en un producto software privativo.

Cuando nosotros le explicamos a sus superiores que es ilegal el distribuir la versión mejorada a menos que sea software libre, nor-

[5] https://gnu.org/philosophy/pragmatic.es.html
[6] https://gnu.org/prep/tasks.html
[7] https://gnu.org/philosophy/free-sw.es.html

malmente deciden distribuirlo como software libre en lugar de desecharlo.

Para cubrir un programa con "copyleft" se debe, en primer lugar, declarar que sus derechos están reservados (tiene copyright). Después deben añadirse unos términos de distribución, los cuales son un instrumento legal que dotará a todo el mundo de los derechos de utilizar, modificar, y redistribuir el código del programa *o de cualquier programa derivado del mismo*, pero sólo si los términos de distribución no son alterados. Así, el código y las libertades se hacen legalmente inseparables.

Los desarrolladores de software privativo usan el copyright para eliminar la libertad de los usuarios; nosotros usamos los derechos de autor para garantizar esa libertad. Es por eso que invertimos el nombre, convirtiendo los derechos de autor (copyright) en copyleft.[8]

[8]Nota del traductor: El nombre es un juego de palabras en inglés para indicar que el copyleft es, de alguna manera, distinto al copyright. El término "left" de "copyleft" significa "izquierda", que es la dirección opuesta a la derecha, "right" (de "copyright") en in-

El copyleft es una forma de usar los derechos de autor en un programa. No implica abandonar los derechos de autor, ya que, si se abandonasen, el uso del copyleft sería imposible.

El copyleft es un concepto general y, por lo tanto, no puede usarse de forma directa; solamente es posible utilizar una implementación especifica del concepto. En el Proyecto GNU los términos específicos de distribución que usamos para nuestro software están contenidas en la Licencia Pública GNU (disponible en formato HTML,[9] texto plano[10] y Texinfo[11]). La Licencia Pública General GNU se llama a menudo GPL de GNU para acortar. También existe una página de preguntas frecuentes[12] sobre la GPL de GNU. También puede leer sobre por qué la FSF obtiene las asignaciones de copyright de sus colaborado-

glés.

[9] https://gnu.org/copyleft/gpl.html

[10] https://gnu.org/copyleft/gpl.txt

[11] https://gnu.org/copyleft/gpl.texi

[12] https://gnu.org/licenses/gpl-faq.es.html

res[13].

Una forma alternativa de copyleft es la Licencia Pública General Affero de GNU (AGPL) (disponible en formato HTML[14] texto[15] y Texinfo[16]). Esta licencia está diseñada para programas que pueden ser utilizados en servidores, y asegura que las versiones modificadas que se utilizan para implementar servicios para los usuarios se publiquen como código fuente disponible al público.

Para algunas (no todas) librerías de GNU, puede resultar aceptable la forma de copyleft que se estipula en la Licencia Pública General Reducida de GNU (LGPL de GNU) (disponible en formato HTML[17] , texto[18] y Texinfo[19]. Para más detalles sobre el uso de la LGPL, consulte el artículo Por qué en su próxima bi-

[13]https://gnu.org/licenses/why-assign.es.html
[14]https://gnu.org/licenses/agpl.html
[15]https://gnu.org/licenses/agpl.txt
[16]https://gnu.org/licenses/agpl.texi
[17]https://gnu.org/licenses/lgpl.html
[18]https://gnu.org/licenses/lgpl.txt
[19]https://gnu.org/licenses/lgpl.texi

blioteca no debería utilizar la Lesser GPL[20].

La Licencia de Documentación Libre de GNU (FDL) (disponible en formato HTML[21], texto plano[22] y Texinfo[23]) es una forma de copyleft diseñada para usarse en manuales, libros de texto u otros documentos para asegurar a todo el mundo la libertad de copiar y redistribuir el trabajo, con o sin modificaciones y de forma comercial o no comercial.

La licencia apropiada se encuentra incluida en muchos manuales y en cada distribución del código fuente de GNU.

Todas estas licencias están diseñadas de manera que usted pueda aplicarlas fácilmente a sus propios trabajos, asumiendo siempre que sea el titular de losderechos de autor. No es necesario modificar la licencia para hacerlo, simplemente hay que incluir una copia de la licencia en el trabajo y añadir notas en los fi-

[20] https://gnu.org/philosophy/why-not-lgpl.html
[21] https://gnu.org/licenses/fdl.html
[22] https://gnu.org/licenses/fdl.txt
[23] https://gnu.org/copyleft/fdl.texi

cheros del código fuente que hagan referencia adecuadamente a la licencia.

Usando los mismos términos de distribución para diferentes programas hace más sencillo el poder copiar código entre estos. Cuando todos tienen los mismos términos de distribución no hay problema ninguno. La segunda versión de la Licencia Pública Reducida de GNU (LGPL de GNU) incluye una clausula que permite cambiar estos términos a los de la licencia GPL ordinaria, de esta manera puede copiar código a otro programa cubierto por la GPL Versión 3. La LGPL de GNU Versión 3 se crea añadiendo una clausula excepcional, creando así una compatibilidad automática.

Si quiere cubrir su programa con la Licencia Pública General de GNU o la Licencia Pública General Reducida de GNU, por favor visite la página de instrucciones para licenciar su software[24] para asesorarse. Por favor, observe que debe usar el texto completo de la licencia escogida. No se permite las copias parciales de

[24] https://gnu.org/copyleft/gpl-howto.html

las licencias.

Si quiere cubrir su manual con la licencia FDL de GNU (también conocida como GFDL), por favor siga las instrucciones que encontrará al final[25] del texto de la licencia GFDL, y la página de instrucciones de la GFDL[26]. Al igual que antes, no se permiten las copias parciales de la licencia.

Desde un punto de vista legal, es un error utilizar la "C" invertida dentro de un círculo en lugar del símbolo del copyrght. El copyleft está basado en la ley del copyright, de manera que la obra tiene que llevar una nota legal de copyright, que debe ir acompañada del símbolo del copyright (la letra "C" dentro de un círculo) o de la palabra "copyright".

