Desarrollar software libre en una empresa

Amadeu Albós Raya

PID_00145047

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

En este módulo nos adentramos en el mundo de la producción de software libre y en sus particularidades más relevantes para el producto, la empresa y la comunidad de usuarios.

En un primer momento, analizaremos el desarrollo de software libre bajo el punto de vista del proyecto, considerando los principales aspectos ligados a la población y la gestión del proyecto, así como la participación de la comunidad de usuarios en diversidad de aspectos.

Mediante el proyecto de software libre se formaliza la relación entre empresa y comunidad de usuarios. La adecuación de las particularidades de dicha relación es fundamental para la consecución de los objetivos del proyecto.

A continuación, describiremos las particularidades de la comunidad de usuarios del software libre y su gestión por parte de la empresa. Esta gestión complementa la metodología de producción e implementa la estrategia relacional comentada anteriormente.

Finalmente, el módulo concluirá con la presentación de un caso de estudio de una empresa real que produce software libre.

Este módulo se organiza como una guía de lecturas externas, cuyo objetivo es profundizar en las particularidades de los diferentes aspectos que se presentan y que son relevantes para la producción empresarial de software libre.

Objetivos

Los objetivos que se deben alcanzar al finalizar este módulo son los siguientes:

1. Familiarizarse con la metodología de producción de software libre.

2. Comprender la importancia que tiene la comunidad de usuarios en el desarrollo de productos basados en software libre.

3. Identificar y analizar los factores relevantes que afectan al éxito de la producción de software libre.

4. Entender la importancia de formalizar una metodología que permita complementar los esfuerzos de la empresa y de la comunidad de usuarios.

5. Profundizar en las implicaciones directas e indirectas de llevar a cabo un proyecto de desarrollo basado en software libre.

1. La producción de software libre

En este primer apartado nos centraremos en la producción de software libre desde el punto de vista de su desarrollo o elaboración, es decir, con independencia de eventuales modelos de negocio que lo exploten de forma lucrativa.

En diversas asignaturas del máster, especialmente en las asignaturas dedicadas a la producción del software[1], se analiza con detalle el proceso tecnológico que caracteriza la elaboración de software libre.

(1)Introducción al desarrollo de software, Ingeniería del software en entornos de software libre, o Conceptos avanzados de desarrollo de software.

Este proceso tecnológico es complementario a las metodologías que nos permiten formalizar un proyecto cooperativo viable y duradero a lo largo del tiempo. En este sentido, la colaboración de la comunidad de usuarios en el proyecto de software libre es fundamental para conseguir una masa crítica de usuarios que permita la viabilidad del proyecto.

En consecuencia, buena parte de estas metodologías y actuaciones van encaminadas a ofrecer soporte y garantías a las relaciones entre el proyecto y la comunidad de usuarios. Para darnos cuenta de la importancia de esta relación, basta con consultar los recursos que ofrecen a la comunidad de usuarios los proyectos de software libre más populares.

Proyectos populares

Por ejemplo, OpenOffice.org (http://contributing.openoffice.org/) y Mozilla (http://www.mozilla.org/contribute/).

Para desarrollar estos conceptos, en los próximos apartados describiremos tres puntos de vista complementarios. En un primer momento, consideraremos algunas ideas básicas sobre la producción de software libre. A continuación, detallaremos de forma breve los principales pasos para poner en marcha un proyecto basado en software libre. Por último, profundizaremos en los principales aspectos de gestión del proyecto de software libre.

1.1. La producción de software libre

La producción de software libre, del mismo modo que la producción de cualquier software, responde a la necesidad de solucionar una problemática tecnológica concreta[2].

(2)Por ejemplo, añadir funcionalidades a una aplicación o solucionar errores de funcionamiento.

Si bien el proceso tecnológico de refinar y hacer evolucionar una aplicación de software libre puede presentar muchas similitudes comparado con una aplicación basada en software propietario, la diferenciación que supone la aper-

tura del modelo le confiere un funcionamiento particular. Es decir, el carácter abierto y cooperativo de su producción incide en la estructura de evolución cuantitativa y cualitativa a lo largo de las versiones.

Muchos son los autores que han escrito sobre las particularidades de producir software libre. Puesto que este módulo no tiene como objetivo detallar o describir exhaustivamente dichas particularidades, dado que se tratan de forma exhaustiva en otras asignaturas, nos centraremos en remarcar algunas de las más interesantes para el caso que nos ocupa.

Para ello, consideraremos algunos de los conceptos presentes en el ensayo *La catedral y el bazar*, de Eric S. Raymond, donde se analizan las particularidades del software libre y, en especial, del caso GNU/Linux.

Web recomendada

E. Raymod (1997). *The cathedral and the bazaar* (http://www.catb.org/~esr/writings/cathedral-bazaar/).

- **El origen de la producción**
  A grandes rasgos, la producción del software libre nace a partir de las necesidades particulares del usuario o desarrollador en su actividad habitual. Es decir, la colaboración en el desarrollo del software se empieza buscando y hallando un problema cuya resolución nos resulte interesante, relevante o necesaria.

Inicios de la producción

Buena parte de los fundamentos del software libre se basa en la publicación de los ajustes o desarrollos concretos que realizan trabajadores en el ejercicio de sus funciones laborales cotidianas.

