

El Proceso de Liberación

en el Marco Legal del Código Abierto

Juanjo Amor

jjamor@opensistemas.com
OpenSistemas

15 Abril 2011



Open

(cc) 2011 Juanjo Amor

(cc) 2008 José Gato, Teófilo Romera

(cc) 2007 Juanjo Amor, Gregorio Robles, Jesús M. González-Barahona

Some rights reserved. This work licensed under Creative Commons Attribution-ShareAlike License. To view a copy of full license, see <http://creativecommons.org/licenses/by-sa/2.0/> or write to Creative Commons, 559 Nathan

Abbott Way, Stanford, California 94305, USA.

¿Por qué?

¿Por qué liberar software?

- La comunidad ayuda: contribuye a mejorar la aplicación.
 - Corrección de errores.
 - Mejoras.
 - Internacionalización ...
- Ayudamos a la comunidad: aportamos una aplicación que puede ser utilizada, o adaptada para cubrir otras necesidades, por la comunidad.
- Visibilidad y prestigio: la empresa consigue mucha visibilidad al distribuir el software sin costes.
- ...

Aspectos legales...

Cuando ya tenemos un programa hecho... hay que tomar decisiones *legales*.

- Pensarse bien la licencia, también para la documentación.
¿Queremos que el software se pueda mejorar y siga siendo libre?
- Decidir quién posee el copyright (personas, empresa...)
- Incluir textos de copyright y mención de licencia en todos los ficheros
- Distribuir junto con la licencia *completa*.
- A partir de aquí, *seguir la licencia* como si no fuera nuestro.

Licencias

¿Qué es una licencia?

- Las obras intelectuales pertenecen a su dueño y por las leyes de *copyright* no otorgan ningún derecho a otros usuarios.
- La licencia es un contrato por el que se ceden algunos derechos a los usuarios, por ejemplo el derecho a usar el programa.

Ejemplos de licencias:

- Los “EULA” (End User License Agreement) que acompañan a muchos programas privativos.
- Las licencias libres.

Licencias Libres

La mayoría de las licencias de software están diseñadas para evitar que copiemos o modifiquemos los programas que compramos. En cambio, las licencias libres se diseñan para garantizar las **cuatro libertades** del software:

- Libertad 0. Libertad de usar el programa, con cualquier propósito.
- Libertad 1. Libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
- Libertad 2. Libertad de redistribuir copias del programa, con lo cual ayudas a tu prójimo.
- Libertad 3. Libertad de distribuir las copias modificadas de tu programa, con lo que permites que tu prójimo se beneficie de las mejoras que haces.

La libertad 1, implica que debe ser posible acceder al **código fuente** del programa.

Licencias copyleft

- Las licencias libres pueden ser *permisivas* o *restrictivas*.
- En las *restrictivas* el usuario está obligado a mantener la misma licencia cuando distribuya copias modificadas del programa.
- Todas otorgan al usuario las cuatro libertades.
- Las licencias restrictivas también se conocen como licencias **copyleft**.

¿Qué licencia escoger?

- Principalmente, una licencia libre.
- Debemos escoger entre permisiva y *copyleft*.
- ... y por último, leer su *letra pequeña*, detalles que nos resultarán importantes a la hora de decidir.

Por lo general, debemos:

- Intentar reutilizar una licencia existente.
- Solo si ninguna nos satisface, crear una licencia nueva.
- ... pero el proceso será más costoso.

Ejemplo: Licencia MIT/X11

Licencia *permissiva*, corta y simple, con tres secciones claras:

- Derecho a usar sin restricciones, modificar, publicar cambios... es decir, las 4 libertades. Además, permite *sublicenciar*, es decir, cambiar la licencia en software derivado.
- Cláusula de obligación de incluir la licencia en todos los ficheros.
- Cláusula de exención de responsabilidad.

¿Por qué elegiríamos esta licencia?

Ejemplo: Licencia GPL

Licencia *copyleft*, la más utilizada, considerada *pata negra* y debe ser nuestra favorita salvo que no queramos *copyleft*.

¿Por qué elegirla?

- Garantiza las cuatro libertades y que cualquier trabajo derivado seguirá siendo software libre.
- Es la más popular y ha sido probada jurídicamente en varios casos.
- Existe un estudio amplio de compatibilidad de esta licencia con otras licencias libres. Esto nos permite saber más fácilmente si podemos integrar código bajo otras licencias con nuestra aplicación GPL.

Ejemplo: Licencia GPL

Actualmente la licencia tiene dos versiones ampliamente usadas: la GPL2 y la GPL3. Frente a la GPL2, la GPL3:

- Mejora compatibilidad con otras licencias libres.
- Establece condiciones mínimas para evitar demandas judiciales a quienes modifiquen programas protegidos por patentes de software.
- Lo más importante: prohíbe el uso del código bajo GPL3 en aplicaciones con DRM que impongan restricciones a los usuarios.

¿Más licencias?

¿Dónde encontrar licencias libres?

- <http://www.gnu.org/licenses/>
- <http://www.opensource.org/>

open
systems

Buenas prácticas...

