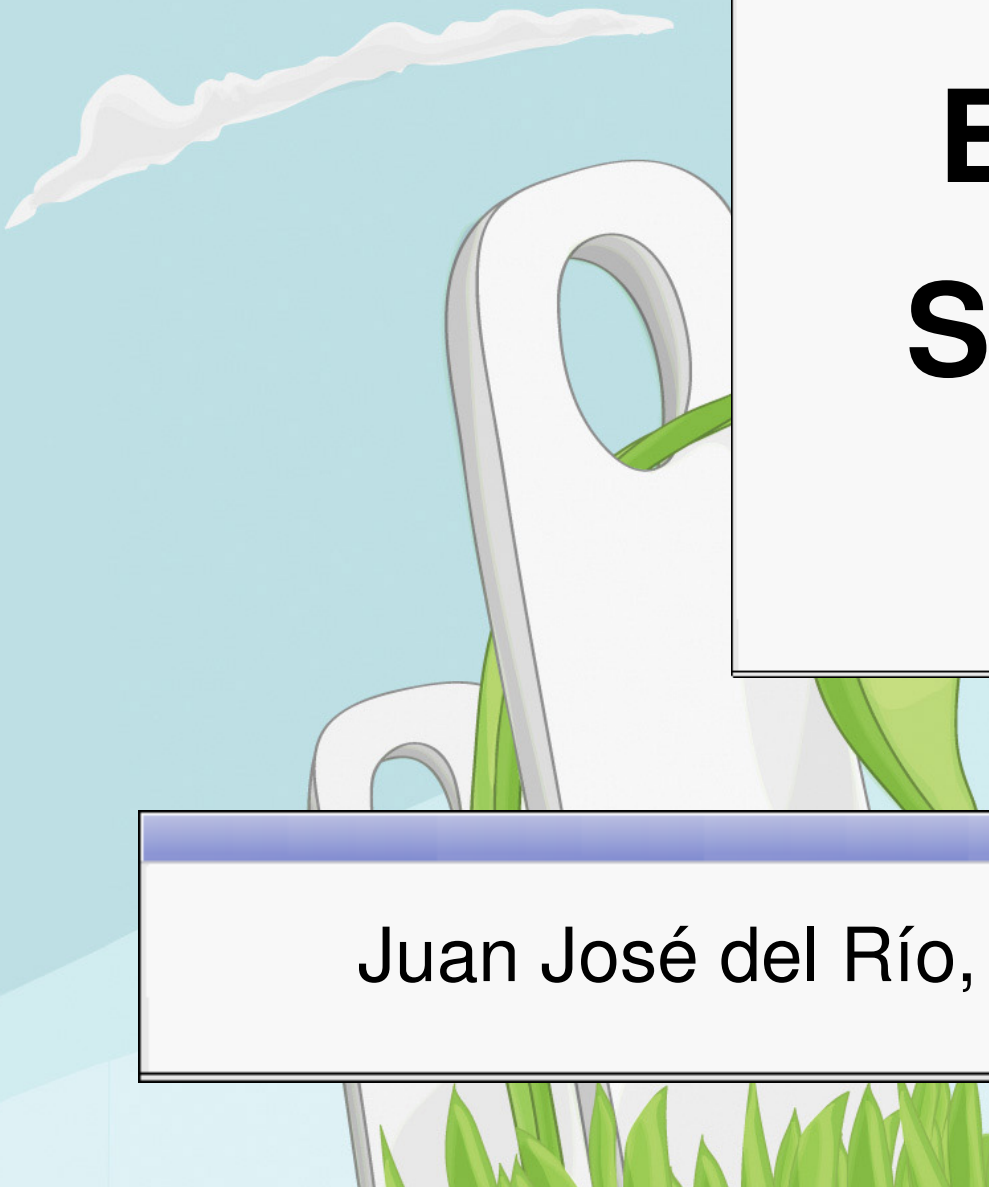


Explicando el Software Libre

Juan José del Río, Juan Miguel Taboada

Introducción al Software Libre

¿Qué es el Software Libre?

¿Cómo influye la licencia en el Software?

¿Qué motivación tienen los desarrolladores?

¿Cómo se financian los proyectos?

¿Cuál es la panorámica general del Software Libre?

Libertad del Software

Libertades del Software Libre según Richard Stallman

(fundador de la Free Software Foundation)

- **0** Ejecutar el programa en cualquier sitio,
con cualquier propósito, y para siempre
- **1** Libertad para estudiarlo y adaptarlo
a nuestras necesidades. (Exige acceso al código fuente)
- **2** Libertad de redistribución, de modo
que se nos permita colaborar con vecinos y amigos.
- **3** Libertad para mejorar el programa y
publicar las mejoras. (Exige acceso al código fuente)

Libertad del Software

¿Cómo aplicar las libertades a un programa?

Con una licencia libre, que plasma tanto las libertades, como restricciones compatibles con ellas.

Consecuencias

No significa software gratuito: este software se puede vender.

Bajos ingresos por distribución debido a la tercera libertad (excepto para grandes volúmenes: distribuciones).

Motivaciones para el Software Libre

Motivación ética

Abanderada por la Free Software Foundation (FSF).

Partidaria del apelativo “libre”: el Software es conocimiento y debe poder difundirse sin trabas.

La ocultación de Software es antisocial y la posibilidad de modificar programas es una forma de libertad de expresión.

Existen ensayos de Richard Stallman y Pekka Himanen.

Motivaciones para el Software Libre

Motivación pragmática

Abanderada por la Open Source Iniciative (OSI).

