

INTRODUCCION AL SOFTWARE LIBRE *

Mikel Egaña Aranguren (pik@sindominio.net)

Soraluze-Septiembre-2002

Índice

| | |
|---|----------|
| 1. Introducción-corolario | 1 |
| 2. Historia del software libre | 2 |
| 2.1. El movimiento Hacker en los 70 | 2 |
| 2.2. La crisis de la comunidad: la FSF y GNU comienzan su difícil andadura | 3 |
| 2.3. Linux de Linus Torvalds: el Kernel que vino del frío | 4 |
| 2.4. Eric Raymond: El modelo de desarrollo tipo Bazar | 4 |
| 2.5. El boom de GNU/linux, distribuciones comerciales, Debian, open source... | 5 |
| 3. Licencias: GPL y BSD | 5 |
| 4. Peligros de cara al futuro | 6 |
| 5. Como usar, aportar y divertirse con el Software libre | 7 |
| 6. Conclusión: el software libre y los movimientos sociales | 8 |
| 7. Miscelánea | 8 |

1. Introducción-corolario

El movimiento de software libre es un movimiento *político* de *inteligencia colectiva* que a través del software libre representa una alternativa a una

*©2002 Mikel Egaña Aranguren (pik). Se otorga permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre GNU, Versión 1.1 o cualquier otra versión posterior publicada por la Free Software Foundation. Se considerará como Secciones Invariantes todo el documento, no habiendo Textos de Portada ni Textos de Contra Portada. Puede consultar una copia de la licencia en: <http://www.gnu.org/copyleft/fdl.html>

sociedad del conocimiento mercantilizada y vertical. Lo contrario al software libre es el software propietario.

El software libre nos da unas libertades/derechos:

- Ejecutar el programa, para cualquier propósito.
- Estudiar el funcionamiento del programa, y adaptarlo a nuestras necesidades (Hace falta tener acceso al código fuente¹).
- Redistribuir copias del programa.
- Mejorar el programa, y poner sus mejoras a disposición del público, para beneficio de toda la comunidad (Hace falta tener acceso al código fuente).

Estas libertades se consiguen a través de una licencia especial llamada GPL (General Public License).

La FSF (Free Software Foundation) es su organización más importante, GNU (GNU IS NOT UNIX) su proyecto más ambicioso y Linux su Proyecto más avanzado y famoso, aunque hay más; HURD,...

Usamos, desarrollamos y promovemos el software libre por ser *libre*, no por ser *bueno*. Hay gente que lo usa por que técnicamente es mejor, sin pararse a pensar en la implicaciones político-sociales que el software libre tiene detrás.

2. Historia del software libre

El movimiento del software libre es algo bastante complejo, que se entiende mucho mejor viendo como surgieron sus diferentes matices y en qué momento surgieron estos matices. Vamos a dividir la historia del software libre en algunas etapas que no tienen límites muy exactos, pero para aclararnos servirá:

¹El código fuente de un programa es como los planos de una máquina compleja. Un programa al fin y al cabo no es más que una serie de instrucciones escritas en un lenguaje específico (C, perl, Java, ...). El código fuente es el fichero de texto que tiene todas esas instrucciones. Luego ese fichero se pasa por un programa especial, llamado compilador, que entiende ese lenguaje y lo transforma en un programa totalmente funcional que se comunica con el ordenador a un nivel muy básico.

2.1. El movimiento Hacker en los 70

En los 70, las ciencias telemáticas (informáticas) se desarrollaron mucho en las universidades como el MIT² o Berkeley. En aquella época surgió el movimiento hacker³ de la mano de un puñado de estudiantes apasionados por la telemática, que, en sus propios laboratorios, hackeaban sin parar y compartían todo el software que producían sin ninguna limitación. Por tanto, el software primigenio era tan libre que ni siquiera había que mencionar la palabra *libre*, ya que todo el mundo ejercía sin restricciones las cuatro libertades antes mencionadas, tan sólo preocupándose de lo puramente técnico y de la diversión. Uno de esos Hackers era Richard Stallman, y trabajaba feliz en el laboratorio de Inteligencia Artificial del MIT.

