

---

# ¿Cómo se desarrolla el software libre?

*Jesús M. González Barahona*  
*jgb@debian.org jgb@gsync.escet.urjc.es*



*Jornada Didàctica sobre Programari Lliure*  
*Algemesí, 27 de marzo de 2006*

---

©2006 Jesús M. González Barahona.

Algunos derechos reservados. Esta presentación se distribuye bajo la  
licencia “Reconocimiento-CompartirIgual 2.5 España” de Creative  
Commons, disponible en

<http://creativecommons.org/licenses/by-sa/2.5/es/deed.es>

Este documento (o uno muy similar) está disponible en

<http://curso-sobre.berlios.de>

## ¿Qué es software libre?

Quien lo recibe puede (pero no está obligado a):

- usarlo como mejor le parezca, donde mejor le parezca.
- redistribuirlo a quien quiera, por los medios que quiera.
- modificarlo (y mejorarlo o adaptarlo).
- redistribuir las modificaciones

Imprescindible: disponibilidad de código fuente.

**software libre no es lo mismo que  
software gratis**

<http://www.fsf.org/philosophy/free-sw.es.html>

<http://opensource.org/docs/osd-spanish.html>

## ¿Y por qué es esto y no otra cosa?

Desde luego no es casualidad...:

- Motivos éticos: porque las cosas deberían ser así.
- Motivos prácticos: porque las cosas funcionan mejor así.

Largas discusiones, que han asentado cierto consenso:

- Debian Free Software Guidelines,
- Definición de “Open Source”.
- Definición de software libre de la FSF

## La ética del programador

- Un buen programador debería contribuir con su trabajo a la Comunidad.
- Un buen programador debería poder aprovechar el trabajo de otros buenos programadores.
- Un buen programador debería poder “arreglar” y mejorar cualquier programa.
- Un buen programador se siente orgulloso de usar su código, y de que otros lo usen.

Buen programador: hacker

Ideas formuladas por Richard Stallman, continuadas por la FSF, la comunidad BSD, y otros.

## ¿Y los argumentos prácticos?

- Nuevos modelos de desarrollo (bazar frente a catedral).
- Ventajas del escrutinio público y de la mejorabilidad.
- Competencia real en el desarrollo y el mantenimiento.
- Viabilidad técnica frente a mercadotecnia.
- Nuevas posibilidades de negocio (ej: desafío a posiciones de monopolio).

Ideas formuladas por Eric Raimond, promovidas por la Open Source Initiative y otros.

## ¿Quién desarrolla software libre?

- Voluntarios, en su tiempo libre
- Voluntarios, en tiempo pagado por su empresa
- Contratados para desarrollar software libre
- Emprendedores buscando un modelo de negocio

## ¿Cómo empiezan los proyectos?

- Solución a problemas individuales
- Evolución o especialización de proyectos anteriores
- Estrategia de empresas
- Promoción de administraciones públicas o fundaciones
- Emprendedores buscando modelos de negocio
- Liberación de software anteriormente propietario

## La importancia de la Comunidad

- Núcleo de un proyecto: sus desarrolladores clave
- Grupo extendido de desarrolladores: colaboradores habituales (parches, funcionalidades específicas, etc.)
- Colaboradores casuales (parches, informes de error, colaboración en soporte)
- Usuarios activos (listas de correo, informes de error)
- Usuarios “normales”

## Forma de trabajo habitual

- Desarrolladores repartidos por todo el mundo, con distintas situaciones personales
- Interacción sólo telemática, y en general asíncrona
- Importancia de la meritocracia y la calidad del trabajo realizado
- Alto nivel de apertura y trazabilidad
- Importancia de la coordinación (habilidades sociales)

## Las herramientas son muy importantes

- Coordinación: listas de correo, chats (IRC, Jabber)
- Código: control de versiones (CVS, SVN, Arch)
- Informes de error (Bugzilla, GNATS)
- Documentación: wikis, sitios web
- Sitios de alojamiento integrado: SourceForge, Berlios, Savannah, Allioth (software: GForge)

## ¿Cómo se consigue la calidad?

- Libera amenudo, libera pronto
- Con suficientes ojos, cualquier errata termina apareciendo
- Revisión de los parches por desarrolladores experimentados
- Debate en abierto de las decisiones técnicas
- Refuerzo de la motivación para informar y corregir errores
- Medidas “tradicionales” (juegos de pruebas y regresión para cada módulo)

## A modo de resumen...

- El desarrollo de software libre es diferente...
- ...a veces muy diferente
- Pero cada vez es más conocido (y respetado)
- Es fácil iniciar un proyecto
- La comunidad es muy importante, y no siempre fácil de conseguir

## Algunas URLs

- Introducción al software libre (libro libre):  
<http://curso-sobre.berlios.de/introsobre>
- Sobre software libre (libro libre):  
<http://gsyc.escet.urjc.es/~grex/sobre-libre/>
- Grupo de trabajo de la Comisión Europea sobre software libre:  
<http://eu.conecta.it>
- Open Sources (O'Reilly)  
<http://www.oreilly.com/catalog/opensources/>
- Articulillos y presentaciones sobre este y otros temas  
<http://sinetgy.org/jgb/>