Partidaria del apelativo “fuente abierta”.

Argumenta que el Software de este tipo tiene ventajas técnicas y económicas.

Consecuencias de la libertad del Software

Para el usuario final

- Encuentra competencia en un mercado que tiende al monopolio.
- No hay que “depender” de la fiabilidad del fabricante, sino que puede observar la calidad de un programa por el nivel de aceptación entre los demás usuarios.
- Es improbable que un producto útil desaparezca porque una empresa deje de desarrollarlo.

Consecuencias de la libertad del Software

Para el usuario final

- Para la evaluación de productos no hay más que instalarlos, no hay que negociar pruebas con los proveedores.
- Se puede adaptar el programa a las necesidades personales.
- Se puede modificar el programa para corregir errores.
- Se mejora el proceso de corrección de errores con respecto a los programas de fuente cerrada.

En definitiva, **el control pasa del proveedor al usuario**.

Consecuencias de la libertad del Software

Para la Administración pública

- Le posibilita cumplir los requisitos de accesibilidad, neutralidad respecto del fabricante, integridad, utilidad, privacidad y seguridad de los datos a largo plazo.

- Tiene la oportunidad de hacer que empresas locales puedan desarrollar, mantener, adaptar o auditar el Software de la Administración.

Consecuencias de la libertad del Software

Para el desarrollador

- Es más fácil competir y adquirir tecnología punta.

- Puede aprovecharse del trabajo de los demás, incluso competir con un producto modificando su código; si bien el competidor también se aprovechará de nuestras modificaciones.

- Distribución de Software barata y global.

Consecuencias de la libertad del Software

Para el integrador

- No más ingeniería inversa para encajar “cajas negras”.
- Puede “limar asperezas” para conseguir una mejor integración en sus productos.
- Dispone de una inmensa colección de Software de donde sacar aplicaciones para su producto integrado.

Consecuencias de la libertad del Software

Para el que proporciona mantenimiento y servicios

- Lo sitúa casi en las mismas condiciones que el productor.

- Se aprecia mejor el valor añadido de los programas, ya que el coste es bajo.

Este es actualmente el negocio más claro con Software Libre y con el que es posible mayor competencia.

Descanso para preguntas

A continuación:

Historia del Software Libre

Historia del Software Libre

El Software comenzó libre, y permaneció así durante sus primeros años.

Luego, años más tarde la situación cambió completamente y se cerró el acceso al código fuente y se empezaron a vender licencias de uso.

En los años 80 se empezó a trabajar para conseguir formalizar el software libre, y la situación comentó a invertirse.

Hoy en día, tras los esfuerzos de últimos 20, existe una gran cantidad de Software Libre, y algunas aplicaciones son líderes en el mercado.

Historia del Software Libre

Y en principio fue libre...

En los años 60, época de los grandes supercomputadores, IBM era la mayor empresa (con gran diferencia).

El Software se daba junto al Hardware y no se separaba.

El Software se distribuía con su código fuente (algunas veces tan sólo se distribuía el código fuente).

No existía el término “Software Libre”, porque no existía otra forma de Software que no fuera esa.

Historia del Software Libre

Auge del Software cerrado

A partir del año 70, IBM deja de distribuir el Software junto al hardware, y comienza a cobrar por él.

El Software se comenzó a ver como algo con valor por sí mismo.

Se comienzan a restringir las libertades del Software.

A mediados de los 70, existía Software cerrado para casi cada ámbito informático.

Historia del Software Libre

Años 70 y 80

Aún siendo abrumadoramente mayor la “exploración” en el campo del Software cerrado, aparecen iniciativas cercanas al Software Libre.

Algunas de ellas son Spice, TeX y Unix.

Unix fue el primer sistema operativo portable. Fue creado en los “Bell Labs” de AT&T. Su desarrollo comenzó en 1972, y se sigue desarrollando hasta hoy, con innumerables variantes.

Historia del Software Libre

Desarrollo temprano de Unix

Durante 1973 y 1974, Unix llegó a universidades y centros de investigación con una licencia que permitía su uso para fines académicos.

Aunque no era de libre distribución, el funcionamiento es similar al que se observó más tarde en las comunidades de Software Libre, y surgieron comunidades de desarrolladores.

Unix se considera, hasta cierto punto, un ensayo temprano de lo que sería GNU, Linux, BSD, etc...

Historia del Software Libre

Problemas con Unix

Unix también fue un ensayo temprano de los sistemas propietarios, que a primera vista compartían algunas características del Software Libre. A finales de los años 70, y durante los 80, AT&T cambió su política respecto a Unix y el acceso a nuevas versiones se hizo difícil y caro.

Todo esto llegó al punto en el que AT&T demandó a la Universidad de Berkeley por publicar el código de Unix BSD, que es la variante de Unix que la Universidad de Berkeley había desarrollado.

Historia del Software Libre

El comienzo: BSD, GNU

GNU/FSF: nace el movimiento del Software Libre

En 1984, Richard Stallman abandona el “AI Lab” del MIT para comenzar el proyecto GNU. Esto se debe a su desacuerdo con los contratos de exclusividad y a la no compartición.

La idea de la Free Software Foundation (FSF) era y es la construcción de un sistema de Software completo, de propósito general, pero completamente libre. El proyecto se llamó GNU (GNU's Not Unix). Las primeras herramientas para GNU fueron un compilador de C (GCC) y un editor de textos (Emacs).

Historia del Software Libre

El comienzo: BSD, GNU
GNU/FSF: nace el movimiento del Software Libre

Richard Stallman estaba preocupado por la licencia del Software, así que creó la licencia GPL (GNU Public License), que es considerada una licencia copyleft, pues utiliza restricciones para evitar que se puedan recortar las libertades del usuario.

