Software Libre

Introducción

Los inicios

En la edad temprana de la informática, en las décadas de los 60 y los 70, los laboratorios de grandes instituciones bullían de actividad. Personas apasionadas por la tecnología desarrollaban su trabajo con los enormes y carísimos *mainframes* (grandes ordenadores de la época).

Era totalmente natural y habitual que el trabajo de estos científicos fuese compartido, como en cualquier otra área científica con la que se desea que la humanidad progrese. Así que los primeros *hackers* (expertos en informática) se ayudaban entre sí sin reparos. El software programado era libre de circular, de usarse y de modificarse, un bien común que crecía fructíferamente en este caldo de cultivo. El desarrollo de bienes públicos basados en ese modelo fue exponencial hasta el punto de que **gran parte de la tecnología en la que se basa hoy Internet** –desde el sistema operativo Unix hasta los protocolos de red– procede de esos años.

Desafortunadamente, a principios de los años ochenta, esta forma de funcionar entra en crisis. Los ordenadores que habían sido carísimos y poco potentes comienzan a hacerse asequibles para un mayor público. Y las grandes empresas ven la oportunidad de vender todo tipo de software de forma separada del ordenador. Y cada copia la vendían sin su **código fuente** para que la competencia lo tuviera más difícil. La nueva industria del software comienza a apoyarse en la legislación sobre propiedad intelectual, expandiéndose y popularizándose.

La cultura *hacker* comienza a sufrir la desinformación por parte de las grandes empresas. Los sistemas comienzan a hacerse incompatibles entre sí, no se dispone del código fuente, y la comunidad se va desmembrando. Los *hackers* son comprados por las empresas y obligados, por contrato, a no ayudar a los colegas de su antigua comunidad. Las dificultades no terminan ahí, las herramientas de programación básicas, como compiladores, depuradores, editores y demás piezas imprescindibles se tornaron propietarias y muy caras.

La antigua comunidad *hacker* parecía destinada a la extinción. Aunque algunos resistían los nuevos tiempos mercantilistas, estaba claro que era cuestión de tiempo y aguante.

El proyecto GNU

Los últimos *hackers* de la antigua comunidad, resistían en algunos laboratorios, intentando seguir trabajando como antaño evitando ser comprados por la nueva industria del software. Tenían convicciones éticas muy profundas respecto al trabajo que hacían y lo que querían aportar al mundo durante su vida. Su espíritu científico les impedía actuar de otra forma.

En uno de estos últimos reductos, trabajaba Richard M. Stallman, un *hacker* del emblemático **Laboratorio de Inteligencia Artificial del Massachussets**

Institute Technology (MIT). Stallman pudo ver cómo su antigua comunidad iba desapareciendo. Pero la gota que colmó el vaso fue que la empresa que les vendió una impresora se negara a facilitarles el código fuente del software que la hacía funcionar. De esta manera tenían que pelearse con el vendedor pidiendo que arreglaran fallos o agregaran nuevas funcionalidades.

Stallman, como él mismo dice, tuvo que tomar una decisión moral radical. O aceptar la nueva forma de funcionar o luchar desde cero por hacer resurgir su antigua comunidad de las cenizas. Y pensando que no deseaba desperdiciar su vida poniendo cadenas y no ayudando a sus colegas abandonó el MIT y comenzó el proyecto que bautizó como GNU (1984).

La tarea que se propuso el proyecto GNU era enorme: construir un sistema operativo completo y libre. Un proyecto de este tipo sólo es viable para grandes empresas con miles de trabajadores en nómina; además estaba la dificultad de carecer de las herramientas básicas, como compiladores y editores. Por otro lado, aún no existía una Internet popular y el modelo bazar (un desarrollo con participación libre y anárquica por parte de los programadores) que hizo posible Linux años más tarde era imposible que se diera.

En 1985, creó la **Fundación por el Software Libre (FSF)**. Desarrolló, prácticamente solo, un compilador, un editor y ciertas piezas básicas del sistema, con una visión estratégica del objetivo asombrosa. Todo esto a la vez que aumentaba el número de *hackers* que colaboraban con la fundación.