En esta época surgió un sistema operativo⁴ llamado UNIX, junto con su lenguaje de programación llamado C, que revolucionó las ciencias informáticas por su calidad. Hoy en día, UNIX y sus derivados (GNU/Linux, *BSD, ...) son de los sistemas operativos más potentes que se usan.

2.2. La crisis de la comunidad: la FSF y GNU comienzan su difícil andadura

Entre los 70 y comienzos de los 80, todo ese mundo cambió drásticamente cuando las empresas empezaron a viciarlo con sus intereses, basados en el copyright, licencias restrictivas, etc. Una de las víctimas del proceso fué el propio sistema UNIX, que sufrió una gran fragmentación, ya que cada empresa cerraba el código fuente del UNIX que producía.

Muchos Hackers empezaron a ser contratados por empresas y dejaron de compartir lo que producían, incluso no podían siquiera hablar de lo que estaban trabajando. Richard Stallman vió como sus compañeros iban cayendo, hasta que quedaron unos pocos, que se enfrentaban a la siguiente disyuntiva: seguir trabajando en software en una empresa produciendo software propietario o no seguir trabajando en software. Richard Stallman optó por lo segundo, de modo que se quedó sin trabajo, aunque el MIT accedió a

²Massachusetts Institute of Technology, uno de los mejores centros del mundo de investigación tecnológica.

³El término *Hacker* ha sufrido una deformación bastante grave por parte de los medios de comunicación y las autoridades. Un hacker, en su acepción original, es alguien apasionado por la tecnología que le gusta usar su talento para darle usos alternativos, ingeniosos y originales a la misma, compartiendo sus conocimientos con sus compañeros. Un hacker no busca inmiscuirse en un sistema ajeno para sacar algún tipo de provecho, eso es un *cracker*.

⁴El sistema operativo de un ordenador es lo que hace que funcione, es lo que comunica los programas con el hardware, es como un programa muy grande en el que se ejecutan todos los demás programas. El sistema operativo reside en el disco duro del ordenador. Un ordenador sin sistema operativo es como una persona en coma, está viva pero no funciona.

dejarle usar sus instalaciones.

Fué entonces cuando Richard Stallman vió la necesidad de crear y promover el software libre, en contraposición al modelo que estaba destrozando su mundo, un mundo en el que los hackers compartían la información sin preocuparse por nada más. Así nació la FSF, Free Software Foundation, que coordinaba la creación y uso de software libre (Software bajo licencia GPL).

Pero no tenía sentido producir software libre si se tenía que ejecutar en un sistema operativo propietario. De modo que nació GNU: GNU is Not Unix. El objetivo de ese proyecto era (es) crear y mantener un sistema operativo totalmente libre, bajo licencia GPL. Una labor realmente titánica, ya que un sistema operativo es algo realmente complejo, bizantino. A Richard Stallman se le tildó de loco, pero siguió adelante. El sistema sería tipo UNIX, y respetaría todos los estándares de programación y comunicaciones para que todo el mundo pudiese usarlo sin problemas. Para tal efecto, Richard Stallman lo primero que creó fueron dos programas que todavía hoy son famosos por su calidad y potencia: EMACS y GCC. Emacs es un procesador de texto para escribir los programas. GCC es un compilador (GNU C Compiler) para compilar los programas.

Empezó el proceso de creación del sistema operativo, y el desarrollo se centró en la distribución, dejando el Kernel para más adelante ⁵.

2.3. Linux de Linus Torvalds: el Kernel que vino del frío

Siguió el desarrollo del sistema operativo GNU, y ya había un kernel en desarrollo, el HURD, pero era muy inmaduro. Había en Finlandia alguien que creía completamente en el software libre (Linus Torvalds), y se empezó a hacer un núcleo para sí mismo, que vino en denominar Linux (Linus + Unix). Publicó en la red su núcleo, licenciándolo bajo GPL. En seguida muchos desarrolladores se sumaron al proyecto, y el kernel se juntó con la distribución (Algo así como si el Dr. Frankenstein juntase el cerebro con el cuerpo). Hoy en día se sigue con el desarrollo de HURD como alternativa cada vez más viable a Linux. Así se creó el binomio GNU + Linux. También existe el binomio GNU + HURD, aunque es menos popular.