A principios de los años 90 estaba cerca de conseguir un sistema completo, pero aún faltaba una pieza fundamental: el núcleo (kernel).

Historia del Software Libre

El comienzo: BSD, GNU

El Computer Science Research Group (CSRG) de Berkeley

El CSRG fue un grupo que colaboró activamente en el desarrollo de Unix. Su desarrollo fue tan famoso, que fue una de las dos fuentes fundamentales de Unix, junto a AT&T.

Pasado un tiempo, AT&T que era la poseedora de los derechos sobre Unix, decidió demandar a la Universidad de Berkeley por su desarrollo de un núcleo que compartía núcleo de Unix llamado BSD.

Los sistemas BSD son hoy en día los más antiguos y consolidados en el mundo del Software Libre.

Historia del Software Libre

En busca de un núcleo

El proyecto GNU desarrolla desde hace más de diez años un núcleo para GNU llamado Hurd.

Para principios de los años 90, GNU sólo necesitaba un núcleo, pero Hurd aún necesitaba mucho tiempo para ser adecuado, así que se plantearon dos proyectos que lo sustituirían temporalmente: BSD y Linux.

Historia del Software Libre

La familia *BSD

En esa época (prinicipios de los 90) BSD consiguió finalmente reescribir todo el código heredado de Unix, y tuvo así un núcleo totalmente libre.

Conforme pasó el tiempo, aparecieron variantes de BSD llamadas OpenBSD, NetBSD y FreeBSD.

El problema que tenía BSD era la demanda de AT&T, todavía pendiente, que desanimaba a utilizar este núcleo.

Historia del Software Libre

GNU/Linux entra en escena

En Julio de 1991 Linus Torvalds, estudiante finlandés de Informática, anuncia en Internet que quiere implementar un sistema libre similar a Minix.

En Marzo de 1994 apareció Linux 1.0, la primera versión estable.

Durante ese periodo se unieron cientos de desarrolladores a Linux que colaboraron en el núcleo, y adaptaron las herramientas de GNU.

A mediados de 1992 empiezan a aparecer las primeras distribuciones de Linux, como SLS (ahora Slackware).

Historia del Software Libre

Finales de los años 90

A mediados de los 90 el Software Libre ya ofrece entornos completos que permiten el trabajo diario. Aún así, faltan modernas interfaces gráficas de usuario (GUI).

En estos años, el Software Libre comienza a tener repercusión mediática, y aparecen textos que versan sobre Software Libre.

En esta época, Netscape Navigator perdió la batalla de los navegadores frente a Internet Explorer; así que ocurrió el hecho inesperado por parte de Netscape de liberar el código fuente de su navegador, surgiendo así el proyecto Mozilla.

Historia del Software Libre

Finales de los años 90

Esta liberación de código fuente por parte Netscape (una gran empresa), despertó el interés sobre el Software Libre en las demás empresas.

Los mercados financieros también se interesaron y Red Hat fue una de las empresas que más se beneficiaron.

En esta época el Software Libre experimenta un gran auge, y surgen multitud de empresas basadas en éste (distribución, servicios añadidos, etc).

Historia del Software Libre

Finales de los años 90

Grandes empresas, como IBM, se posicionan claramente a favor del Software Libre. La mayoría, exploran el Software Libre con diversas estrategias (portar elementos a plataformas libres, dar soporte, etc).

Desde el punto de vista técnico, lo más destacable de esta época es la aparición de dos grandes proyectos para construir entornos de escritorio: KDE (Octubre 1996) y GNOME (Agosto 1997).

Historia del Software Libre

Finales de los años 90

Apache mantiene desde sus principios (1995) la mayor cuota en servidores web. GCC es el compilador de C más portable y uno de los más usados. GNAT se hace con la mayor parte del mercado de Ada 95 en pocos años. Y así sucesivamente...

En 1998 se crea la Open Source Iniciative (OSI), que es la organización que abandera el movimiento Open Source.