- **La comunidad de usuarios**
  La comunidad de usuarios de software libre, que agrega tanto usuarios finales como desarrolladores y programadores, es la base que da sentido a la definición de desarrollo de software libre.
  Tratar a los usuarios como colaboradores en el proyecto de producción es la forma más fácil de depurar y mejorar el código de forma rápida (si la base de colaboradores es suficiente).
  Por ende, los colaboradores representan uno de los recursos más valiosos para el desarrollo de la aplicación, por lo que también resulta valioso reconocer las buenas ideas y las soluciones que aportan.

- **Las versiones de la aplicación**
  Una de las características de la producción de software libre es la reutilización y reescritura del código original para derivarlo en un código nuevo, ya sea libre de errores, con nuevas funcionalidades o con mejor rendimiento (entre otros aspectos).
  Por otra parte, en los proyectos de desarrollo de software libre se promueve la liberación de código de forma rápida y frecuente, de forma que la actividad del proyecto sea dinámica e incesante.

- **La coordinación de la producción**
  La persona –o personas– que coordina el proyecto tiene que saber gestionar el potencial global de la comunidad de usuarios, dirigiendo la evolución del proyecto sin coerción y aprovechando los medios y sinergias que ofrece una red como Internet.

La herencia del código de la aplicación y de su gestión de coordinación es importante para el futuro del proyecto de desarrollo de software libre. La elección del sucesor que controle y gestione la producción no se debe dejar al azar.

1.2. El proyecto de software libre

Complementando las consideraciones tecnológicas y funcionales de las aplicaciones basadas en software libre, uno de los objetivos primordiales de todo proyecto de software libre es la difusión de la aplicación o la obtención de una masa crítica de usuarios.

Es decir, es poco útil para el futuro del proyecto que el código generado, a pesar de resolver problemas o carencias concretas, no sea conocido y aplicado por los usuarios potenciales. Además, constituye un objetivo necesario para su posterior mantenimiento y evolución en el tiempo. En el caso del software libre, el cumplimiento de este aspecto es fundamental para la creación de una comunidad de usuarios estable y duradera.

Se han escrito diversas guías que, en mayor o menor medida, aportan los conceptos necesarios para la creación y gestión de proyectos basados en software libre. En este apartado desarrollaremos esta cuestión mediante el artículo *Free Software Project Management HOW TO* de Benjamin Mako, que revisa las principales particularidades del proyecto desde el punto de vista práctico.

Web recomendada

B. Mako (2001). *Free Software Project Management HOW TO* (http://mako.cc/projects/howto).

Puesta en marcha

Antes de lanzar el proyecto basado en software libre, es muy importante diseñar una estructura sólida que permita soportar el posterior proceso de desarrollo con garantías suficientes.

En general, la estructura básica del proyecto debe dar respuesta a los siguientes aspectos:

- La necesidad de crear un proyecto nuevo, ya sea por ideas y objetivos propios o por la existencia de proyectos afines.
- La definición de las principales características de la aplicación (funcionalidades, licencia, numeración, etc.).
- La infraestructura básica de soporte a la difusión del nuevo proyecto y a la colaboración en su desarrollo (página web, correo de contacto, etc.).

Los desarrolladores

Una vez puesto en marcha el proyecto, nuestro segundo objetivo será la integración y consolidación de los usuarios y desarrolladores de la aplicación. Con relación a estos últimos, deberemos crear políticas y estrategias que nos permitan definir y estructurar su colaboración.

Las políticas de cooperación deben dar respuesta principalmente a dos objetivos concretos:

- la coordinación de la producción interna y externa, como la delegación de las responsabilidades y los protocolos de aceptación de las contribuciones.

- la gestión de la producción, como, por ejemplo, la estructura de ramas de desarrollo y los repositorios asociados.

Los usuarios

En los productos basados en software libre, los usuarios a menudo son también desarrolladores (y viceversa). Uno de los principales objetivos que debemos tener en cuenta son las pruebas o tests de la aplicación, ya sean funcionales, operativos, de calidad, etc.

La infraestructura de soporte

La actividad cotidiana del proyecto basado en software libre no se podría realizar sin una infraestructura de soporte adecuada a los objetivos cooperativos del mismo.

En la puesta en marcha del proyecto ya se han llevado a cabo las actuaciones básicas en este sentido, pero una vez está en funcionamiento, es necesario adecuar, mejorar y complementar los recursos existentes de acuerdo con la evolución del proyecto.

Recursos habituales

Algunos de los recursos habituales en los proyectos de software libre son: documentación, listas de correo, sistemas de control de errores y de versiones, foros, *chat*, *wiki*, etc.

La aplicación

Sin duda, el componente más relevante del proyecto y del que depende el resto de los aspectos considerados es la aplicación. Una de las características más relevantes que tiene que presentar la aplicación es que el usuario tenga suficientes garantías sobre el funcionamiento de cada una de las versiones que se lanzan al mercado.

El lanzamiento de versiones es un tema delicado que conviene meditar con calma. A grandes rasgos, tendremos en cuenta los siguientes aspectos:

- control de revisiones en términos de funcionalidades y corrección de errores (versiones alfa, beta, distribución candidata, etc.)