⁵Los sistemas operativos tienen la siguiente estructura: un Kernel (O núcleo) y una distribución. El kernel es lo más importante, es lo que se comunica con el hardware, distribuye recursos entre programas, ...la distribución es todo lo demás; los programas, el sistema de archivos, el interfaz gráfico...En un coche el kernel sería el motor y la distribución la carrocería. El kernel es lo más importante, y se puede cambiar sin cambiar la distribución.

2.4. Eric Raymond: El modelo de desarrollo tipo Bazar

Eric Raymond, conocido Hacker y protagonista de una de las primeras crisis de la comunidad, tipificó el modelo de desarrollo del kernel de Linux como modelo *bazar*, en contraposición al modelo *catedral*. En una catedral todo funciona bajo una estricta jerarquía que es inflexible. En cambio, en un bazar, todo es totalmente horizontal, nadie tiene más poder que los demás, y sin embargo el orden acaba por emerger. En el caso del software libre, el orden emerge del *bazar* como un fenómeno de inteligencia colectiva. Este tipo de desarrollo tiene sus ventajas (Comentamos también las técnicas, pero las políticas son prioritarias):

- Políticas o estructurales:
 - El desarrollo tecnológico es accesible a todo el mundo, no a unas pocas élites.
 - Tenemos la seguridad de que nadie está metiendo código malicioso en los programas que usamos, ya que el desarrollo es supervisado por miles de programadores en todo el mundo. Después del 11-S este punto adquiere una importancia vital. No es casualidad que el gobierno Alemán haya retirado todos sus sistemas windows de sus embajadas.
 - El desarrollo tecnológico va a al ritmo de las exigencias de los usuarios, no al ritmo de los departamentos de marketing.
- Técnicas:
 - Seguridad: el desarrollo abierto hace que el software sea más seguro, ya que hay millones de ojos supervisando el código para encontrar vulnerabilidades.
 - El desarrollo abierto produce software de mejor calidad.

2.5. El boom de GNU/linux, distribuciones comerciales, Debian, open source...

A principios de los noventa el dúo GNU/linux empezó a popularizarse, y empezaron a desarrollarse grandes distribuciones de la mano de empresas, que usaban el mismo núcleo (Linux), aunque no todas eran totalmente GNU, ya que incluían software propietario. Algunas distribuciones famosas son: RED HAT, SuSE, GENTOO,... (Hay unas doscientas). Hay una distribución que es la única que sigue fielmente los criterios de la FSF y usa sólo software libre. Por otra parte, es mantenida totalmente por voluntarios, no habiendo ninguna empresa detrás. Por consiguiente, políticamente es la más coherente. Se llama DEBIAN.

A la par del boom de las distribuciones, el conocido hacker Eric Raymond, en un movimiento posibilista, decidió que el término *software libre* no era bueno para las empresas, y encabezó un movimiento que promulgaría el uso de la palabra *software de código abierto* para referirse al software libre. Según Eric Raymond, así el software libre llegaría a más gente de la mano de las empresas. Utilizando el término *open source* (Código abierto) el software libre perdía toda su carga política e ideológica. Mucha gente no estuvo de acuerdo y se produjo una de las primeras grandes divisiones dentro de la comunidad del software libre. Una de las empresas más famosas que se adhirió a esta tendencia fue Netscape con su navegador web. Es importante no mezclar el software libre y el software de código abierto, aunque técnicamente son prácticamente lo mismo, filosóficamente no tienen nada que ver.

3. Licencias: GPL y BSD

Dentro del software libre hay dos licencias principales:

- GPL: General Public License.