- Subir el software a un sitio donde se pueda descargar (con sus fuentes)
- Anunciar el software (en sitios como Freshmeat o similar)
- Simplificar el proceso de compilación e instalación.
Empaquetar para las distribuciones más populares.
- Documentar la distribución.

open
s i s t e m a s

Buenas prácticas (II)

- Limpieza del código
- Documentación (en inglés)
- Hacer el software traducible (*internacionalización*)
- Abrir canales de *feedback* (listas de correo, *bugtrackers*, foros)
- Infraestructura para el trabajo colaborativo, que sea fácil de usar
- Gestionar contribuciones, parches, modificaciones y sugerencias de terceros

Un sitio de referencia

¿Qué buscamos?

- Visibilidad (que el usuario sepa que existimos)
- Base de usuarios (gente que use el programa)
- Colaboradores (facilitar la ayuda a usuarios)
- Co-desarrolladores (gente que contribuya a futuras versiones)
- Promover y soportar un modelo de negocio
- ...

Según lo que busquemos, la infraestructura requerida será diferente

Dar una buena imagen

- La primera impresión: el sitio web
- La segunda impresión: el proceso de instalación
- La tercera impresión: las *demos*
- No es mala idea tener material en el sitio web como: artículos, *screencasts*, etc.

Fundamental: dar una buena impresión al visitante

El día a día

Si queremos que el sitio se mantenga como “la referencia”:

- Los usuarios deben estar motivados para usar los recursos del sitio
- La información debe estar actualizada
- Debe haber información sobre aplicaciones relacionadas
- Documentación, buena y abundante
- Herramientas de comunicación con usuarios: que puedan dar su *feedback*

Organización del sitio web

- Presentación del proyecto
- Página de descarga (binarios and fuentes)
- Página de novedades (en formato “blog”)
- Documentación (on-line y descargable)
- Ejemplos de uso, trucos. . .
- Espacio para la participación (informe de fallos, parches, etc).
- Utilidades relacionadas.
- Información de soporte.

Utilidades relacionadas

- Repositorio de software (ftp, http)
- CVS / Subversion, CVS-Web, Bonsai: si queremos facilitar el acceso a *lo último de lo último*
- Gnats, BugZilla: gestión de partes (fallos, sugerencias de mejora, etc.)
- CMS (Drupal, etc): ayudan a crear comunidad
- Blogs de desarrollador
- Mailman: listas de correo con archivos públicos en algunos casos (muy importante)
- SourceForge, Savannah, BerliOS: todo lo anterior, unido en una **Forja**

Para el software

Empaquetado:

- tar.gz
- paquete Debian
- paquete RPM
- Sources

Objetivo: Que sea lo más fácil posible instalarlo.

Cómo distribuir el ejecutable

- Licencia (COPYING)
- Información general (README). Muy importante: citar la web y la documentación.
- Información de instalación (INSTALL)
- Documentación (puede que en otro paquete)
- Binario para una plataforma concreta (un binario estático puede hacerlo más utilizable en diversas distribuciones del sistema operativo)
- Paquetes con su número de versión *claramente* indicado

“Marketing”

- Congresos, etc.
- Sitios web relacionados (específicamente, de noticias)
- Grupos de noticias
- Listas de correo
- Redes sociales
- ¿Community management?

Importante: crear noticias frecuentes, pero evitar el spam!

Muy importante: Estar preparados para cuando se haga el anuncio.

Recursos humanos

Además de programadores, necesitamos:

- Gestión de infraestructura (100 / 20)
- Coordinador de desarrollo, incluyendo QA (40-200 / 20)
- Atención al gestor de bugs (?? / 10-15)
- Mantenimiento de documentación e información on-line (60 / 10)
- Relaciones públicas (?? / 20)

(horas al momento de liberar / horas por semana)

Fuente: "Open Source as a Business Strategy", Brian Behlendorf

Conclusiones

Como conclusiones:

- Muy importante: atraer usuarios al sitio (“campana de marketing”)
- Muy importante: hacer que los usuarios se queden (“ventaja competitiva”)
- Objetivo principal: crear comunidad
- Facilitar la vida a los usuarios
- Muy importante: automatizar todo lo automatizable ...
- Por supuesto... depende de dónde queramos llegar

Referencias

- “Open Source as a Bussiness Strategy”, por Brian Behlendorf (en Open Sources, Voices from the Open Source Revolution)
<http://www.oreilly.com/catalog/opensources>
- “Whether and How To Publish Software Under an Open Source License”, por Sebastien Blondeel
<http://publish.idealx.org/>
- “Aprende a Liberar”, Wiki de Mancomun.org
http://wiki.mancomun.org/index.php/Aprende_a_liberar
- “Guía Básica del Software de Fuentes Abiertas”, CENATIC
http://www.cenatic.es/phocadownload/guiabasica_sfa.pdf

El Proceso de Liberación

en el Marco Legal del Código Abierto

Juanjo Amor

jjamor@opensistemas.com
OpenSistemas

15 Abril 2011