- cuándo lanzar la versión completa, es decir, cuándo estará preparado el código para ofrecer las garantías que esperamos y esperan los usuarios.

- cómo lanzar la versión (empaquetada, código fuente, formato binario, etc.).

Difusión del proyecto

Finalmente, y como ya hemos comentado al principio, difundir la existencia del proyecto es importante para el mismo, pero esta tarea se debe seguir considerando a lo largo del tiempo si queremos consolidar los fundamentos.

A medida que el proyecto avance, pensaremos en destacar el proyecto en listas de correo relacionadas con el software libre o en *Usenet*, incluir el proyecto en otros portales públicos (como *Freshmeat* o *Sourceforge*), o también anunciar las nuevas versiones de la aplicación en las listas de correo del mismo proyecto.

1.3. La gestión del proyecto

En este apartado profundizaremos en los aspectos de gestión del proyecto que, como promotores del mismo, deberemos tener en cuenta para dotarlo de garantías de éxito.

Los conceptos que presentamos en este apartado son complementarios a los expuestos en los apartados anteriores, puesto que posibilitan concretar y mejorar las diferentes actuaciones que se han considerado. En este sentido, es posible encontrar coincidencias directas o indirectas con estos argumentos.

Para relacionar los aspectos básicos de la gestión del proyecto basado en software libre, tendremos en cuenta las consideraciones consignadas en *Producing Open Source Software* de Karl Fogel, en especial el capítulo 5 titulado "Money".

Web recomendada

K. Fogel (2005). *Producing Open Source Sotware: How to Run a Successful Free Software Project* (capítulo 5 "Money"). (http://producingoss.com/en/money.html).

Financiación

Las particularidades de los proyectos de software libre hacen que muchas de las contribuciones estén subvencionadas informalmente (por ejemplo, cuando el trabajador de una empresa publica los ajustes que ha hecho en el código en el ejercicio de sus funciones).

Aun así, también se hacen donaciones y subvenciones que permiten obtener ingresos directos para el funcionamiento del proyecto, pero hay que tener en cuenta la gestión de estos fondos, ya que buena parte del apoyo que recibe un proyecto de software libre se basa en la credibilidad que merecen sus participantes.

Tipos de participación

Existen muchos tipos (y combinaciones posibles) de participación financiera en un proyecto de software libre. Además, en este modelo de financiación también influyen aspectos que no sólo dependen del mismo proyecto sino también de su entorno y contexto de actuación.

A grandes rasgos, la participación en un proyecto de software libre tiene relación con la colaboración de sus participantes, el modelo de negocio que explota la empresa que lo promueve (en caso de existir), las actuaciones de marketing llevadas a cabo, las licencias de los productos involucrados y las donaciones realizadas.

Contratos indefinidos

El equipo de desarrolladores de la aplicación es muy importante para el desarrollo del proyecto y su evolución futura. La estabilidad y permanencia de los participantes en sus puestos de responsabilidad permite solidificar las bases y la credibilidad del proyecto delante de la comunidad de usuarios.

Un proyecto estable

La credibilidad es fundamental para todos los actores relacionados directa o indirectamente con el proyecto, puesto que no se puede transferir a los eventuales sustitutos, y su pérdida puede afectar en mayor o menor medida al futuro de la aplicación y del proyecto, por lo que debemos tomar medidas adecuadas para controlar y gestionar activamente el proyecto.

Descentralización

Una de las características más relevantes (y deseables) de las comunidades de usuarios de software libre es la distribución y descentralización de las decisiones que se toman en el proyecto.

En este sentido, la organización del proyecto debe tener en cuenta esa estructura como forma de motivar y reforzar la comunidad de usuarios de la aplicación, de manera que el consenso surja de la misma interacción entre sus miembros.

Transparencia

El anterior aspecto de descentralización nos da una idea de la transparencia y justificación que debe reinar en la relación entre proyecto y comunidad.

Tanto los objetivos del proyecto como las líneas de evolución de la aplicación, deben estar claros y ser conocidos por todos los implicados en el mismo. En este sentido, la influencia del promotor sobre la evolución futura debe presentar un comportamiento honesto y transparente, de forma que garantice la credibilidad del proyecto[3].

(3)Por ejemplo, el manifiesto de Openbravo (http://www.openbravo.com/es/about-us/openbravo-manifesto/).

Credibilidad

La credibilidad del proyecto (tanto en conjunto como la de sus miembros) ha aparecido en muchos de los aspectos que hemos ido comentando hasta ahora. Su relevancia está muy relacionada con la comunidad de usuarios de software libre y supone una condición importante para el mantenimiento a lo largo del tiempo.

El dinero o la posición jerárquica no pueden generar la credibilidad necesaria en las actuaciones de cada uno de los miembros en cada momento. Es decir, la metodología, los procedimientos o protocolos establecidos, o el funcionamiento u operativa deben ser los mismos para todos sin excepción.

Contratos

La contratación de trabajadores es un aspecto a cuidar especialmente en los proyectos de software libre a causa de las repercusiones en la estructura y funcionamiento. Hay que velar para que todos los detalles y procesos relacionados con la contratación se mantengan abiertos y transparentes..