La letra "C" invertida no tiene ningún alcance legal, por lo tanto no sirve como nota de copyright. Puede resultar gracioso usarla por ejemplo en la tapa de un libro o un póster, pero ¡tenga cuidado cuando la incluya en una

[25] https://gnu.org/copyleft/fdl.html#addendum
[26] https://gnu.org/copyleft/fdl-howto.html

página web![27]

---

[27] https://en.wikipedia.org/wiki/Copyleft#Symbol

[28] http://creativecommons.org/licenses/by-nd/3.0/us/deed.es

3

El manifiesto de GNU[1]

El manifiesto de GNU, que aparece a continuación, lo escribió Richard Stallman[2] en los inicios del proyecto GNU para solicitar participación y apoyo. Durante esos primeros años se hicieron algunas pequeñas actualizaciones para reflejar la evolución del proyecto, pero ahora creemos que es mejor no modificarlo y dejarlo tal cual lo han leído la mayoría de las

[1] http://www.gnu.org/gnu/manifesto.es.html
[2] http://www.stallman.org/

personas.

Con el transcurso del tiempo hemos aprendido que ciertos malentendidos comunes podían evitarse con una redacción diferente. Las aclaraciones al pie de página que hemos añadido a partir de 1993 ayudan a clarificar estos puntos.

Para obtener información actualizada sobre el software de GNU disponible, consulte nuestro servidor web[3], en particular la lista de software[4]. Para informarse de cómo colaborar, consulte la página http://www.gnu.org/help/help.html.

¿Qué es GNU? ¡GNU No es Unix!

GNU, que significa "Gnu No es Unix", es el nombre del sistema de software completamen-

[3] http://www.gnu.org/home.html
[4] http://www.gnu.org/software/software.html

te compatible con Unix que estoy escribiendo para entregarlo libremente a todas las personas que puedan utilizarlo[5]. Algunos voluntarios me están ayudando. Las aportaciones de tiempo, dinero, programas y equipos son muy necesarias.

Hasta el momento tenemos un editor de texto, Emacs con Lisp, para escribir coman-

[5]Esta expresión resultó poco precisa. La intención era decir que nadie tendría que pagar por el **permiso** para usar el sistema GNU. Pero la expresión no es del todo clara, y a menudo se interpreta que las copias de GNU deberían distribuirse siempre a un costo bajo o sin costo. Esta nunca fue la intención. Más adelante, el manifiesto menciona la posibilidad de que las empresas provean servicios de distribución para obtener ganancias. A partir de entonces, aprendí a distinguir cuidadosamente entre "free" (libre) en el sentido de libertad y "free" (gratis) referido al precio, ya que en inglés, el término "free" puede referirse tanto a la libertad como al precio. El software libre es aquel que ofrece a los usuarios la libertad de distribuirlo y modificarlo. Algunos pueden obtener copias sin pagar, mientras que otros pagan para obtenerlas, y si los fondos ayudan a apoyar la mejora del software, tanto mejor. Lo importante es que todos los que posean una copia tengan la libertad de colaborar con los demás al usar el programa.

dos de edición, un depurador de código fuente, un generador parser compatible con yacc, un enlazador y alrededor de treinta y cinco utilidades. Una shell (intérprete de comandos) que está casi terminada. Un nuevo compilador portable y optimizador de C se autocompiló y posiblemente lo publicaremos este año. Existe un núcleo inicial, pero se necesitan muchas más características para emular a Unix. Cuando el núcleo y el compilador estén completos, será posible distribuir un sistema GNU apropiado para el desarrollo de programas. Usaremos el formateador de documentos TeX, pero también estamos trabajando en una versión de nroff. Usaremos también el sistema libre y portable de ventanas X. Después de esto agregaremos un Common Lisp portable, un juego Imperio, una hoja de cálculo y cientos de otras cosas, además de la documentación en línea. Esperamos proporcionar, con el tiempo, todas las utilidades que vienen normalmente con un sistema Unix y más.

GNU podrá ejecutar programas de Unix, pero no será idéntico a Unix. Haremos todas

las mejoras que sean convenientes, en base a nuestra experiencia con otros sistemas operativos. Concretamente, planeamos tener nombres de archivos más largos, números para las versiones de los archivos, un sistema de archivo a prueba de caídas y tal vez incorporemos un sistema para completar los nombres de archivos, un soporte de visualización independiente del terminal y, quizá en el futuro, un sistema de ventanas basado en Lisp a través del cual varios programas Lisp y programas comunes de Unix puedan compartir una pantalla. Tanto C como Lisp estarán disponibles como lenguajes de programación del sistema. Intentaremos también dar soporte a UUCP, MIT Chaosnet y protocolos de Internet para las comunicaciones.

GNU está orientado inicialmente a las máquinas de la clase 68000/16000 con memoria virtual, porque son las máquinas donde es más sencilla su ejecución. El esfuerzo adicional para hacerlo funcionar en máquinas más pequeñas se lo dejaremos a quienes quieran utilizarlo en ellas.

Para evitar una horrible confusión, por favor pronuncie la *g* en la palabra "GNU" cuando se refiera al nombre de este proyecto[6].

Por qué debo escribir GNU

Considero que la Regla de Oro me exige que si me gusta un programa lo debo compartir con otras personas a quienes también les guste. Los vendedores de software quieren dividir a los usuarios y dominarlos para llevarlos a aceptar no compartir su software con los demás. Me rehúso a romper la solidaridad con otros usuarios de esta manera. Mi conciencia me impide firmar un acuerdo de confidencialidad o un acuerdo de licencia de software. Durante años trabajé en el Laboratorio de Inteligencia Artificial oponiéndome a estas tendencias y otras descortesías, pero al final fueron demasiado lejos: no podía permanecer en una

[6] GNU se pronuncia en inglés de forma muy similar a "new", que significa "nuevo".

institución donde tales cosas se hicieran en mi nombre en contra de mi voluntad.

Para poder continuar a utilizar las computadoras sin deshonra, he decidido agrupar un conjunto suficiente de software libre para poder vivir sin usar ningún software que no sea libre. He renunciado al Laboratorio de Inteligencia Artificial para evitar que el MIT pueda usar alguna excusa legal que me impida distribuir software de GNU[7].

[7]La expresión "regalar" es otro indicio de que yo todavía no había separado claramente la cuestión del precio de la cuestión de la libertad. Ahora recomendamos no usar esta expresión al hablar acerca del software libre. Para una explicación más detallada, consulte el artículo "Palabras y frases confusas que vale la pena evitar" (en http://www.gnu.org/philosophy/words-to-avoid.html#GiveAwaySoftware))

Por qué GNU será compatible con Unix

Unix no es mi sistema ideal, pero no es tan malo. Las características esenciales de Unix parecen ser buenas, y pienso que puedo añadir lo que le falta sin echarlas a perder. Y un sistema compatible con Unix facilitará su adopción por parte de muchas otras personas.