Historia del Software Libre

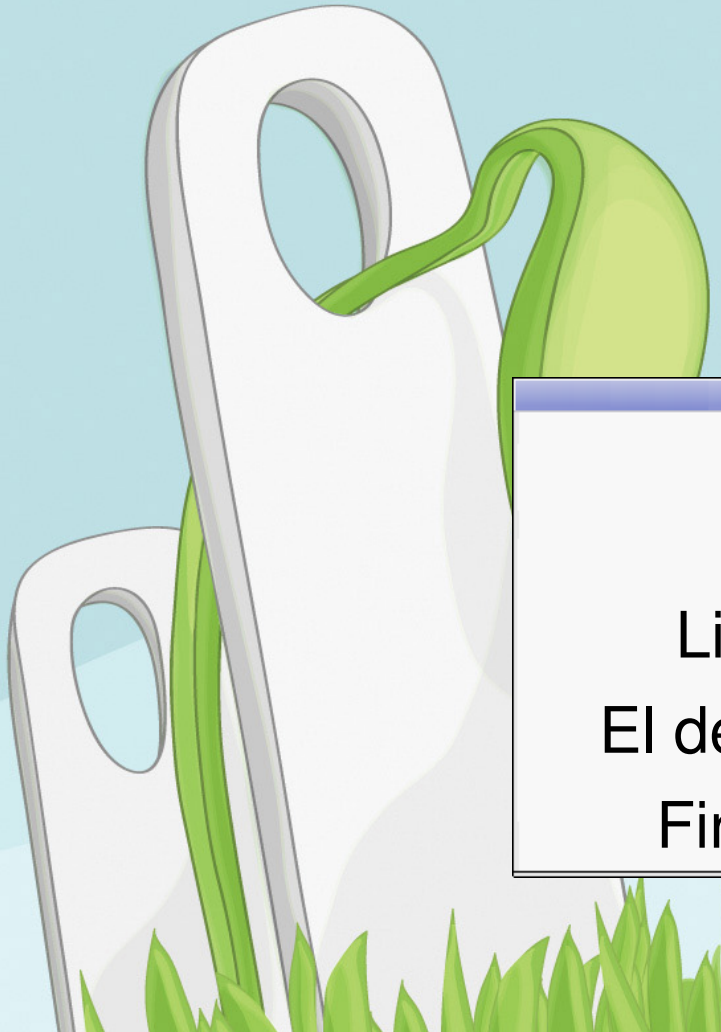
Principios del 2000

El Software Libre es un serio competidor en el mercado de los servidores, y empieza a hacerse un hueco entre los usuarios con GNOME 2.x, KDE 3.x y OpenOffice.

Los sistemas libres, comparados con los propietarios, son similares en facilidad de instalación y mantenimiento.

Todas las empresas están posicionadas con respecto al Software Libre: la mayoría a favor, IBM completamente a favor, y Microsoft completamente en contra.

Descanso para preguntas



A continuación:

Licencias en el Software Libre

El desarrollador y sus motivaciones

Financiación del Software Libre

Licencias en el Software Libre

Lo que diferencia al Software Libre del resto es su licencia de uso.

En la licencia el autor especifica qué puede hacer el usuario con su obra: uso, redistribución, modificación, etc., y en qué condiciones.

Aunque estas diferencias en las licencias puedan parecer pequeñas, luego se reflejan en innumerables diferencias en los métodos de desarrollo, redistribución, financiación, etc.

Existen diversas licencias libres, siendo las más famosas las licencias: BSD y GPL.

Licencias en el Software Libre

Licencia BSD (Berkeley Software Distribution)

Esta licencia tuvo su origen en la publicación de versiones de Unix realizadas por la Universidad de Berkeley.

La única obligación que impone es la de dar crédito a sus autores, mientras que permite tanto la redistribución binaria, como la del código fuente, aunque no obliga a distribuir éste último.

Licencias en el Software Libre

Licencia GPL (GNU Public License)

Es la licencia más conocida en el mundo del Software Libre, y su autoría corresponde a la Free Software Foundation.

La licencia GPL hace un uso curioso de la legislación sobre copyright, pues consigue un efecto totalmente contrario al típico. Es por ello, que se la conoce como una licencia de tipo copyleft (all rights reversed).

Esta licencia obliga a distribuir la aplicación como código fuente o, en caso que se distribuya la aplicación compilada, se pueda acceder libremente al código fuente de ésta.

Licencias en el Software Libre

Licencia GPL (GNU Public License)

Esta licencia se considera “viral”, porque establece que las aplicaciones que sean una modificación de una aplicación bajo licencia GPL ha de estar asimismo bajo licencia GPL, o una licencia compatible (una que no contradiga ninguna cláusula de la GPL).

La licencia GPL está diseñada para asegurar la libertad del código, ya que un programa licenciado con GPL es muy difícil cerrarlo.

El desarrollador y sus motivaciones

Se desconocen los recursos humanos que hay detrás del Software Libre dado que las colaboraciones son en su mayor parte anónimas.

Aún así, en estos últimos años se han intentado reunir datos sobre este aspecto, de forma que se pueda saber más sobre el movimiento del Software Libre.

Entre las incógnitas más importantes es la motivación que impulsa a los desarrolladores a colaborar en proyectos a sabiendas de que los beneficios económicos, al menos los directos, son prácticamente inexistentes; y los indirectos, difícilmente cuantificables.

El desarrollador y sus motivaciones

¿Quienes son los desarrolladores?

- Personas jóvenes (edad media: 27 años).
- Edades de incorporación al Software Libre: 21-23 años.
- El 70% de los desarrolladores cuentan con educación universitaria.
- En el 30% restante muchos no son universitarios porque están aún en edad escolar.
- Porcentaje de mujeres: 1% - 3%
- Tienen pareja: 60%
- Tienen hijos: 16%

El desarrollador y sus motivaciones

¿Qué hacen los desarrolladores?

- Ingenieros software: 33%
- Estudiantes: 21%
- Programadores: 11%
- Consultores: 10%
- Profesores de Universidad: 7%

- 20% no pertenece al campo de las Tecnologías de la Información

El desarrollador y sus motivaciones

Distribución geográfica (Debian GNU/Linux)

- Estados Unidos: 297
- Alemania: 136
- Reino Unido: 75
- Australia: 52
- Francia: 51
- Canadá: 49
- España: 34
- Japón: 33
- Italia: 31

El desarrollador y sus motivaciones

Dedicación

- Media: 11 horas semanales

- Menos de dos horas: 22,5%
- De dos a cinco horas: 26,1%
- De seis a diez horas: 21%
- De once a veinte horas: 14,1%
- De veinte a cuarenta horas: 9,2 %
- Más de cuarenta horas: 7,1%

El desarrollador y sus motivaciones

Motivaciones

- Aprender y desarrollar nuevas habilidades: 80%
- Compartir conocimientos y habilidades: 50%
- Participar en una nueva forma de cooperación: 33%
- Por reputación: 9%
- Por beneficio económico: 5%

Financiación del Software Libre

Cada proyecto tiene su propia forma de financiación, desde el que es realizado de forma completamente altruista con recursos de los desarrolladores, hasta el que es llevado a cabo por una empresa que factura el 100% del coste, o el que es llevado por una entidad interesada.