Aquellos *hackers* se dieron cuenta de que necesitaban algo más para mantener su software tal como era, libre. Así que trabajaron ideando la manera de proteger la libertad del software de forma jurídica. Finalmente Stallman escribe el **manifiesto GNU**, una declaración de principios, y más tarde, inspirado por el manifiesto GNU, la **General Public License (GPL)**. La GPL fue uno de los mayores logros de la FSF; protegía la libertad del software que desarrollaban en el ámbito jurídico.

Stallman le da la vuelta al **copyright** e inventa el **copyleft**. Un concepto entre broma y reivindicación del deseo de los científicos por compartir el conocimiento. La licencia GPL, que promueve el copyleft, trata de proteger la libertad del software.

Pero ¿qué es el Software libre?

El software, en general, es como una **«Receta» que le indica al ordenador qué ha de hacer**.

El **Código ejecutable (o binario)**:

- Es una ristra de 0 y 1 que el ordenador sabe interpretar.
- Codifica las «recetas» que el ordenador sabe ejecutar.
- Es difícil de escribir, leer y **casi imposible** modificar para las personas.

El **Código fuente**:

- «Receta» codificada en un lenguaje de alto nivel.
- El ordenador no sabe ejecutarlo, hay que traducirlo a código ejecutable.

- Ejemplo «Hello world»