Cómo estará disponible GNU

GNU no está en el dominio público. Todos tendrán permiso para modificar y redistribuir GNU, pero a ningún distribuidor se le permitirá restringir su redistribución posterior. Es decir, no se autorizarán modificaciones privativas[8]. Quiero asegurarme que todas las versiones de GNU permanezcan libres.

[8] https://gnu.org/philosophy/categories.html#ProprietarySoftware

Por qué muchos programadores quieren colaborar

He encontrado muchos programadores que están entusiasmados con GNU y quieren colaborar.

Muchos programadores están descontentos con la comercialización del software de sistema. Puede permitirles ganar más dinero, pero los hace sentirse en conflicto con otros programadores en lugar de sentirse como compañeros. El fundamento de la amistad entre programadores es el compartir programas, pero los acuerdos de mercadotecnia que los programadores suelen utilizar básicamente prohíben tratar a los demás como amigos. El comprador de software debe escoger entre la amistad y la obediencia a la ley. Naturalmente, muchos deciden que la amistad es más importante. Pero aquellos que creen en la ley a menudo no se sienten a gusto con ninguna de las opciones. Se vuelven cínicos y piensan que la programación es sólo una manera de ganar dinero.

Al desarrollar y utilizar GNU en lugar de programas privativos, podemos ser hospitalarios con todos y obedecer la ley. Además, GNU sirve como ejemplo de inspiración y como bandera para animar a otros a unirse a nosotros en el compartir. Esto puede darnos una sensación de armonía que es imposible obtener cuando utilizamos software que no es libre. Porque para cerca de la mitad de los programadores con quienes hablo, esto es un motivo de felicidad importante que el dinero no puede reemplazar.

Cómo colaborar

> Hoy en día, para conocer las tareas en las que puede colaborar en el ámbito del software, consulte la lista de proyectos prioritarios[9] y la lista se busca ayuda,[10] la lista general de tareas para paquetes software de GNU. Para colaborar de

[9] http://fsf.org/campaigns/priority-projects
[10] http://savannah.gnu.org/people/?type_id=1

otras formas, consulte la guía para colaborar con el Proyecto GNU[11].

Pido a los fabricantes de ordenadores que donen máquinas y dinero. A los individuos les pido donaciones en forma de programas y trabajo.

Una de las consecuencias que puede esperar si dona máquinas es que GNU se ejecutará en ellas con anticipación. Las máquinas deben estar completas, listas para utilizar sistemas, aprobadas para su uso en una zona residencial y no requerir ventilación o fuentes de energía sofisticadas.

He encontrado muchos programadores ansiosos de contribuir trabajando a tiempo parcial para GNU. Para la mayoría de los proyectos, tal trabajo distribuido a tiempo parcial sería muy difícil de coordinar, las partes escritas de forma independiente no funcionarían correctamente unidas. Pero para la tarea particular de reemplazar Unix, este problema no existe. Un sistema completo Unix contie-

[11] http://www.gnu.org/help/help.html

ne cientos de programas de utilidades, cada uno de los cuales se documenta por separado. La mayoría de las especificaciones de interfaz se fijan por compatibilidad con Unix. Si cada colaborador puede escribir un reemplazo compatible para una sola utilidad Unix, y hacer que funcione correctamente en lugar del original en un sistema Unix, entonces estas utilidades funcionarán correctamente cuando se ensamblen. Aun teniendo en cuenta las leyes de Murphy acerca de algunos problemas inesperados, el montaje de estos componentes será una tarea factible (el núcleo requirá una comunicación más estrecha y deberá trabajarse en un grupo pequeño y compacto).

Si obtengo más donaciones, podría contratar a algunas personas a tiempo completo o parcial. El sueldo no será alto para los estándares de los programadores, pero estoy buscando a gente para quien la construcción de un espíritu comunitarios sea tan importante como ganar dinero. Lo veo como una forma de permitir que estas personas se dediquen con todas sus energías a trabajar en GNU ahorrándoles la

necesidad de ganarse la vida de otra manera.

Por qué se beneficiarán todos los usuarios de computadoras

Una vez que GNU esté terminado, todo el mundo podrá obtener un buen sistema de software tan libre como el aire[12].

Esto significa mucho más que ahorrarse el dinero para pagar una licencia Unix. Significa evitar el derroche inútil de la duplicación de esfuerzos en la programación de sistemas. Este esfuerzo se puede inventir en cambio en el avance de la tecnología.

[12] Aquí también omití distinguir cuidadosamente entre los dos diferentes significados de "free" (que en inglés puede significar "gratis" o "libre", N. d. T.). La afirmación tal como está escrita no es falsa, se pueden obtener copias gratuitas de software de GNU –de los amigos o a través de Internet–, pero sugiere una idea errónea.

El código fuente del sistema completo estará disponible para todos. Como resultado, un usuario que necesite cambios en el sistema siempre será libre de hacerlo él mismo, o contratar a cualquier programador o empresa disponible para que los haga. Los usuarios ya no estarán a merced de un programador o empresa propietaria de las fuentes y que sea la única que puede realizar modificaciones.

Las escuelas podrán ofrecer un entorno mucho más educativo, y alentar a todos los alumnos a estudiar y mejorar el código. El laboratorio de computación de Harvard solía tener la política de que ningún programa podía ser instalado en el sistema si no se publicaba previamente su código fuente, llegando al punto de negarse a instalar ciertos programas. Yo me inspiré mucho en esa política.

Por último, el lastre de considerar quién es dueño de qué sistema de software y de lo que está o no está permitido hacer con él, habrá desaparecido.

Los acuerdos que obligan a la gente a pagar

por usar un programa, incluyendo el licenciamiento de las copias, siempre incurren en un costo enorme para la sociedad a través de los mecanismos engorrosos necesarios para calcular la cantidad (es decir, qué programas) una persona debe pagar. Y sólo un estado policial puede forzar a todos a obedecer. Considere la posibilidad de una estación espacial en donde el aire debe fabricarse con un gran costo: cobrar a cada persona por litro de aire puede ser justo, pero usar una máscara para medir el aire durante todo el día y toda la noche es insoportable, incluso si todo el mundo puede permitirse el lujo de pagar la factura del aire. Y las cámaras de video en todas partes para ver si alguna vez alguien se quita la máscara son indignantes. Es mejor apoyar a la planta de aire con un impuesto y desechar las máscaras.