En este apartado nos vamos a centrar en los proyectos donde hay financiación externa.

Financiación del Software Libre

Financiación pública

Puede ser a través de un gobierno o una institución pública.

Normalmente proviene de fondos de I+D.

La entidad que financia normalmente no tiene la intención de recuperar la inversión (al menos no de forma directa).

Algunas de las motivaciones son: investigación científica, promoción de estándares o ayuda a la sociedad.

Financiación del Software Libre

Financiación privada sin ánimo de lucro

Es muy parecida a la financiación pública.

Es promovida habitualmente por fundaciones y organizaciones no gubernamentales.

La motivación suele ser crear Software Libre para su uso en algún ámbito que la entidad financiadora considera especialmente relevante.

Un caso claro es la Free Software Foundation (FSF).

Financiación del Software Libre

Financiación por quien necesita mejoras

Tiene lugar cuando se necesitan mejoras sobre un producto.

Pueden ser tanto mejoras, como correcciones de errores.

Normalmente la mejora resultante de este proceso también es Software Libre.

Financiación del Software Libre

Financiación indirecta

No es necesario que sea la misma empresa que desarrolla la aplicación la que obtenga un beneficio económico a través de esta vía.

Se puede conseguir a través de: Libros, Hardware, CDs con programas, etc.

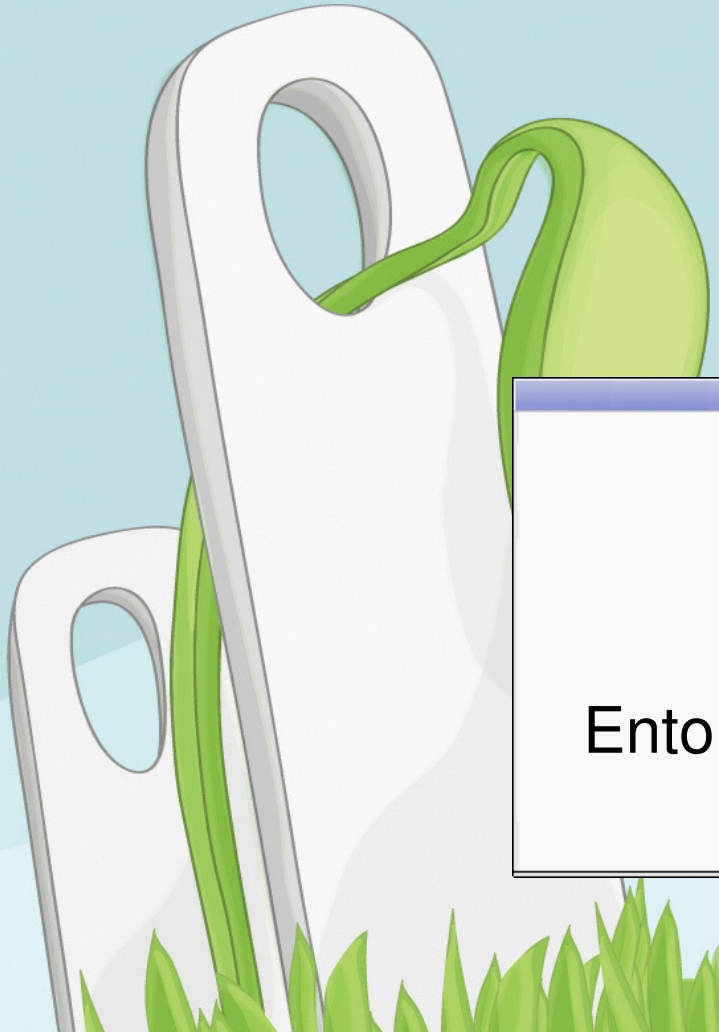
Financiación del Software Libre

Financiación como inversión interna

Financiar los inicios de una aplicación que es útil para una organización, de forma que se asegure que el proyecto pueda ser mantenido posteriormente por la comunidad.

Una vez el proyecto está siendo mantenido completamente por la comunidad, se pueden retirar muchos de los recursos.

Descanso para preguntas



A continuación:

Entornos y tecnologías de desarrollo

Entornos y tecnologías de desarrollo

¿Qué herramientas hay disponibles para el desarrollador?

¿Cómo podemos aprovechar las colaboraciones a nuestro proyecto?

¿Cómo sacar partido a la comunidad de Software Libre?

Lenguajes y herramientas

La mayoría del Software Libre está escrito en C, debido a la gran cantidad de desarrolladores que lo conocen.

También existe soporte para otros lenguajes como C++, C# y Java.

También son populares los lenguajes interpretados como Perl, Python, PHP, Ruby, etc...

Lenguajes y herramientas

Las herramientas básicas para desarrollar Software se basan en un editor de textos (Vim, Emacs) y un compilador (GCC).

En el caso de proyectos que poseen una gran cantidad de ficheros fuente, el uso de una herramienta de compilación con gestión de dependencias (make) también se hace necesaria.

Finalmente, para proyectos distribuidos, el uso de herramientas para gestión de múltiples versiones (cvs, subversion), es igualmente imprescindible.

Mecanismos de colaboración

El auge del Software Libre comenzó a la vez que Internet, aunque anteriormente todo dependía de cintas magnéticas enviadas por correo. Internet ha posibilitado una comunicación fluida a los desarrolladores.