Aunque en principio pueda parecer contradictorio, el software libre tiene copyright, y ese copyright nos da potestad para licenciarlo bajo GPL. Es un copyright un tanto especial, se llama *copyleft* y se describe como *all rights reversed*, haciendo alusión al copyright tradicional, que dice que todos los derechos están reservados para el productor (*all right reserved*), pero en este caso se les *da la vuelta* (*reversed*) y recaen sobre el *consumidor*⁶. El ponerle un copyleft al software es una garantía jurídica de que ese software siempre será GPL, que ninguna empresa o particular que coga nuestro código lo licenciará como software propietario. O, en caso de hacerlo, podríamos demandarla y probablemente ganaríamos el juicio.

Por otra parte, el software libre (Con copyleft y licenciado bajo GPL) tiene un carácter vírico, ya que si se usa para construir otros programas estos también deben estar licenciados bajo GPL. Es decir, si un desarrollador decide incluir código GPL en su programa debe licenciar todo el programa como GPL. Es como si alguien estuviese construyendo un coche y decidiese usar un carburador GPL; tendría que licenciar todo el coche con GPL. Es como una infección, es algo vírico, como un virus. De esta manera la GPL se extiende.

⁶El término *consumidor* no es el más adecuado, ya que precisamente una de las cualidades del software libre es que la barrera entre consumidor y productor se difumina.

- BSD: Berkeley Software Distribution.

También es considerada software libre, aunque difiere de la GPL en que no es vírica, es decir, cualquiera puede usar código licenciado bajo BSD y licenciarlo como quiera. Esta licencia se utiliza en los sistemas operativos *BSD, que son diferentes de GNU.

Hay muchas licencias y matices, pero estas son las dos principales, y la GPL es la que más se usa. En esta dirección está todo bastante detallado:

<http://www.gnu.org/licenses/license-list.html>

4. Peligros de cara al futuro

Aquí se listan los peligros de cara al futuro para el software libre. En realidad todos se engloban en un intento mucho más amplio por parte de los gobiernos y las multinacionales de la información de acallar un movimiento social que promulga la compartición del conocimiento, la cultura y la tecnología. Es en ese mismo sentido donde está actuando por ejemplo la SGAE (Sociedad General de Autores y Editores) intentando criminalizar a la gente que comparte música. Así que la lucha en realidad es mucho más amplia.

- Patentes de software: las patentes de software impedirían a los desarrolladores utilizar mucho código, al estar éste patentado. En estados unidos están ya funcionando, habiendo patentes tan absurdas como la operación lógica XOR. Hay más información en:

<http://proinnova.hispalinux.es/>

- Distribuciones mezclando software libre y software propietario: como ya se ha comentado antes, muchas distribuciones mezclan software libre y propietario, confundiendo así al usuario. Este no es consciente de lo que está usando y por qué, y de nada sirve que alguien use software libre si no entiende su trasfondo político.
- Hardware secreto: muchos fabricantes de hardware no dan acceso a las especificaciones técnicas de sus dispositivos, o lo hacen sólo a compañías como Microsoft. Así, muchos desarrolladores de software libre tienen que deducir como funcionan los dispositivos a base de ingeniería inversa⁷.

⁷La ingeniería inversa es un proceso dentro del hacking que consiste en analizar un dispositivo o programa hasta conocer todos sus fundamentos. Sería como deducir los planos detallados de un coche desmontándolo a mano (Pero más difícil, claro).