De hecho, es importante revisar y aceptar estos cambios con la colaboración de la comunidad de usuarios, hasta el punto de que en algunos casos puede ser preferible o deseable contratar directamente desarrolladores de la comunidad con permisos de escritura sobre el repositorio oficial (*committers*).

Recursos

El proyecto de software libre no sólo está basado en la evolución y mantenimiento del código de una aplicación basada en software libre, sino que debe considerar también otros aspectos de soporte complementarios.

Recursos complementarios

Este es el caso de la gestión de la calidad del código producido, la protección legal de las contribuciones, la documentación y utilidad de la aplicación y la provisión de recursos de infraestructura para la comunidad de software libre (páginas web, sistemas de control de versiones, etc.).

Estos recursos pueden motivar diferencias significativas en la difusión y popularización tanto de la aplicación como del proyecto en la comunidad de usuarios de software libre.

Marketing

Finalmente, aunque se trate de un proyecto basado en software libre, deben aplicarse medidas de marketing para su difusión y popularización tanto de la aplicación como del proyecto en general.

En este sentido es importante recordar que todo el funcionamiento del proyecto está expuesto al público en general, y cada una de las afirmaciones que se vierten pueden resultar fácilmente demostrables o revocables. El establecimiento de medidas para controlar la imagen y funcionamiento del proyecto han de permitir ganar credibilidad, transparencia y verificabilidad.

De entre estas medidas es importante remarcar la importancia de mantener una política abierta, honesta y objetiva respecto de los proyectos competidores. Por una parte porque sustenta un valor seguro para la comunidad de usuarios, y por la otra porque favorece el desarrollo de estrategias de coopetencia entre proyectos conexos.

2. La comunidad de usuarios

Tal como se ha puesto de relieve en el primer apartado de este módulo, el papel que juega la comunidad de usuarios de software libre es muy importante en el paradigma de desarrollo de software libre.

Tanto los usuarios como los desarrolladores que forman parte de la comunidad colaboran en el mantenimiento, soporte y evolución de la aplicación a lo largo del tiempo, asegurando así la cohesión y estabilidad del proyecto.

En consecuencia, su participación es fundamental para dotar de garantías los objetivos del proyecto, y debe ser considerada como tal por cualquier organización lucrativa que pretenda explotar una oportunidad de negocio basada en la producción de software libre.

En este sentido, la relación entre comunidad y empresa debe basarse en la credibilidad y la transparencia de todas las actuaciones y decisiones que se toman, de forma que ambas partes puedan extraer provecho de esta relación. No en vano el posicionamiento de la empresa con respecto a los productos basados en software libre debe estar bien definido y estructurado para favorecer la creación de una comunidad de colaboradores a su alrededor.

Hay que tener en cuenta que la comunidad de usuarios es una organización dinámica y evolutiva en el tiempo, por lo que será necesario establecer metodologías de gestión que permitan mantener la relación en su estado óptimo. Esta gestión incluye el establecimiento de procedimientos para identificar el estado actual de la comunidad, evaluar la calidad de las contribuciones al proyecto por parte de los miembros y definir los aspectos legales relacionados con estas contribuciones.

En los próximos apartados examinaremos cada uno de estos aspectos de forma individual.

2.1. Gestión de la comunidad

Para conseguir los objetivos el proyecto, la empresa que emprende un proyecto de desarrollo de software libre debe estructurar minuciosamente su relación con la comunidad de usuarios.

En el primer apartado de este módulo ya hemos mencionado los principales aspectos sobre los que fundamentar un proyecto de software libre. En caso de que una empresa actúe como promotora del proyecto, será necesario establecer y organizar una estrategia adecuada a los objetivos empresariales, pero teniendo en cuenta que tiene que ofrecer contrapartidas a la colaboración que espera obtener de la comunidad de usuarios.

En este sentido, y como en cualquier otro proyecto de software libre, aspectos como la credibilidad o la transparencia –entre otros– tendrán un papel muy importante en la creación de una comunidad de usuarios en torno al proyecto.

Ben Collins-Sussman y Brian Fitzpatrick han identificado y clasificado las diferentes estrategias *Open Source* que puede seguir una empresa basada en el desarrollo de software libre en la conferencia titulada "What's in it for me?" de OSCON 2007.

Web recomendada

B. Collins-Sussman; B. Fitzpatrick (2007). "What's in it for me?"
(http://www.youtube.com/watch?v=ZtYJoatnHb8).

Esta clasificación caracteriza los dos componentes principales de la relación entre empresa y comunidad:

- por una parte, la orientación, la estructura y el funcionamiento general del proyecto, así como la responsabilidad de la empresa en lo mismo.
- De la otra, los beneficios y los inconvenientes para la empresa y para la comunidad de usuarios resultantes de escoger una estrategia concreta para llevar a cabo el proyecto.

En este sentido, el trabajo de Collins-Sussman y Fitzpatrick describe una guía de buenas prácticas para formalizar una relación saludable entre empresa y comunidad de usuarios.

En los próximos apartados presentaremos de forma breve los principales rasgos de esta clasificación de estrategias *Open Source*.

Fake Open Source

Esta estrategia se basa en abrir o liberar el código fuente de la aplicación bajo una licencia no aprobada por OSI.