Actualmente, la colaboración se basa en herramientas que hacen uso extensivo de Internet (wikis, IRC, listas de correo, cvs, etc).

Gestión de fuentes

Es necesario mantener una historia de los ficheros en el desarrollo de un proyecto porque pueden introducirse errores que no se detectan hasta versiones posteriores.

Igualmente, es necesario tener una lista de los cambios realizados, para poder registrar al autor de cada cambio, y las razones de éste.

Es igualmente importante, cuando existen varias ramas de un proyecto, y hay que corregir errores que afectan a ambas.

Gestión de fuentes

Las dos herramientas de gestión de fuentes más ampliamente usadas son cvs y subversion, siendo ésta última la que está experimentando un mayor auge hoy en día.

La razón de ello radica en que cvs es una herramienta diseñada a principios de los años 80, y carece de muchas de las funcionalidades que subversion incorpora.

La ventaja de CVS, es que está más documentada, y existen más personas con experiencia usándolo.

Documentación

Uno de los principales atractivos para el desarrollador de Software Libre es la gran cantidad de información técnica disponible.

Ésta suele ser escrita habitualmente por el propio desarrollador de la aplicación, lo que hace que sea fidedigna.

Gestión de errores

Otro de los puntos fuertes del Software Libre es la ayuda que suponen los informes de errores que los usuarios envían a los desarrolladores.

Para facilitar esta tarea, existe una serie de aplicaciones sobre todo basadas en la web, que han demostrado con creces su utilidad a lo largo de los años.

Igualmente, a través de este medio, se pueden realizar sugerencias a los desarrolladores.

Gestión de flujo de trabajos

Dada la gran cantidad de colaboradores anónimos y puntuales en los proyectos de Software Libre, no se hace un uso extensivo de estas herramientas en los proyectos.

No obstante, se suelen definir tareas a completar, y se espera a que alguna persona se haga cargo de ella.

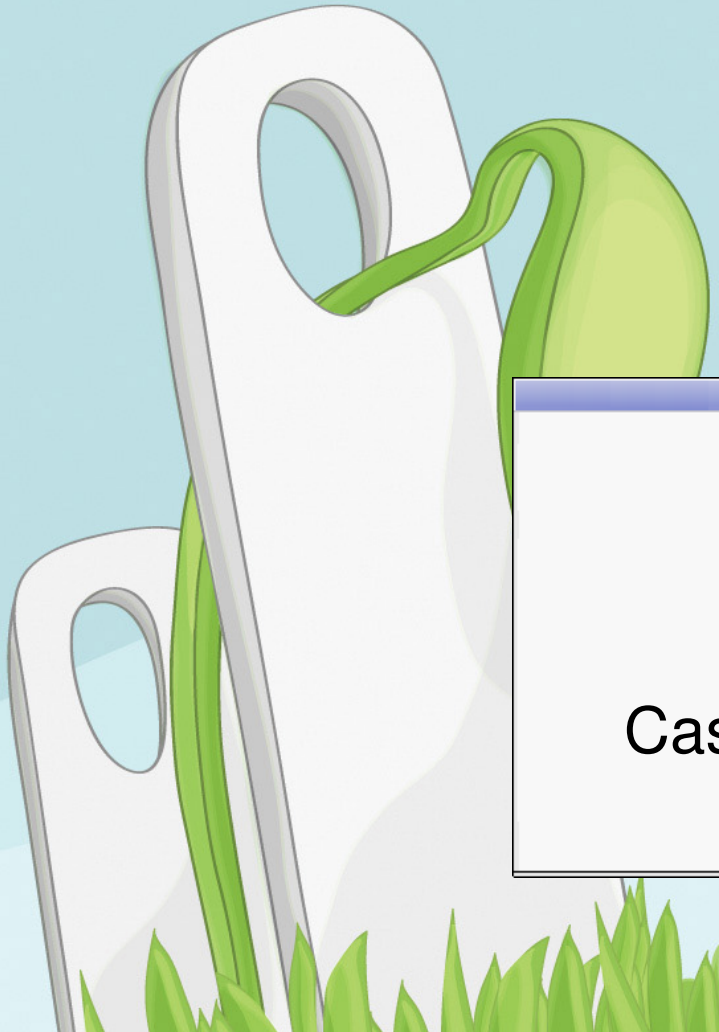
Aunque no brilla por su gran sofisticación, este sistema ha proporcionado buenos resultados a lo largo de los años.

Sitios de soportes al desarrollo

Si necesitamos albergar la página web de nuestro proyecto, foros, listas de correo, servidor de noticias, sistemas de gestión de errores, gestores de flujo de trabajos, un sistema de control de versiones, un sitio donde proporcionar los programas precompilados a los usuarios, servicio de documentación, copia de seguridad para prevención de desastres, etc. disponemos de diferentes sitios web que lo proporcionan gratuitamente.

Entre todos ellos destaca SourceForge (www.sourceforge.net).

Descanso para preguntas



A continuación:

Casos de éxito del Software Libre

Casos de éxito del Software Libre

¿Cuáles son los proyectos del Software Libre con mayor éxito?

GNU/Linux

El núcleo (kernel) Linux es una estrella del Software Libre. Tanto lo es, que el conjunto del núcleo, las librerías y las aplicaciones tienen como nombre Linux.

Asimismo, suele confundirse el Software Libre con Linux, no siendo esto cierto, porque hay Software Libre que no funciona en Linux, y Software que funciona en Linux que no es Software Libre.