Copiar todo o parte de un programa es tan natural para un programador como respirar, además es productivo. Debería ser libre.

Algunas objeciones, fácilmente rebatibles, a los objetivos de GNU

"Nadie lo va a usar si es libre, porque eso significa que no cuenta con ningún tipo de asistencia".

"Hay que cobrar por el programa para pagar por el servicio de asistencia".

Si la gente prefiere pagar por GNU más el servicio en lugar de recibir GNU sin servicio, una empresa que preste solamente el servicio a las personas que hayan obtenido GNU debe ser rentable[13].

Debemos distinguir entre el soporte en forma de trabajo de programación real y lo que es simplemente guiar al usuario. El primero es algo que uno no puede confiar a un proveedor de software. Si su problema no es compartido por bastante gente, el vendedor no se preocu-

[13] Ya existen varias compañías de este tipo.

pará en solucionarlo.

Si su empresa necesita poder contar con soporte, la única manera es tener todo el código fuente y las herramientas necesarias. Entonces puede contratar a cualquier persona disponible para corregir el problema, y no estará a merced de ningún individuo. Con Unix, el precio del código fuente deja fuera de consideración a la mayoría de las empresas. Con GNU esto será sencillo. Puede ser que no haya ninguna persona competente disponible, pero por este problema no se puede culpar a los acuerdos de distribución. GNU no elimina todos los problemas del mundo, sólo algunos de ellos.

Mientras tanto, los usuarios que no saben nada acerca de las computadoras necesitan que los guíen: hacer cosas que podrían hacer por sí mismos fácilmente, pero no saben cómo.

Estos servicios podrán ser prestados por empresas que vendan solamente el servicio de asesoría y reparación. Si bien es cierto que los usuarios prefieren gastar dinero y obtener un

producto con el servicio, también estarán dispuestos a adquirir el servicio habiendo obtenido el producto en forma gratuita. Las empresas de servicios competirán en calidad y precio, los usuarios no estarán atados a ninguna en particular. Mientras tanto, aquellos de nosotros que no necesitamos el servicio deberíamos tener la posibilidad de utilizar el programa sin tener que pagar por el servicio.

"No se puede llegar a muchas personas sin publicidad, y para financiarla es necesario cobrar por el programa".

"No tiene sentido publicitar un programa que la gente puede obtener gratuitamente".

Hay diversas formas de publicidad gratuita o muy barata que se puede utilizar para informar a los usuarios de computadoras acerca de algo como GNU. Pero quizás sea cierto que uno puede llegar a más usuarios de microcomputadoras con publicidad. Si esto es realmente así, un negocio que publicite el servicio pago de copiado y envío por correo del soft-

ware de GNU debería ser lo suficientemente exitoso como para pagar por su publicidad y mucho más. De esta manera, solo los usuarios que se benefician de esta publicidad la pagarán.

Por otro lado, si mucha gente consigue GNU de sus amigos, y esas empresas no tienen éxito, esto demostrará que la publicidad no era realmente necesaria para difundir GNU. ¿Por qué es que los defensores del libre mercado no quieren dejar que el libre mercado lo decida?[14]

"Mi compañía necesita un sistema operativo privativo para tener una ventaja competitiva".

GNU quitará el software de sistema operativo del entorno de la competencia. No podrá

[14] Aunque es una organización sin ánimo de lucro más que una empresa, la Free Software Foundation durante diez años ha obtenido la mayoría de los fondos a partir de su servicio de distribución. Puede comprar artículos de la FSF (en http://www.gnu.org/order/order.html)) para apoyar su actividad.

obtener una ventaja en esta área, pero tampoco la competencia podrá tenerla frente a usted. Ambos competirán en otras áreas, mientras se benefician mutuamente en esta. Si su negocio consiste en vender un sistema operativo, no le gustará GNU, pero ese es su problema. Si su negocio es de otro ámbito, GNU puede salvarlo de ser empujado dentro del costoso negocio de la venta de sistemas operativos.

Me gustaría ver que el desarrollo de GNU se mantuviera gracias a donaciones de algunos fabricantes y usuarios, reduciendo el coste para todos[15].

"¿No merecen los programadores una recompensa por su creatividad?"

Si hay algo que merece una recompensa, es la contribución social. La creatividad puede ser una contribución social, pero solo en la medida en que la sociedad sea libre de aprovechar los resultados. Si los programadores merecen

[15] Un grupo de empresas de informática alrededor de 1991 reunió fondos para apoyar el mantenimiento del compilador C de GNU.

ser recompensados por la creación de programas innovadores, entonces, por la misma razón merecen ser castigados si restringen el uso de estos programas.

"¿No debería un programador poder pedir una recompensa por su creatividad?"

No hay nada malo en querer un pago por el trabajo o en buscar maximizar los ingresos personales, siempre y cuando no se utilicen medios que sean destructivos. Pero los medios habituales en el campo del software hoy en día se basan en la destrucción.

Extraer dinero de los usuarios de un programa limitando su uso es destructivo porque las restricciones reducen la cantidad y las formas en que el programa puede ser utilizado. Esto reduce la cantidad de beneficios que la humanidad obtiene del programa. Cuando hay una elección deliberada de restringir, las consecuencias dañinas son una destrucción deliberada.

La razón por la que un buen ciudadano no

utiliza estos medios destructivos para volverse más rico es que si todos lo hicieran, podríamos empobrecernos todos por una mutua destrucción. Esto es ética kantiana, o la Regla de Oro. Como no me gustan las consecuencias que resultarían si todos acapararan información, debo considerar como erróneo que alguien lo haga. Específicamente, el deseo de ser recompensado por la creatividad de uno no justifica privar al mundo en general de toda o parte de esa creatividad.

"¿No se morirán de hambre los programadores?"

Podría responder que nadie está obligado a ser programador. La mayoría de nosotros no puede conseguir dinero parándose en la calle y haciendo muecas. No estamos, por consiguiente, condenados a pasar nuestras vidas de pie en la calle haciendo muecas, y muriéndonos de hambre. Podemos dedicarnos a otra cosa.

Sin embargo, esta es una respuesta errónea porque acepta la suposición implícita del interrogador: que sin la propiedad del software a

los programadores no se les puede pagar un centavo. En este supuesto es todo o nada.