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
puts("Hello World!");
return EXIT_SUCCESS;
}
```

 En este video Richard Stalman hace una analogía entre el software y las recetas de cocina.

El Software libre es, simplificando mucho el concepto, el **software que va acompañado del código fuente** y que por tanto se puede estudiar y mejorar.

Asegura 4 libertades:

- “libertad 0”, ejecutar el programa con cualquier propósito (privado, educativo, público, comercial, militar, etc.)
- “libertad 1”, estudiar y modificar el programa (para lo cual es necesario poder acceder al código fuente)
- “libertad 2”, copiar el programa de manera que se pueda ayudar al vecino o a cualquiera
- “libertad 3”, Mejorar el programa y publicar las mejoras

El software no libre restringe alguna de estas libertades. Normalmente sólo se ofrece con el código binario. Por ello el **software no libre** recibe el nombre de **privativo**.

El núcleo Linux

Volvamos a la historia del proyecto GNU. A principios de los 90, el proyecto GNU ha crecido mucho, existe una comunidad como antaño y la cantidad de software libre crece exponencialmente. La licencia GPL protege el trabajo de la comunidad.

GNU casi puede sostenerse por sus propios pies, el sistema libre es casi una realidad. Sólo falta una pieza, pero fundamental: el núcleo. Esta pieza es tan fundamental como difícil de desarrollar. Se iba retrasando reiteradamente. La llamaban **Hurd** (actualmente sigue sin acabar).

Ajeno a todo esto, un estudiante de informática Finlandés se frustra con las limitadas posibilidades del *kernel* de propósito académico, **Minix**, desarrollado por su profesor, A. Tanenbaum. Con mucho talento y ánimo decide comenzar un proyecto de pruebas con el que crear un núcleo que saque el máximo partido a los nuevos procesadores de 32 bits de Intel, los 386.

Los ordenadores eran habituales en las casa e Internet comenzaba a hacerse popular. A los pocos meses tenía un desarrollo primitivo y rudimentario de su sistema. Y decidió algo de lo que no imaginaba su repercusión: usando los grupos de *news* de Internet, publica el código que había programado y pide ayuda para avanzar con el proyecto; su motivación principal era simplemente

lúdica.

Inusitadamente el interés por su núcleo crece rápidamente. Cada vez tiene más colaboradores que prueban, corrigen y desarrollan nuevas funcionalidades en el sistema. El poder de Internet comienza a vislumbrase, haciendo crecer el nuevo núcleo de forma distribuida (modelo de desarrollo bazar) como nadie hubiera podido imaginar.

En 1992 ya se dispone de un *kernel* primitivo, pero funcional. La gente de GNU hizo el resto del milagro permitiendo que el recién publicado *kernel*, llamado **Linux**, pudiera funcionar con el resto de su sistema. Había nacido **GNU/Linux** el primer sistema operativo completo libre.

Durante el resto de la década el nuevo sistema no para de crecer acompañado por el auge de Internet. Multitud de talentos en informática se sienten atraídos por el software libre y GNU/Linux, la comunidad de *hackers* florece como nunca.

Emergen multitud de nuevos proyectos para cubrir cualquier tipo de carencia en el ámbito del software libre. Aparece el servidor web Apache, haciendo del sistema GNU/Linux un compañero inseparable de Internet.

Más tarde aparecen proyectos como KDE o GNOME para crear escritorios libres. Estamos en la segunda mitad de la década de los noventa. Estos proyectos maduran, creando nuevas comunidades y dando soporte a la creación de todo tipo de aplicaciones libres.

Se crean empaquetados de GNU/Linux para todos los gustos y necesidades. A esta recopilación de software se le llama **distribución**. LliureX es una más de ellas, enfocada al sistema educativo valenciano.

También van apareciendo nuevos modelos de negocio basados en el software libre. Creando una industria que permite hacer negocios manteniendo la libertad del software producido. Sobre todo se basa el negocio en los servicios, el software deja de ser un producto. Mientras más extendido esté el software libre más necesidad de servicios sobre éste habrá. Por eso se incentiva la copia y la distribución gratuita, al contrario que el software privativo.

GNU/Linux es, hoy día, seguramente el mayor logro de Internet, y seguramente uno de los mayores y más complejos patrimonios de la humanidad.

Distribuciones GNU/Linux

Una distribución GNU/Linux se define como un conjunto de programas que permiten tanto la instalación en el ordenador del Sistema Operativo Linux como su uso posterior. Su objetivo es facilitar la instalación, la configuración y el mantenimiento de un sistema GNU/Linux. Integra un núcleo, un conjunto de aplicaciones de sistema y una colección de programas de usuario listos para instalar. Son como los helados que están todos hechos con la misma materia prima y los hay de muchos sabores. Cada sabor sería una distribución GNU/Linux.