Web recomendada

Open Source Initiative (http://www.opensource.org/).

En realidad, no se trata de una estrategia *Open Source* real, puesto que no sólo se pierden los beneficios[4], sino que incluso algunos miembros de la comunidad podrían llegar a boicotear el proyecto.

(4)Por ejemplo, la mejora del software, la credibilidad del proyecto o la fortaleza de las relaciones entre empresa y usuarios.

A pesar de esto, el proyecto puede obtener cobertura mediática y atraer atención con un coste o esfuerzo relativamente bajo.

Throw code over the wall

Esta estrategia es similar a la anterior, pero esta vez la empresa abre o libera el código bajo una licencia aprobada por OSI, aunque sigue sin preocuparse o responsabilizarse por el futuro del proyecto.

Es decir, abriendo el código y olvidándose de él, la empresa difunde una imagen de poca credibilidad, puesto que libera una aplicación para la que no existe una comunidad de usuarios que permita mantener el proyecto activo. En este caso, es posible que se puedan crear comunidades alternativas que desarrollen el software al margen de los objetivos empresariales.

Develop internally, post externally

Esta estrategia se basa en desarrollar la aplicación de forma interna en la empresa, publicando los avances en un repositorio público.

De esta forma, la empresa mejora tanto las relaciones públicas con la comunidad de usuarios como la credibilidad dentro del mundo del software libre. Por su parte, la comunidad podría puntualmente colaborar en el proyecto. Aún así, el desarrollo totalmente interno incita a la creación de comunidades paralelas que no sigan el calendario empresarial (que genera una cierta desconfianza).

Open monarchy

Esta estrategia se basa en la exposición pública tanto de las discusiones como del repositorio de la aplicación, aunque los usuarios con derechos sobre él son internos a la empresa.

En este caso, se mejora sustancialmente la credibilidad y transparencia de las empresas y las aportaciones por parte de la comunidad (lo que redunda en mejor código), aunque la empresa sigue teniendo la última palabra en todas las decisiones que se toman. Este hecho constituye un riesgo para el mantenimiento de la comunidad a largo plazo, incluso existe riesgo de bifurcación (*fork*) en el proyecto.

Consensus-based development

Esta estrategia explota prácticamente todas las posibilidades de relación entre empresa y comunidad, puesto que prácticamente todo se realiza de forma pública.

En este caso, el proyecto se basa tanto en la toma de decisiones distribuida y descentralizada como en sistemas de funcionamiento meritocráticos entre los colaboradores.

Estas características redundan en un modelo sostenible a largo plazo con voluntarios de alta calidad, puesto que la empresa gana en credibilidad, transparencia y confianza delante de la comunidad y otras empresas basadas en software libre.

Aún así, los beneficios a corto plazo son escasos y el volumen de trabajo es significativo. En este caso, el rol de los líderes del proyecto es relevante para el funcionamiento estratégico de toda la organización.

2.2. Características de la comunidad

La comunidad de usuarios de software libre es una organización dinámica y evolutiva, en el sentido que existen diversos factores que influyen y moldean en mayor o menor medida su situación y tendencia futura.

Considerando un proyecto de software libre, es deseable crear cuanto antes una comunidad de usuarios sólida en torno a la aplicación, puesto que parte del éxito y de los objetivos del proyecto redundan en ella.

Una vez creada la comunidad, es importante programar actuaciones que permitan no sólo mantenerla estable sino también ampliarla y hacerla evolucionar al menos al mismo ritmo que lo hace el producto. Como paso previo a cualquier actuación en este sentido, es necesario conocer con precisión el estado actual de la comunidad de usuarios y la tendencia evolutiva que sigue en los últimos tiempos en relación con el proyecto.

Identificar con exactitud el estado actual de una comunidad de usuarios puede resultar relativamente complicado en la práctica, principalmente por sus características de distribución y descentralización[5].

(5)Esta problemática puede trasladarse y ejemplificarse con la problemática de evaluar la situación en un momento dado (fotografía instantánea) de un sistema distribuido o descentralizado.

Aún así, podemos tener en cuenta una serie de indicadores que nos permitan establecer una aproximación suficientemente realista como para tomar decisiones en este sentido.

En el artículo de Crowston y Howison "Assessing the Health of a FLOSS Community" se detalla una guía simple, pero efectiva, para identificar y evaluar el estado de una comunidad de usuarios de software libre. Esta guía considera los principales indicadores que se deben tener en cuenta para evaluar la salud de la comunidad y, por extensión, del proyecto basado en software libre.

Web recomendada

K. Crowston; J. Howison (2006). "Assessing the health of a FLOSS Community" (http://floss.syr.edu/publications/Crowston2006Assessing_the_health_of_open_source_communities.pdf)

En los próximos apartados presentaremos de forma breve algunas de sus conclusiones.

Ciclo de vida y motivaciones

Diversos autores están de acuerdo en considerar que los proyectos se inician en el seno de un grupo reducido de promotores para luego estructurarse y desarrollarse de forma pública.

Una vez puesto en marcha el proyecto, se debe iniciar una segunda fase que permita el refinamiento progresivo del concepto inicial. Es decir, la compartición de ideas, sugerencias y conocimientos ha de permitir revolucionar el concepto original. Este proceso no se puede realizar sin la cooperación de la comunidad de software libre.