La verdadera razón por la que los programadores no se morirán de hambre es porque aún es posible que se les pague por programar, solo que no se les pagará tanto como en la actualidad.

Restringir la copia no es la única forma para hacer negocios con el software. Es la forma más común[16] porque es con la que se obtiene más dinero. Si se prohibiera o fuese rechazada por el comprador, el negocio del software se desplazaría hacia otras formas de organización que actualmente no se usan tan a menudo.

[16] Creo que me equivoqué al decir que el software privativo era la base más común para ganar dinero en el campo del software. Parece ser que en realidad el modelo de negocio más común era y es el desarrollo de software a medida, que no ofrece la posibilidad de percibir una renta, por lo que la empresa tiene que seguir haciendo el trabajo real para seguir recibiendo ingresos. El negocio del software a medida podrá seguir existiendo, más o menos igual, en un mundo de software libre. Por lo tanto, ya no supongo que los programadores ganarían menos en un mundo de software libre.

Siempre existen muchos modos para organizar cualquier tipo de negocio.

Probablemente la programación no será tan lucrativa bajo esta nueva forma como lo es actualmente. Pero esto no es un argumento en contra del cambio. No se considera una injusticia que los empleados en los comercios obtengan los salarios que ganan actualmente. Si los programadores ganaran lo mismo, no será tampoco una injusticia (en la práctica ganarán considerablemente más).

"¿La gente no tiene derecho a controlar cómo se usa su creatividad?"

El "control del uso de las ideas de alguien" realmente constituye el control de las vidas de otras personas, y por lo general se utiliza para hacerles la vida más difícil.

Las personas que han estudiado cuidadosamente el tema de los derechos de propiedad intelectual[17] (por ejemplo los abogados) dicen

[17]En la década de 1980 todavía no me había dado cuenta de lo confuso que era hablar de "la cuestión" de la "propiedad intelectual". Esa expresión es obvia-

que no existe un derecho intrínseco a la propiedad intelectual. Los tipos de los supuestos derechos de propiedad intelectual que reconoce el gobierno fueron creados mediante actos legislativos específicos con fines específicos.

Por ejemplo, el sistema de patentes se estableció para animar a los inventores a revelar los detalles de sus inventos. El objetivo era ayudar a la sociedad más que a los inventores. El periodo de validez de diecisiete años para una patente era corto comparado con el ritmo de desarrollo de la técnica. Dado que las patentes solo son relevantes para los fabricantes, para quienes el costo y el esfuerzo de un acuerdo de licencia son pequeños comparados con la puesta en marcha de la producción, las

---

mente prejuiciosa, más sutil es el hecho de que agrupa leyes dispares que plantean cuestiones muy diferentes. Hoy en día insto a la gente a rechazar completamente el término "propiedad intelectual", para no inducir a otros a pensar que esas leyes forman un tema coherente. Para hablar con claridad, hay que referirse a las patentes, el copyright y las marcas registradas por separado. Veáse una explicación más amplia (ver http://www.gnu.org/philosophy/not-ipr.html)) de cómo esta expresión genera confusión y prejuicios.

patentes a menudo no hacen mucho daño. No representan un obstáculo para la mayoría de los individuos que usan productos patentados.

La idea del copyright no existía en tiempos antiguos, cuando los autores frecuentemente copiaban extensivamente a otros autores en obras de no ficción. Esta práctica era útil, y ha sido la única forma de que las obras de muchos autores, aunque solo sea en parte, hayan sobrevivido. El sistema de copyright se creó expresamente con el propósito de promover la autoría. En el ámbito para el que se inventó –libros, que sólo podían copiarse de forma económica en una imprenta– hacía muy poco daño y no obstruía a la mayor parte de los individuos que leían los libros.

Todos los derechos de propiedad intelectual son solamente licencias otorgadas por la sociedad porque se pensaba, con razón o sin ella, que la sociedad en su conjunto se beneficiaría de su concesión. Pero, en cada situación particular, tenemos que preguntarnos: ¿nos beneficia realmente otorgar esta licencia? ¿qué tipo de acto le estamos permitiendo hacer a

una persona?

El caso de los actuales programas es muy diferente al de los libros de hace cien años. El hecho de que la forma más sencilla de copiar un programa sea de un vecino a otro, el hecho de que un programa esté formado tanto por el código fuente como el código objeto, siempre distintos, y el hecho de que el programa se use en lugar de leerlo y disfrutarlo, se combinan para crear una situación en la que una persona que hace valer un copyright está dañando a la sociedad en su conjunto tanto materialmente como espiritualmente; nadie debería hacerlo a pesar de que la ley se lo permita.

"La competición hace que las cosas se hagan mejor".

El paradigma de la competencia es una carrera: al premiar al ganador, estamos alentando a todos a correr más rápido. Cuando el capitalismo realmente funciona de esta manera, hace un buen trabajo; pero sus partidarios están equivocados al suponer que siempre funciona así. Si los corredores olvidan por qué se

otorga el premio y se centran en ganar sin importar cómo, pueden encontrar otras estrategias, como atacar a los otros corredores. Si los corredores se enredan en una pelea a puñetazos, todos llegarán tarde a la meta.

El software privativo y secreto es el equivalente moral de los corredores en una pelea a puñetazos. Es triste decirlo, pero el único árbitro que tenemos no parece objetar las peleas, solo las regula ("por cada diez metros que corras, puedes realizar un disparo"). Lo que debería hacer es separarlos y penalizar a los corredores, incluso por tratar de enredarse en una pelea.

"¿No dejarán todos de programar si no hay un incentivo económico?"

De hecho, mucha gente programará sin absolutamente ningún incentivo económico. La programación ejerce una atracción irresistible en algunas personas, generalmente en quienes son los mejores en ese ámbito. No hay escasez de músicos profesionales que sigan en lo suyo aunque no tengan esperanzas de ganarse

la vida de esa forma.

En realidad esta pregunta, aunque se formula muchas veces, no es adecuada para la situación. El pago a los programadores no va a desaparecer, solo se va a reducir. La pregunta correcta es: ¿Alguien programará si se reduce el incentivo económico? Mi experiencia muestra que sí lo harán.

Por más de diez años, muchos de los mejores programadores del mundo trabajaron en el Laboratorio de Inteligencia Artificial por mucho menos dinero de lo que podrían haber obtenido en otro sitio. Tenían muchos tipos de recompensas que no eran económicas: fama y aprecio, por ejemplo. Y la creatividad también es divertida, es una recompensa en sí misma.