GNU/Linux

El modo de trabajo en Linux

El desarrollo se basaba en enviar código fuente con parches y nuevas funcionalidades a listas de correo. Una vez se aceptan por los responsables de cada una de las áreas del núcleo, se integran dentro de éste. Luego se pasó a usar BitKeeper.

Existe un desarrollo paralelo de dos ramas del núcleo: una estable (par) y una inestable (impar). No hay plazos de entrega.

Resultado: gran estabilidad y se tarda poco tiempo en corregir errores (a veces es cuestión de minutos).

GNU/Linux

Cuando se encuentra un error en el núcleo

El núcleo de Linux cuenta con miles de personas/mes dedicadas a su desarrollo, lo que hace que sea evidentemente un éxito.

Con estas cifras, es bastante probable encontrar errores en el núcleo, y solucionarlos o contactar rápidamente con alguien que pueda solucionarlo.

Realmente, este procedimiento es muy costoso en recursos, pues pueden haber docenas de parches para un mismo problema y sólo uno es aceptado; los demás se desechan.

GNU/Linux

Algunos datos (para versión 2.4.21 del 16.6.2003)

- Más de 3.250.000 líneas de código fuente
- Estimación de coste: 150 millones € (25.000 millones Ptas)
- Número medio de desarrolladores: 132
- Claro predominio de C como lenguaje (95% de las líneas totales de código fuente)
- Usa ensamblador en las partes que requieren un mayor rendimiento (5% de las líneas totales de código fuente)

FreeBSD

Se tiene como objetivo la creación de un sistema operativo que pueda ser utilizado sin ningún tipo de obligaciones ni ataduras, pero con todas las ventajas de la disponibilidad del código, y de un cuidadoso proceso que garantice la calidad del producto.

El usuario puede hacer lo que quiera con el sistema operativo: licencia BSD.

FreeBSD

Algunos datos (para versión 4.8)

- Más de 1.500.000 líneas de código fuente
- Estimación de coste: 400 millones € (66.000 millones Ptas)
- Número medio de desarrolladores: 235
- Claro predominio de C como lenguaje (95%)
- Otros lenguajes usados: shell (2,7%), C++ (1,7%), ensamblador (1,5%), Perl (1,2%), Yacc (0,75)

KDE

KDE ~ K Desktop Environment (version 3.1.3)

Entorno de escritorio amigable, abierto, estable, confiable y poderoso

- Más de 6.100.000 líneas de código fuente
- Estimación de coste: 300 millones € (50.000 millones Ptas)
- Número medio de desarrolladores: 200
- Traducido a más de 50 lenguas (incluido esperanto)
- Predominio de C++ (82%)

GNOME

GNOME ~ GNU Network Object Model Environment

Entorno de escritorio para el usuario final completo, libre y fácil de usar. También es una plataforma potente de desarrollo.

- Más de 9.200.000 líneas de código fuente
- Estimación de coste: 350 millones € (58.700 millones Ptas)
- Número medio de desarrolladores: 250
- Predominio de C (86,1%), seguido de C++ (6,27%)

Apache

El servidor web de mayor implantación.

- Cuota de uso (1999): 57%
- Cuota de uso (2003): 68%
- Más de 85.000 líneas de código fuente
- Estimación de coste: 3,5 millones € (580 millones de Ptas)
- Número medio de desarrolladores: 200
- Predominio de C (92%), seguido de shell (7%)

Firefox

Navegador web

- Más de 3.500.000 líneas de código fuente
- Estimación de coste: 120 millones € (20.000 millones de Ptas)
- Número medio de desarrolladores: 140
- Predominio de C++ (54%), seguido de C (33%) y Java (7%)

OpenOffice.org

Suite de Ofimática (procesador de textos, hoja de cálculo, presentaciones, dibujo, formulas matemáticas, editor HTML, base de datos)

- Soporte casi completo de documentos de Microsoft Office
- Exportación a PDF
- Más de 4.000.000 líneas de código fuente
- Estimación de coste: 140 millones € (23.400 millones de Ptas)
- Número medio de desarrolladores: 150
- Predominio de C++ (85%) seguido de Java (6%)

Debian GNU/Linux

La distribución de Linux más grande en la actualidad

- Más de 105.000.000 líneas de código fuente
- Estimación de coste: 3.000 millones €
  (500.000 millones de Ptas)
- Número medio de desarrolladores: 3950
- Predominio de C (63%) seguido de C++ (12%)