Por otra parte, las motivaciones de los miembros de la comunidad para participar en el proyecto se relacionan principalmente con el desarrollo intelectual, el compartimiento del conocimiento, el interés por la aplicación, la ideología o filosofía subyacente al proyecto o al software libre, la reputación u obligación comunitaria.

Estructura y tamaño de la comunidad

La comunidad de usuarios de una aplicación basada en software libre puede estructurarse de muy diversas formas, teniendo en cuenta las actuaciones y decisiones de los promotores del proyecto y las características de la aplicación y/o de su producción.

En general, podemos considerar que la comunidad de usuarios de una aplicación es saludable si presenta una estructura jerárquica funcional adecuada a sus objetivos en torno a un núcleo activo de desarrolladores.

Estructura jerárquica

Este concepto puede relacionarse con la estructura de capas de una cebolla (*onion-shaped* en inglés), de forma que en el centro se sitúen los miembros más activos e implicados con el proyecto y en la capa más externa los menos participativos.

A grandes rasgos, podemos identificar las siguientes tipologías de miembros en un proyecto:

- Desarrolladores del núcleo de la aplicación, con derechos de escritura sobre el repositorio y un historial de contribuciones al proyecto significativo.

- Líderes del proyecto, que motiven y lleven el proyecto y su comunidad de usuarios a la madurez y estabilidad.

- Desarrolladores en general, que aportan código pero no tienen derechos de escritura sobre el repositorio. A menudo realizan tareas de revisión.

- Usuarios activos, que prueban la aplicación, informan sobre errores, crean documentación y enlazan el proyecto con los usuarios pasivos (entre otras actividades).

Nota

Esta primera clasificación de tipologías no es una estructura cerrada, puesto que cada proyecto la adapta a sus características particulares.

Procesos de desarrollo

El proceso de desarrollo de software libre puede resultar en muchas ocasiones poco formalizado en los proyectos, debido principalmente a la ausencia de mapas de ruta, asignaciones explícitas de trabajo o falta de priorización de las funcionalidades de la aplicación.

La organización del proyecto es una característica relevante para el funcionamiento y coordinación de la producción, aunque un cierto grado de duplicación de esfuerzos puede considerarse como un síntoma positivo de la relación e implicación de la comunidad en el proyecto.

Del mismo modo, el ciclo de evaluación y posterior aceptación de las contribuciones de los miembros de la comunidad al proyecto aporta información precisa sobre la salud de su funcionamiento. Por ejemplo, la refutación de una contribución puede demostrar una visión cohesionada y cualitativa del proyecto a largo plazo.

2.3. Gestión de la calidad

En ocasiones, la calidad del software libre ha sido foco de debate entre promotores y detractores, poniendo de relieve aspectos como la apertura del modelo de desarrollo o la capacitación de los colaboradores que contribuyen al proyecto, por ejemplo.

Como en cualquier proyecto de software, la producción de software libre debe establecer medidas de control de la calidad a lo largo de su ciclo de vida. Es decir, la calidad debe poder evaluarse y compararse con los baremos esperados en cualquier etapa de la producción y de la explotación, así como desde cualquier punto de vista (promotores, usuarios o comunidad).

En este sentido, si bien la apertura y descentralización del modelo de desarrollo del software libre favorece los mecanismos de control y gestión de la calidad, no constituye una solución por sí misma ni debe dejarse sin planificar en base a estas particularidades.

Para desarrollar los aspectos relacionados con la calidad en la producción de software libre, tendremos en cuenta el artículo "Managing Quality in Open Source Software" de Dhruv Mohindra, que realiza un estudio detenido sobre el control de calidad en entornos de software libre. En los próximos apartados revisaremos los principales conceptos que presenta el artículo.

Web recomendada

D. Mohindra (2008). "Managing Quality in Open Source Software"
(http://www1.webng.com/dhruv/material/managing_quality_in_oo.pdf).

La calidad en el software libre

En general, la calidad de una solución de software puede valorarse tanto por su arquitectura o diseño interno como por las funcionalidades que proporciona al usuario.

Las particularidades de apertura y descentralización del modelo de desarrollo del software libre constituyen una infraestructura que permite establecer políticas de gestión de la calidad basadas en la localización y resolución de problemáticas (entre otros aspectos). Aun así, en ocasiones la falta de claridad y/o estructura de los procesos de producción puede generar resultados que no son los esperados.

Evaluación de la calidad

Existen diversas metodologías y métricas formales para evaluar la calidad funcional de una aplicación. Las métricas cuantificables dependen en gran medida de la tipología del mismo software, por lo que deben ser elegidas en función de las características y objetivos de la aplicación.

En relación con la calidad no cuantificable, cabe destacar el papel que tiene la comunidad de software libre en este aspecto. Por una parte, los tests que realiza el equipo de calidad, y por otra, la propia actividad de los usuarios de la aplicación denotando las evidencias de errores de funcionamiento o de mejora del producto.

En este sentido, el mismo carácter y funcionamiento distribuido y descentralizado de la comunidad de usuarios también es importante para dotar de más garantías de calidad el proceso de producción.