Actualmente, existen muchas distribuciones diferentes basadas en GNU/Linux. Las hay para toda clase de ordenadores y dispositivos electrónicos: ordenadores portátiles o de sobremesa, pocketPC o PDA, puntos de acceso de redes wireless, teléfonos móviles, etc. La naturaleza del software libre permite esto: cualquiera puede coger el código desarrollado hasta el momento y

adaptarlo a sus propias necesidades. Es un hecho que aumenta el número de empresas y usuarios que eligen sistemas basados en GNU/Linux por sus elevadas prestaciones y la cantidad de software disponible.

De todos modos, aunque existen decenas de distribuciones, hay algunas más populares que se han extendido mucho. La filosofía de software libre hace que muchas empresas que han creado sus propias distribuciones de GNU/Linux no restrinjan el acceso a su código. Aun así, el **soporte que ofrecen y el material que venden les aporta beneficios**, permitiendo su subsistencia. Asimismo cabe considerar que en muchas de estas distribuciones se incluye software propietario que algunos usuarios prefieren, si bien en muchos casos existen programas homólogos con licencia libre.

Dos de las distribuciones GNU/Linux más conocidas y utilizadas se describen brevemente a continuación:

- **Debian GNU/Linux**: una de las primeras distribuciones de GNU/Linux que aparecieron y aún siguen existiendo y evolucionado. Debian fue fundada en agosto de 1993 por Ian Murdock, tenía como objetivo construir una distribución GNU/Linux de forma abierta y comunitaria, desvinculada de intereses comerciales particulares. Está desarrollada por un grupo de colaboradores voluntarios distribuidos por todo el mundo y no cuenta con el respaldo de ninguna empresa. Aunque **es de las más estables y seguras** que existen, su sistema de instalación y configuración necesita de conocimientos previos.

- **Ubuntu**: el 8 de julio de 2004 el sudafricano Mark Shuttleworth y la empresa Canonical Ltd. anunciaron la creación de la distribución Ubuntu (http://www.ubuntu.com/). Tras varios meses de trabajo y un breve período de pruebas, la primera versión de Ubuntu fue lanzada el 20 de octubre de 2004. A lo largo de 2005 esta distribución, basada en Debian y en el escritorio GNOME, ha ido sumando usuarios hasta convertirse a día de hoy en una de las mejores y más populares versiones de GNU/Linux.

Lliurex es la distribución de GNU/Linux de la Generalitat Valenciana. Esta basada en Edubuntu (una distribución basada en Ubuntu y orientada a la Educación).

Existen multitud de distribuciones libres, muchas derivadas y especializadas a partir de otras. En http://upload.wikimedia.org/wikipedia/commons/8/8c/Gldt.svg se puede ver las distribuciones y su evolución temporal. La FSF mantiene un listado de distribuciones completamente libres (no incluyen ningún programa privativo).

¿Por qué utilizar Software libre?

Richard Stallman ha dedicado a este tema un interesante artículo titulado «Por qué las escuelas deberían usar exclusivamente software libre» (http://www.gnu.org/philosophy/schools.es.html) cuya lectura recomendamos. Muy resumidamente, en dicho artículo se exponen los siguientes puntos:

- El software libre supone un ahorro de costes para las escuelas.

- Si las escuelas enseñan software libre, entonces los estudiantes utilizarán software libre cuando se gradúen. Esto ayudaría a que la sociedad en su conjunto se librara del dominio (y abuso) de las megacorporaciones.
- El software libre les permite a los estudiantes aprender cómo funciona. El software libre anima a todos a aprender.
- La misión fundamental de las escuelas es enseñar a ser buenos ciudadanos y buenos vecinos: «si traen software a la escuela, deben compartirlo con los demás niños».

No debe separarse GNU/Linux de su filosofía de la libertad. No se elige para un entorno educativo únicamente por su eficiencia (que también), si hacemos lo contrario estamos equivocados.

No se trata de sustituir un sistema operativo por otro porque sea más barato, seguro y fiable. Enseñar con GNU/Linux no es sólo usarlo sino **trasmitir el espíritu de colaboración y cooperatividad que implica cualquier empresa de conocimiento**. El software libre es en sí mismo educativo por los valores que le acompañan.

Cuando un profesor enseña a sus alumnos con una aplicación propietaria se coloca en una auténtica encrucijada, porque esto obliga al alumno a comprar el software o a copiarlo ilegalmente. Ahora bien, si el profesor no le deja copiar el programa está negando su ayuda y si lo deja copiar está enseñando al alumno que se puede violar la ley cuando ésta no nos gusta o va contra nuestro propio interés.