Otras aplicaciones

XMMS: Reproductor de audio multiplataforma

Mplayer: Reproductor y codificador multimedia

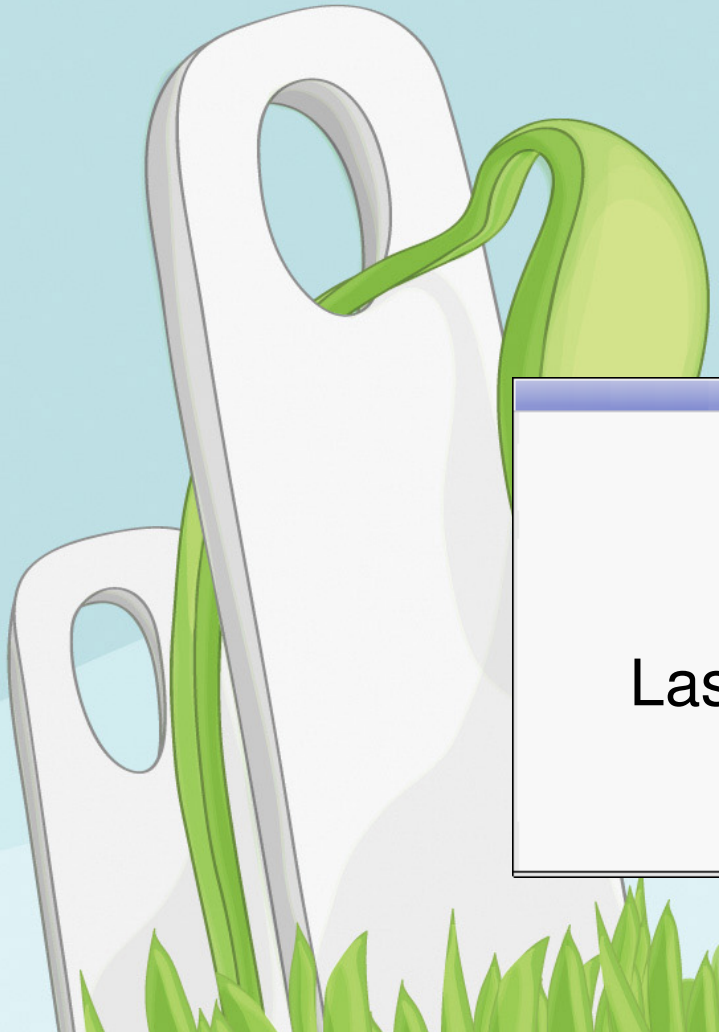
The GIMP: Programa de creación y manipulación de imágenes

Evolution: Suite de trabajo en grupo (e-mail, calendario, etc.)

Gaim: Cliente de mensajería instantánea multiprotocolo

Synaptic: Gestión de paquetes de software

Descanso para preguntas



A continuación:

Las relaciones del Software Libre

El software cerrado

Las relaciones del Software Libre

¿Cuáles son las relaciones del Software Libre con las empresas? ¿Y con los gobiernos?

El Software Libre y las empresas

Todas las empresas de Software mantienen relaciones estrechas con el Software Libre, ya sea de aceptación o de rechazo.

IBM, una de las mayores empresas tecnológicas, es un gran aliado del movimiento del Software Libre, y oferta productos, soluciones, consultoría, etc. para aplicaciones de Software Libre.

Asimismo, también existen multitud de empresas por todo el mundo que basan la totalidad, o la parte de sus proyectos clave en Software Libre.

El Software Libre y los gobiernos

Existen múltiples iniciativas en los gobiernos para usar Software Libre dado que se adapta mejor a sus necesidades y requerimientos. Algunas a nivel español son:

- Linex : Distribución GNU/Linux para Extremadura
- Guadalinex : Distribución GNU/Linux para Andalucía

Llurex (Valencia), Madux (Madrid), etc.

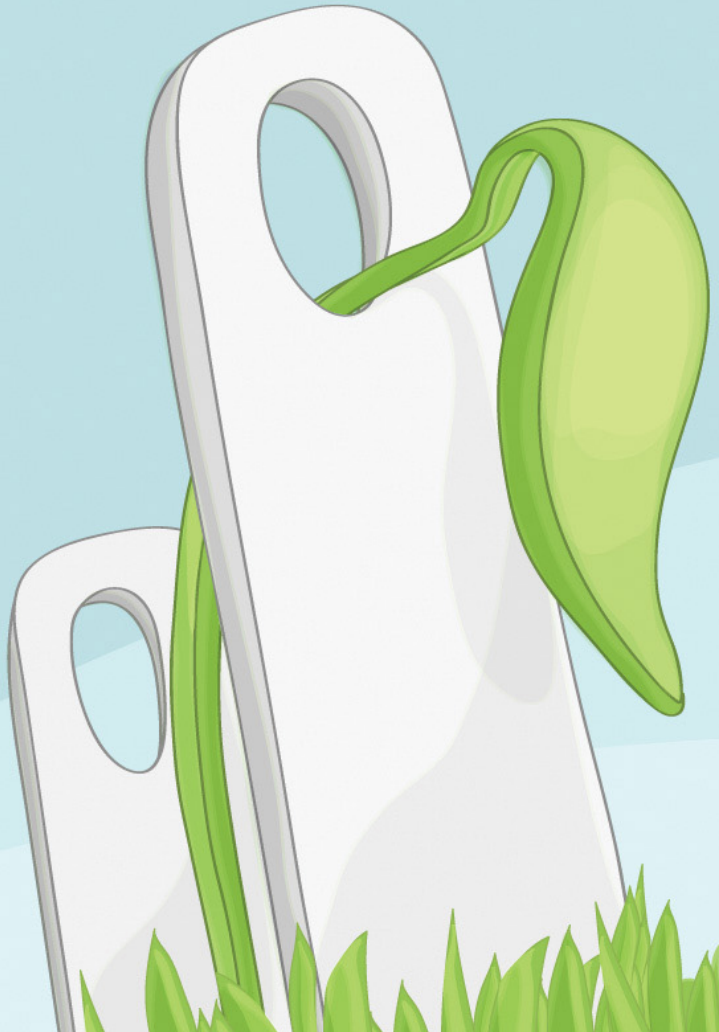
Software cerrado

¿Puede convivir con el Software Libre?

Los problemas del Software cerrado

- Mayor coste de adquisición
- Dificultad de adaptación a las necesidades personales
- Generalmente es imposible probado antes de comprarlo (excepto Freeware, Shareware, etc)
- Imposibilidad de auditar el código fuente para verificar seguridad, etc.
- Dependencia de una sola empresa

Gracias por la atención prestada



Ponente

Juan José del Río
juanjose@simpleoption.com

Juan Miguel Taboada
juanmi@centrologic.com