Control y revisión

Un factor importante para la calidad del producto final es el control y la revisión de todo el proceso de desarrollo. En general, los proyectos de producción de software libre utilizan sistemas de control de versiones para soportar con eficiencia y eficacia la evolución de los diferentes componentes del proyecto.

Hay distintas formas de organizar el control y la revisión de la evolución del software, así como las ramas de desarrollo y repositorios (entre otros). En cualquier caso, conviene adaptar la metodología de producción y los sistemas de control y revisión de las evoluciones a las particularidades del proyecto y del producto que se está creando.

Mitos del software libre

A pesar del paso de los años, el software libre todavía arrastra algunos mitos –tanto positivos como negativos– que pueden influir en mayor o menor medida en su evaluación.

Estos mitos no tienen ninguna base sólida sobre la que fundamentar una gestión coherente y sostenible de la calidad, por lo que se debe concretar y evaluar cada uno de ellos por separado.

A continuación se comentan algunos mitos habituales relacionados con la calidad del software libre.

- El hecho de que el código fuente sea público no garantiza que sea seguro y/o de calidad, puesto que en cualquier caso depende del interés y revisión de la comunidad.

- La congelación de funcionalidades no aumenta la estabilidad de la aplicación por sí misma, porque lo importante es escribir buen código desde el principio.

- La mejor forma de entender un proyecto no es corregir sus eventuales fallos, puesto que la documentación es significativamente mejor para este objetivo.

- En general, los usuarios no disponen de la última versión del repositorio con la corrección de errores al día.

A grandes rasgos, los procesos de prueba y revisión, así como las discusiones públicas y la cultura *hacker* propias de la comunidad de usuarios, deben complementarse con una planificación y gestión activa de la calidad de la producción.

Esta gestión debe ir encaminada a cubrir eventuales lagunas en uno o más aspectos del producto, por ejemplo la planificación de la producción, desarrollo de las funcionalidades o la documentación de la aplicación.

Consideraciones adicionales sobre la calidad

En general, tanto la publicación del código fuente, como la incorporación de sistemas de gestión de errores y el hecho de compartir la responsabilidad sobre el producto entre todos los implicados son aspectos clave de la gestión de la calidad.

En este sentido, también resulta importante para la calidad general del proyecto considerar la transparencia en todas las actuaciones, confiar en el equipo de desarrollo, revisar y probar todas las partes del código fuente y promover tanto la filosofía de igual a igual como la importancia de hacerlo bien desde el principio.

2.4. Legalidad y contribuciones

En un proyecto basado en software libre con participación de la comunidad de usuarios, resulta especialmente relevante la gestión legal de las contribuciones de cada miembro implicado.

Esta gestión es importante tanto para los promotores del proyecto como para los miembros de la comunidad, puesto que establece las características de autoría y titularidad de los derechos del código resultante. Su relevancia también se ve especialmente influida por las implicaciones que puede tener la combinación de códigos de diferentes autores en un mismo producto.

Para desarrollar estos conceptos partiremos de la lectura del apartado 2.4 "Autores y titulares de derecho" de los materiales didácticos de la asignatura *Aspectos legales y de explotación del software libre*.

Web recomendada

M. Bain y otros (2007). *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya

(http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/aspectes-legals-i-dexplotacio/materials/).

El autor

El autor de una obra es la persona física o jurídica que crea la obra, por lo que irrenunciablemente le pertenece la autoría de su creación original.

Las obras realizadas por diversas personas permiten distinguir entre varias situaciones:

- Una obra en colaboración es el resultado unitario de una composición de diversas partes que pueden ser explotadas independientemente.

- Una obra colectiva es la reunión de distintas contribuciones que no se pueden explotar de forma independiente.

- En una obra encargada (o con contrapartidas económicas), la autoría recae sobre la persona física o jurídica que realiza el encargo.

En el mundo del software libre, la autoría depende en gran medida de las consideraciones anteriores, teniendo en cuenta que en algunas ocasiones la transferencia de la titularidad puede resultar útil y práctica.

Por otra parte, las condiciones en las que se realizan las obras derivadas (preexistencia de contenido) pueden variar sustancialmente a causa tanto del autor como de la misma obra. En cualquier caso, las licencias libres deben especificar las condiciones de derivación y redistribución de las obras.

El titular original y el titular derivado

El titular original de la obra es siempre su autor. Aun así, algunos derechos sobre la obra pueden ser cedidos a otras personas, ya sean físicas o jurídicas.

En este caso, la persona que recibe la cesión de parte de los derechos sobre una obra se convierte en titular derivado de la misma. Conviene recalcar que sólo el titular de un derecho concreto puede conceder licencias sobre ese derecho.

Identificación del titular

Para poder ejercer los derechos anteriormente comentados tiene que ser posible identificar al autor de cada obra. Esto puede resultar complicado en el mundo del software libre, ya que los contribuyentes al proyecto pueden ser múltiples y variados.

Para paliar estos problemas, los proyectos basados en software libre mantienen listas de los autores que han contribuido. En algunas ocasiones, estos proyectos pueden requerir la cesión de la totalidad o parte de los derechos antes de aceptar la contribución.