A estas consideraciones éticas pueden añadirse varios argumentos puramente técnicos que justificarían por sí solos el uso de software libre.

10 razones para usar software libre

- **Porque es sólido como una roca.**

Por ejemplo *Debian*, al igual que algunas otras distribuciones de GNU/linux, busca la estabilidad por encima de todo.

- **Porque tiene una larga tradición.**

El proyecto GNU es el primer movimiento visible para difundir el Software Libre.

- **Porque no le afectan virus ni troyanos.**

No existe programa al 100% seguro contra virus, pero hoy en día los sistemas basados en GNU/Linux son más estables y menos propensos a ataques de virus y troyanos que otros sistemas. ¿Qué nos deparará el futuro? ¿Quién lo sabe? El presente es de Linux.

- **Porque tiene exactamente lo que necesitas**.

El Software Libre no se mueve por ventas ni márketing. No hay versiones nuevas cada año, sino cuando son necesarias. Por eso, programas como OpenOffice.org te ofrecen las funciones que necesitas, ni más ni menos.

- **Porque permite hacer prácticamente de todo**.

¿Quién dijo que el Software Libre era simple e inútil? Existen programas como Blender, ampliamente extendidos entre los desarrolladores 3D, que demuestra que hay programas libre muy complejos y especializados.

- **Porque te da libertades, no te las quita**.

La libertad de distribución y modificación de los programas son los que hacen grande al SL.

- **Porque tiene una amplia comunidad que te apoya**.

Si tienes un problema con Windows, ¿qué haces? Reinstalas el sistema. ¿Por qué? ¿Crees que la informática realmente está tan atrasada como para no poder solucionar tus problemas?.

- **Porque, si quieres, puedes aprender con él**.

Al estar disponible el código del programa y tener la libertad para modificarlo, cualquiera que quiera aprender informática puede examinar este código.

- **Porque es transparente y no te engaña**.

Puedes examinar lo que hace realmente. Y si tú no sabes, tranquilo, otro lo hará por tí. ¿Sabes realmente lo que hace tu sistema operativo cuando escribes tu número de cuenta corriente en tu ordenador? Yo sí.

- **Porque posiblemente ya lo estés usando, ¡sin saberlo!**

Emule es uno de los muchos ejemplos de SL. Está totalmente extendido, funciona de maravilla, y ¡tiene mil y una modificaciones en la red!. La gran mayoría de servidores de Internet funcionan con SL.

(Fuente: http://weblog.topopardo.com/?p=1220)

Copyright

 Un vídeo bastante aclaratorio de qué es el la propiedad Intelectual y el copyright

Según la wikipedia:

...

Licencias: GPL, BSD, dominio público

De nuevo, veamos qué dice la wikipwdia al respecto de qué es una licencia:

Licencias privativas

Las **licencias de software no libre** establecen los derechos de uso, distribución, redistribución, copia, modificación, cesión y en general cualquier otra consideración que se estime necesaria. Por lo general, **no permiten que el software sea modificado**, desensamblado, **copiado o distribuido** libremente, regula el **número de copias que pueden ser instaladas** e **incluso los fines concretos para los cuales puede ser utilizado**. La

mayoría de estas licencias **limitan fuertemente la responsabilidad derivada de fallos en el programa**.

Algunos ejemplos de este tipo de licencias son las llamadas CLUFs: Contrato de Licencia para Usuario Final o EULAs: End User License Agreement, por sus siglas en Inglés.

Licencias libres

En http://es.wikipedia.org/wiki/Licencia_de_software hay una detallada explicación de las distintas licencias libres que existen. A continuación detallamos las más importantes (o al menos las más utilizadas)

dominio público

Cuando un autor cede un software al dominio publico, renuncia a todos sus derechos (salvo la autoría) y por tanto se permite uso, copia, modificación o redistribución para cualquier fin. No es necesario acreditar al autor.

Licencia BSD

La licencia BSD es la licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution). Es una licencia de software libre permisiva. Esta licencia tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público. La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre.

Hay que dar crédito a los autores.

Licencia Apache

La licencia Apache (Apache License o Apache Software License para versiones anteriores a 2.0) es una licencia de software libre creada por la Apache Software Foundation (ASF). La licencia Apache (con versiones 1.0, 1.1 y 2.0) requiere la conservación del aviso de copyright y el disclaimer, pero no es una licencia copyleft, ya que no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas.

La licencia Apache y la BSD son muy similares, aunque la licencia Apache incluye una clausula que garantiza que si el código contiene el uso de alguna patente, este puede ser utilizado igualmente (sin tener que pagar por el uso de dicha patente).