Luego la mayoría se fue cuando se les ofreció la oportunidad de hacer ese mismo trabajo interesante por mucho dinero.

Lo que muestran los hechos es que la gente programa por razones distintas a la de la riqueza; pero si se les da la oportunidad de ganar también mucho dinero, eso los llenará

de expectativas y lo van a exigir. Las organizaciones que pagan poco no podrán competir con las que pagan mucho, pero no tendría que irles tan mal si las que pagan mucho fueran prohibidas.

"Necesitamos a los programadores desesperadamente. Si ellos nos pidieran que dejemos de ayudar a nuestro prójimo, tendríamos que obedecer".

Uno nunca está tan desesperado como para tener que obedecer este tipo de exigencia. Recuerde: millones para nuestra defensa, ¡pero ni un centavo para tributos![18]

"Los programadores necesitan tener alguna forma de ganarse la vida".

A corto plazo, esto es verdad. Sin embargo, hay bastantes maneras de que los programadores puedan ganarse la vida sin vender el derecho a usar un programa. Esta manera es frecuente ahora porque es la que les da a los

[18] Véase http://es.wikipedia.org/wiki/Caso_XYZ para más información sobre el contexto de esta sentencia.

programadores y hombres de negocios más dinero, no porque sea la única forma de ganarse la vida. Es fácil encontrar otras formas, si quieres encontrarlas. He aquí unos cuantos ejemplos:

Un fabricante que introduce una nueva computadora pagará por adecuar los sistemas operativos al nuevo hardware.

La venta de enseñanza, los servicios de asistencia y mantenimiento también pueden dar trabajo a programadores.

La gente con ideas nuevas podría distribuir programas como freeware[19], pidiendo donaciones a los usuarios satisfechos, o vendiendo servicios de asistencia. Yo he conocido a personas

[19]Posteriormente aprendimos a distinguir entre "software libre" y "freeware". El término "freeware" significa que el software se puede redistribuir libremente, pero por lo general no ofrece la libertad para estudiar y modificar el código fuente, así que la mayoría de esos programas no son software libre. Veáse "palabras y frases confusas que vale la pena evitar" (ver http://www.gnu.org/philosophy/words-to-avoid.html#Freeware)) para más detalles.

que ya trabajan así y con mucho éxito.

Los usuarios con que tengan las mismas necesidades pueden formar un grupo de usuarios y pagar sumas de dinero. Un grupo contratará a empresas de programación para escribir programas que a los miembros del grupo les gustaría utilizar.

Todo tipo de desarrollo puede ser financiado con un impuesto al software:

Supongamos que todos los que compren una computadora tengan que pagar un tanto por ciento de su precio como impuesto de software. El Gobierno entrega este dinero a una agencia como la la Fundación Nacional de las Ciencias (NSF) para que lo emplee en el desarrollo de software.

Pero si el comprador de la computadora hace por sí mismo un donativo para el desarrollo de software puede verse exento de este impuesto. Puede donar al proyecto de su elección –a menudo, elegido porque espera utilizar los resultados tan pronto como se haya completado. Puede tomar crédito por cada cantidad

de donación hasta la totalidad del impuesto que tenía que pagar.

La tasa total de impuesto podría decidirse mediante el voto de los contribuyentes, sopesada de acuerdo con la cantidad sobre la que se aplicará el impuesto.

Las consecuencias:

- La comunidad usuaria de computadoras apoya el desarrollo de software.
- Esta comunidad decide qué nivel de apoyo es necesario.
- Los usuarios a quienes les importa a qué proyectos se destine su parte pueden escogerlos por sí mismos.

A largo plazo, hacer programas libres es un paso hacia el mundo post-escasez, donde nadie tendrá que trabajar duro para ganarse la vida. La gente será libre para dedicarse a actividades entretenidas, como la programación, después de haber dedicado diez horas obligatorias a la semana a las tareas requeridas, como legislar, el asesoramiento familiar, la reparación

de robots y la exploración de asteroides. No habrá necesidad de ganarse la vida mediante la programación.

Hemos alcanzado ya una gran reducción de la cantidad de trabajo que la sociedad en su conjunto debe realizar para mantener su productividad actual, pero solo un poco de esta reducción se ha traducido en descanso para los trabajadores, dado que hay mucha actividad no productiva que se requiere para acompañar a la actividad productiva. Las causas principales de esto son la burocracia y las luchas isométricas contra la competencia. El software libre reducirá en gran medida estos drenajes en el campo de producción de software. Debemos hacerlo, para que los avances técnicos en la productividad se traduzcan en menos trabajo para nosotros.

4

El software libre es ahora aún más importante[1]

Por Richard Stallman[2]

Una versión considerablemente adaptada de este artículo se

[1] http://www.gnu.org/philosophy/free-software-even-more-important.es.html

[2] http://www.stallman.org/

publicó en Wired[3].

"Algunas sugerencias para colaborar con el movimiento del software libre"[4].

Han pasado treinta años desde la creación del movimiento del software libre, cuyo objetivo es promover el software que respeta la libertad de los usuarios y la comunidad. A este software lo llamamos "libre" (usamos esta palabra para enfatizar que nos referimos a la libertad, y no al precio[5]). Algunos programas privativos, como Photoshop, son muy caros; otros, como Flash Player, son gratuitos; en ambos casos, esos programas someten a los usuarios al poder del propietario del programa.

Mucho ha cambiado desde que empezamos. Hoy en día casi todo el mundo en los países avanzados posee ordenadores (a veces llamados "teléfonos") y se conectan a Internet con

[3] http://www.wired.com/opinion/2013/09/why-free-software-is-more-important-now-than-ever-before

[4] https://gnu.org/help

[5] En inglés, el término "free" puede significar "libre" o "gratuito".

ellos. El software privativo sigue sometiendo a los usuarios al control ajeno sobre sus tareas informáticas, pero ahora existe un nuevo medio para ello: el "servicio sustitutivo del software", o SaaSS, que significa dejar que el servidor de otra persona realice sus tareas informáticas.

Tanto el sofware privativo como el SaaSS pueden espiar al usuario, encadenarlo, e incluso atacarlo. Los abusos son habituales en los servicios y productos de software privativo porque los usuarios no tienen ningún control sobre ellos. Esta es la diferencia fundamental: el software privativo y el SaaSS están bajo el control de otra entidad (normalmente una corporación o un Estado). El software libre, por el contrario, pone el control en manos de los usuarios.