- Campañas en contra del soft libre: FUDs (Fear, Uncertainty, Doubt): grandes lobbies como Microsoft intentan desprestigiar el movimiento del software libre confundiendo al gran público sobre la dificultad y supuestas características negativas del software libre.
- LSSI: Ley Sobre Los Servicios de la Información. Es una ley, ya aprobada aunque todavía no aplicada, que supondrá un serio retroceso de la libertad de expresión en la red. Más información en:

<http://kriptopolis.com>

5. Como usar, aportar y divertirse con el Software libre

- El software libre no es elitista, precisamente lucha contra una idea de elitismo que han impuesto las elites mercantilistas del software: quieren que creamos que el desarrollo del software solo atañe a gente hiper cualificada e hiper pagada. De hecho, los lobbies del software propietario son los primeros interesados en crear incultos informáticos, usuarios que delegan absolutamente todo el conocimiento tecnológico en supuestos expertos. Ese delegacionismo hace también que el uso del software sea totalmente irracional y empujado por el ritmo del mercado.
- Como participar:
 - Usando soft libre, y sabiendo por qué se usa.
 - Difundiendo la idea del software libre.
 - Mandando fallos de los programas. El software libre no es perfecto, y si al usarlo se detecta algún fallo, es muy apreciable mandar una descripción del mismo al desarrollador del programa. O mejor todavía, arreglarlo uno mismo y publicar la corrección.
 - Creando diseños, iconos, etc...para los programas e interfaces gráficos.
 - Por supuesto, desarrollando *in sensu esctrieto*, es decir, programando.
 - Ayudando a crear documentación de los programas, traduciendo documentación ya existente,...
 - Ayudando a otros a usar software libre.
- Como empezar: Lo mejor es conseguir una distribución (A ser posible DEBIAN, pero se puede empezar con una más fácil como MANDRAKE), leerse la documentación e intentar instalarla en el ordenador. Hay muchísima ayuda disponible en la red, ya que la cultura

del compañerismo está muy desarrollada en el mundo del software libre. Aún así, siempre se intenta animar a los *principiantes* a que intenten valerse por sí mismos en el mundo de GNU/Linux (para evitar el delegacionismo tecnológico antes mencionado).

6. Conclusión: el software libre y los movimientos sociales

El software libre supone una forma de autogestión en un campo que tradicionalmente no ha formado parte de los movimientos antagonistas en general: la tecnología y el conocimiento. Así, el software libre es parte de una revolución aún mayor, una revolución *cognitiva* donde se lucha contra el delegacionismo tecnológico al que quieren empujarnos muchos gobiernos y entidades. Se podría decir que el software libre es una forma de autogestión cognitiva, un cuestionamiento radical de la lógica polarizada que ha gobernado la tecnología los últimos 20 años. Muchos movimientos sociales no lo ven así, y choca ver como alguien se rasga las vestiduras por ir a un McDonalds pero usa sin problemas el software propietario de Microsoft, que, aparte de ser propietario y por tanto estructuralmente injusto, seguramente sea un instrumento de vigilancia de bastante eficiencia.

7. Miscelánea

- GNU: el sitio oficial GNU es: <http://www.gnu.org> (En castellano <http://www.es.gnu.org>).
- El sitio oficial FSF es: <http://www.fsf.org>
- Debian: el sitio oficial Debian es: <http://www.debian.org>
- Barrapunto: el punto de encuentro de la comunidad de software libre de habla hispana: <http://barrapunto.com>
- Hay algunos buenos textos sobre software libre y telemática antagonista en la biblioweb de sindominio: <http://sindominio.net/biblioweb>
- Para buscar ayuda: <http://forogugs.sindominio.net> (Una buena búsqueda en <http://google.com> también ayuda).
- Metabolik BioHacklab (<http://sindominio.net/metabolik>): un hacklab es un laboratorio donde se hackea, se enseña, se aprende, se lucha y se disfruta de la tecnología desde un punto de vista social. Este Hacklab está en el gaztetxe de Udondo, y uno de sus proyectos más importantes es indymediaEH: <http://indymedia.euskalherria.org>. En el estado hay otros 4, y se juntan en un evento llamado Hackmeeting

(<http://sindominio.net/hackmeeting>), en el que también toman parte personas que no son de los hacklabs. Este año el Hackmeeting es en Madrid, en el centro social ocupado y autogestionado Laboratorio, los días 4, 5 y 6 de octubre.

| |
|--|
| Documento realizado mediante L ^A T _E X 2 _ε en un sistema GNU/linux (Debian Woody) |
|--|