Licencia GPL

La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License o simplemente sus siglas del inglés GNU GPL, es una licencia creada por la Free Software Foundation en 1989 (la primera versión, escrita por Richard Stallman), y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

- Hay que dar crédito a los autores.

- Si se utiliza software con licencia GPL en un programa, éste debe continuar teniendo la licencia GPL (vacuna o virus?).

Existen varias licencias “hermanas” de la GPL, como la licencia de documentación libre de GNU (GFDL), la Open Audio License, para trabajos musicales, etc.

Licencia LGPL

La Licencia Pública General Reducida de GNU, o más conocida por su nombre en inglés GNU Lesser General Public License o simplemente por su acrónimo del inglés GNU LGPL, es una licencia de software creada por la Free Software Foundation que pretende garantizar la libertad de compartir y modificar el software cubierto por ella, asegurando que el software es libre para todos sus usuarios.

La principal diferencia entre la GPL y la LGPL es que la última puede enlazarse a (en el caso de una biblioteca, 'ser utilizada por') un programa no-GPL, que puede ser software libre o software no libre. A este respecto, la GNU LGPL versión 3 se presenta como un conjunto de permisos añadidos a la GNU GPL.

Modelos de negocio

Fuente: http://www.slideshare.net/kikebar/modelos-de-negocio-con-software-libre

http://curso-sobre.berlios.de/introsobre/2.0.1/sobre.html/book1.html

Puede decirse que son muchos los modelos de negocio que se están explorando alrededor del software libre, algunos más clásicos y otros más innovadores.Hay que tener en cuenta que entre los modelos más habituales en la industria del software no es fácil usar los basados en la venta de licencias de uso de programas producidos, pues en el mundo del software libre ese es un mecanismo de financiación muy difícil de explotar. Sin embargo, sí se pueden utilizar los basados en el servicio a terceros, con la ventaja de que sin ser necesariamente el productor de un programa puede darse soporte completo sobre él.

El modelo tradicional se basa en

- venta de licencias de utilización de un paquete
- sin posibilidad de modificación
- uso bajo limitaciones de distribución, copias y uso
- servicio de actualizaciones periódicas, formación, consultoría,...
- a veces, “recompra” anual

En el mundo del software libre **es difícil cobrar licencias de uso** (como hacen muchas empresas que venden software privativo), pero no imposible. En general, no hay nada en las definiciones de software libre que impida que una empresa cree un producto y sólo lo distribuya a quien pague una cierta cantidad. Por ejemplo, un determinado productor podría decidir distribuir su producto con una licencia libre, pero sólo a quien le pague 1.000 euros por

copia (de forma similar a como se hace en el mundo clásico del software propietario).

Sin embargo, aunque esto es teóricamente posible, en la práctica es bastante difícil que suceda. Porque una vez que el productor ha vendido la primera copia, quien la recibe puede estar motivado a tratar de recuperar su inversión vendiendo más copias a menor precio. Es fácil deducir cómo este proceso continuaría en cascada hasta la venta de copias a un precio cercano al coste marginal de copia, que con las tecnologías actuales es prácticamente cero.

Producto vs Servicio

- **Producto**: bien físico tangible susceptible de ser entregado a un comprador y que implica la transferencia de propiedad del vendedor hacia el comprador
- **Servicio**: la provisión de servicios es una actividad económica que no tiene consecuencias de propiedad

El software como “producto”:

- No es un bien físico
- Coste marginal=0 ⇒ el precio no tiene relación real con los costes reales de producción.
- No hay transferencia de propiedad
- El comprador no es propietario
- El productor decide las condiciones

Problemas derivados del modelo actual:

- Necesidad de mantener la información y el conocimiento como secreto
- Complejidad cada vez mayor del software
- Actualizaciones y nuevas versiones constantes y no siempre necesarias
- Fuerte dependencia de unos pocos proveedores
- Desventaja de las industrias locales del software
- Barreras de entrada de las PyMES al software privativo cada vez más elevadas

El software libre se basa en un cambio de paradigma:

- “Pasar de vender software como producto a vender software como servicio”
- Consultoría
- Integración
- Personalización
- Mantenimiento
- Formación

Enlaces y Material complementario

...

- El software libre ahorró a Brasil 225 millones de dólares en el 2010
- El software libre ahorra a la economía europea 450 mil millones de euros al año

Tracedump:

```
newBaseSize: 12pt
newBaseSizeInPt: 12
```