¿Por qué es importante el control? Porque la libertad consiste en poder ejercer el control de su propia vida. Si usted utiliza un programa para realizar actividades que afectan a su vida, su libertad depende del control que tenga sobre el programa. Usted merece tener el control

de los programas que utiliza, especialmente si los usa para hacer cosas que para usted son importantes.

Para que los usuarios puedan ejercer el control del programa, son necesarias cuatro libertades esenciales[6].

0. La libertad de ejecutar el programa como usted quiera, para cualquier propósito.

1. La libertad de estudiar el código fuente del programa y modificarlo para que haga lo que usted quiera. Los programadores escriben los programas en un determinado lenguaje de programación (algo así como inglés combinado con álgebra): eso es el "código fuente". Cualquiera que sepa programar y tenga el programa en forma de código fuente, puede leer este código, entender cómo funciona y también modificarlo. Cuando todo lo que tenemos es la forma ejecutable del programa (esto es, una serie de números que

[6] https://gnu.org/philosophy/free-sw.html

un ordenador puede ejecutar, pero cuya comprensión resulta extremadamente difícil para una persona), entender el programa y modificarlo se convierte en una tarea de suma complejidad.

2. La libertad de hacer copias exactas y distribuirlas cuando se desee. Esto no es una obligación, sino una opción. Si el programa es libre, esto no significa que usted tenga la obligación de facilitar copias, o que se las tengan que facilitar a usted. Distribuir programas sin las libertades es maltratar a los usuarios. Sin embargo, si no se distribuyen y se usan privadamente no se está maltratando a nadie.

3. La libertad de distribuir copias de sus versiones modificadas cuando lo desee.

Con las dos primeras libertades, cada uno de los usuarios ejerce el control sobre el programa individualmente. Con las otras dos libertades, cualquier grupo de usuarios puede ejercer un *control colectivo* sobre el programa.

Con todas las cuatro libertades, los usuarios controlan el programa. Si falta alguna de ellas, o si son inadecuadas, el programa es privativo (no es libre) e injusto.

Para actividades prácticas también se utilizan obras de otro tipo, como recetas de cocina, material pedagógico (libros de texto, manuales de consulta, diccionarios y enciclopedias), tipos de letra, diagramas de circuito para construir hardware o patrones para fabricar objetos útiles (no meramente decorativos) con impresoras 3D. Como no se trata de software, el movimiento del software libre no abarca estas obras en sentido estricto, pero aplica el mismo razonamiento y llega a la misma conclusión: tales obras también deben tener las cuatro libertades esenciales.

Con el software libre usted puede experimentar aportando modificaciones al programa para que haga lo que usted quiera (o deje de hacer algo que usted no quiera). Manipular software puede parecerle ridículo si usted está acostumbrado a las cajas herméticas del software privativo, pero en el mundo libre es algo

muy común, y además es una buena manera de aprender a programar. Incluso el pasatiempo tradicional de los norteamericanos de experimentar en la reparación de sus propios automóviles está siendo obstruida por el hecho de que hoy los coches contienen software privativo.

La injusticia de lo privativo

Si los usuarios no controlan el programa, el programa controla a los usuarios. En el caso del software privativo, siempre hay alguna entidad (el "propietario" del programa) que controla el programa y, a través del programa, ejerce su poder sobre los usuarios. Un programa que no es libre es un yugo, un instrumento de poder injusto.

En casos extremos (aunque tales casos se han generalizado bastante), los programas privativos están diseñados para espiar a los usuarios, restringirlos, censurarlos y abusar

de ellos[7]. Por ejemplo, todo esto lo hace el sistema operativo de las iCosas[8] de Apple, y también Windows en los dispositivos móviles con chips ARM. Windows, el firmware de los teléfonos móviles y el navegador Google Chrome para Windows incluyen una puerta trasera universal que permite a una cierta empresa modificar el programa de forma remota sin necesidad de pedir permiso. El Kindle de Amazon contiene una puerta trasera que puede borrar libros.

Con el objetivo de acabar con la injusticia del software privativo, el movimiento del software libre desarrolla programas libres para que los usuarios puedan liberarse. Comenzamos en 1984 desarrollando el sistema operativo libre GNU[9]. Hoy, millones de ordenadores funcionan con GNU, sobre todo en la combinación GNU/Linux[10].

[7] https://gnu.org/philosophy/proprietary.html

[8] Adaptación de "*iThings*", término ideado para referirse de manera lúdica a artefactos tales como *iPod*, *iPad*, *iPhone* y similares.

[9] https://gnu.org/gnu/the-gnu-project.html

[10] https://gnu.org/gnu/gnu-linux-faq.html

Distribuir programas sin conceder libertad supone un maltrato hacia los usuarios. Sin embargo, si un programa no se distribuye, no se estará maltratando a nadie. Si usted escribe un programa y lo usa de forma privada, esto no es malo para los demás. Estará perdiendo la oportunidad de hacer el bien, pero esto no es lo mismo que hacer el mal. Entonces, cuando decimos que todo el software debe ser libre, queremos decir que todas las copias de un programa deben conceder las cuatro libertades, no que todo el mundo tenga la obligación de ofrecer copias a los demás.

El software privativo y el SaaSS

El software privativo fue el primer medio que usaron las empresas para tomar el control de las tareas informáticas de las personas. Hoy existe otro medio, llamado "servicio sustitutivo del software" (SaaSS), que significa que un servidor ajeno realiza las tareas informáticas

del usuario.

El SaaSS no implica que los programas en ese servidor sean privativos (aunque suelen serlo). Sin embargo, usar un SaaSS provoca las mismas injusticias que usar un programa privativo: son dos caminos que conducen al mismo lugar dañino. Tomemos el ejemplo de un SaaSS de traducción: el usuario envía un texto al servidor, el servidor lo traduce (del inglés al español, por ejemplo) y devuelve la traducción al usuario. De esta forma, el trabajo de traducción está bajo el control del administrador del servidor, no del usuario.

Si usted usa un SaaSS, quien controla el servidor controla sus tareas informáticas. Esto implica confiar todos los datos relevantes al administrador del servidor, quien además estará obligado a mostralos al Estado; entonces, ¿a quién sirve realmente ese servidor?[11]

[11] https://gnu.org/philosophy/who-does-that-server-really-serve.html

Injusticias primarias y secundarias

Cuando usted usa programas privativos o el SaaSS, en primer lugar se está haciendo mal a sí mismo, ya que le está concediendo a otra persona un poder injusto sobre usted. Por su propio bien, debería evitarlo. Si se compromete a no compartir, también estará perjudicando a otros. Respetar tal compromiso es malo, y romperlo es menos malo, pero para ser honesto de verdad, no debe comprometerse en absoluto.

Hay casos en los que el uso de software privativo ejerce presión directa sobre otras personas para que hagan lo mismo. Skype es un claro ejemplo: cuando alguien usa el cliente del programa privativo Skype, está forzando a otra persona a que también lo use y, por lo tanto, que también renuncie a sus libertades. Google Hangouts presenta el mismo problema. Es incorrecto hacer propuestas como esas. Debemos rechazar el uso de esos programas, aun-

que sea brevemente, incluso en el ordenador de otra persona.

Utilizar programas privativos y el SaaSS conlleva otro perjuicio: premia al instigador, promueve el desarrollo de ese programa o "servicio", y conduce a que más y más personas caigan bajo el dominio de la empresa en cuestión.

Todas las formas de daño indirecto adquieren una mayor dimensión cuando el usuario es un ente público o una escuela.

El software libre y el Estado

Los entes públicos existen para los ciudadanos, no para sí mismos. Cuando realizan tareas informáticas, lo hacen para los ciudadanos. Tienen el deber de conservar el control absoluto sobre esas tareas a fin de garantizar su correcta ejecución en beneficio de los ciudadanos. En esto consiste la soberanía informática del Estado. Nunca deben permitir que el

control de las tareas informáticas del Estado caiga en manos privadas.

Para conservar el control de las tareas informáticas que realizan en nombre de los ciudadanos, los entes públicos no deben usar software privativo (software que está bajo el control de entidades que no son estatales). Tampoco deben delegar la realización de esas tareas a un servicio programado y ejecutado por un ente distinto del Estado, porque eso sería un SaaSS.

El software privativo no ofrece protección alguna contra un peligro crucial: su desarrollador. Y el desarrollador podría ayudar a otros a perpetrar un ataque. Antes de corregir los errores de Windows, Microsoft los muestra a la NSA, la agencia de espionaje digital del gobierno de EEUU[12]. No sabemos si Apple hace lo mismo, pero está bajo la misma presión gubernamental que Microsoft.

[12] http://arstechnica.com/security/2013/06/nsa-gets-early-access-to-zero-day-data-from-microsoft-others/

Software libre y educación

Las escuelas (y todas las instituciones educativas) influyen sobre el futuro de la sociedad a través de lo que enseñan. Para que esta influencia sea positiva, deben enseñar exclusivamente software libre. Enseñar el uso de un programa privativo equivale a imponer la dependencia, que es lo contrario de la misión educativa. Capacitando a los alumnos en el uso del software libre, las escuelas dirigirán el futuro de la sociedad hacia la libertad, y ayudarán a los programadores talentosos a dominar el oficio.

También enseñarán a los estudiantes el hábito de cooperar y de ayudar a los demás. En todas las aulas se debe aplicar la siguiente regla: "Alumnos, este es un lugar donde compartimos nuestro conocimiento. Si traéis software al aula, no podéis quedároslo para vosotros. Debéis compartir copias con el resto de la clase, incluyendo el código fuente en caso de que algún otro quiera aprender. Por eso no se permite traer software privativo a clase, excepto

para someterlo a la ingeniería inversa".

Los desarrolladores de software privativo querrían que penalizáramos a los buenos estudiantes que comparten software y frustráramos a aquellos que son lo bastante curiosos como para querer modificarlo. Esto significa impartir una mala educación. En la sección http://www.gnu.org/education/ encontrará más información acerca del uso de software libre en las instituciones educativas.

Software libre: Mucho más que "ventajas"

A menudo me piden que describa las "ventajas" del software libre. Pero el término "ventajas" es demasiado débil cuando se trata de la libertad. La vida sin libertad es tiranía, y eso se aplica a la informática y a cualquier otra actividad de nuestras vidas. Debemos rechazar conceder el control de nuestras tareas de computación a los propietarios de un progra-

ma o de un servicio informático. Es lo que hay que hacer por razones egoístas, aunque no solo por razones egoístas.

La libertad incluye el ser libre de cooperar con los demás. Negar esta libertad equivale a mantener a las personas divididas, primer paso para tiranizarlas. En la comunidad del software libre somos muy conscientes de la importancia de la libertad para cooperar porque nuestro trabajo consiste en una cooperación organizada. Si un amigo suyo viene a visitarlo y lo ve usando un programa, puede pedirle una copia. Un programa que le impide a usted que lo redistribuya, o le indica que "no debe hacerlo", es antisocial.

En informática, la cooperación incluye redistribuir copias exactas de un programa entre otros usuarios. También incluye distribuir sus versiones modificadas. El software libre estimula estas formas de cooperación, mientras que el software privativo las prohíbe. Prohíbe redistribuir copias, y al impedir que los usuarios tengan el código fuente, también les impide modificar los programas. El SaaSS tiene

los mismos efectos: si usted realiza sus tareas de computación en una web alojada en un servidor ajeno, mediante una copia ajena de un programa, no puede ver ni tocar el software que se está usando para hacerlas, y por lo tanto no puede redistribuirlo ni modificarlo.

Conclusión

Todos merecemos tener el control de nuestra propia actividad informática. ¿Cómo podemos conseguirlo? Rechazando el software que no es libre en los ordenadores que nos pertenecen o que usamos regularmente, y rechazando el SaaSS; desarrollando software libre[13] (para los que somos programadores); rehusando desarrollar o promover software privativo o el SaaSS; difundiendo estas ideas[14].

Nosotros, y otros miles de usuarios, lo venimos haciendo desde 1984, y gracias a

[13] https://gnu.org/licenses/license-recommendations.html

[14] http://www.gnu.org/help

eso hoy tenemos el sistema operativo libre GNU/Linux, que cualquiera puede usar, sea programador o no. Únase a nuestra causa, ya sea como programador o como activista. Hagamos que todos los usuarios de ordenadores sean libres.

Traducción: Sergi Ruiz Trepat, 2014.

Revisiones: Javier Fdez. Retenaga.

[15] http://creativecommons.org/licenses/by-nd/3.0/us/deed.es