3. Caso de estudio

En los apartados anteriores se examinan tanto el proyecto de software libre como la gestión de la comunidad de usuarios. Ambos apartados presentan los principales aspectos relacionados con la producción de software libre desde el punto de vista de gestión del proyecto.

Para finalizar este módulo, dedicaremos el último apartado a concretar buena parte de las ideas y propuestas presentadas anteriormente estudiando un caso concreto de empresa basada en software libre.

Los próximos apartados pretenden servir de guía para identificar y clarificar cómo una empresa basada en la producción de software libre implementa su metodología particular, formaliza y gestiona la relación con la comunidad de ususarios y cómo afronta muchas de las decisiones que debe tomar a medida que avanza el tiempo.

En este apartado nos centraremos en el caso de Openbravo, S. L.

3.1. La empresa

Openbravo, S. L. es una empresa dedicada a desarrollar soluciones profesionales basadas en software libre para las empresas.

Nota

Toda la información que se presenta en este apartado se ha obtenido principalmente de su sitio web (http://www.openbravo.com/).

Modelo de negocio

El modelo de negocio que explota la empresa se basa en la prestación de servicios relacionados con los productos que desarrolla. Tal y como se ha comentado en otros módulos, su estrategia de negocio está basada en el asociacionismo y la coopetencia entre empresas para explotar la misma oportunidad de negocio.

Estrategia empresarial

El modelo de negocio se implementa mediante *partners* que ofrecen servicios a los clientes finales (como por ejemplo la personalización y el soporte). En cierto modo, esta particular jerarquía entre productor, distribuidor (o *partner*) y cliente establece una atmósfera de cooperativismo con objetivos comunes.

Para completar esta estrategia, la empresa publica un manifiesto a modo de declaración de intenciones, que incluye tanto aspectos relacionados con el software libre (por ejemplo, la transparencia, la apertura o la colaboración), como los compromisos de la empresa respecto a terceras personas (por ejemplo, accesos libres o gestión de contribuciones).

Gestión y dirección

La gestión de la Empresa combina cargos internos y externos a la organización, tanto en el equipo directivo como en el Consejo de Administración, producto de la inversión externa que ha recibido la empresa y también de la particular metodología basada en software libre.

3.2. Los productos

Openbravo produce dos soluciones basadas en software libre que funcionan de forma independiente o en combinación. Ambos productos son distribuidos bajo licencias libres y con descarga directa a través de Internet.

Los productos que ofrece Openbravo son los siguientes:

- **Openbravo ERP**
  Openbravo ERP (*Enterprise Resource Planning*) es un sistema de gestión empresarial en entorno web, que integra de forma modular diversas funciones de gestión, como aprovisionamientos, almacén, producción o contabilidad.
  El producto está licenciado bajo MPL 1.1 y puede funcionar en diferentes entornos y sistemas de base de datos e integrarse con Openbravo POS.

  Entre las muchas informaciones que se proveen sobre el producto destaca el mapa de ruta de desarrollo del proyecto.

Web recomendada

Mozilla Public License 1.1 (http://www.mozilla.org/MPL/MPL-1.1.html).

Características principales de Openbravo ERP

http://sourceforge.net/projects/openbravo/.

- **Openbravo POS**
  Openbravo POS (*Point Of Sales*) es un sistema de terminal de punto de venta que puede ser integrado con Openbravo ERP.
  El producto está licenciado bajo GNU/GPL y puede funcionar en diferentes entornos y con diversos sistemas de base de datos. Está especialmente diseñado para terminales táctiles.

  Entre las informaciones que se proveen también destaca el mapa de ruta del producto de desarrollo del proyecto.

Web recomendada

GNU General Public License (http://www.gnu.org/licenses/gpl.html).

Características principales de Openbravo POS

http://sourceforge.net/projects/openbravopos/.

3.3. La comunidad de usuarios

La comunidad de usuarios de software libre juega un papel relevante en la estrategia empresarial de Openbravo. En los siguientes apartados mencionaremos los principales aspectos.

Estrategia *Open Source*

Para identificar la estrategia *Open Source* de Openbravo debemos tener en cuenta las particularidades de la metodología de desarrollo de los productos así como también la estructura de negocio que los explotan.

En este sentido, el núcleo de ambos productos es principalmente desarrollado de forma interna a la empresa, manteniendo repositorios públicos y una comunidad de usuarios activa a su alrededor. Por otra parte, en el desarrollo de los complementos, personalizaciones y extensiones sobre el producto original entran en juego tanto la comunidad de usuarios como los *partners*.

Este último caso debe analizarse por separado, puesto que corresponde a la explotación de una oportunidad por parte de una organización diferente.

Con todo esto, Openbravo presenta una estrategia *Open Source* que combina diferentes orientaciones:

- Para el desarrollo y revisión de los productos, la estrategia se aproxima a *Open Monarchy*, debido principalmente al desarrollo interno del núcleo de los productos, los repositorios públicos de código fuente, la aceptación final de los cambios sobre el núcleo a cargo de la empresa y la planificación del desarrollo de los productos (por ejemplo, los mapas de ruta establecidos).

- Para el desarrollo de los complementos (por ejemplo, la documentación, etc.), la estrategia se aproxima al *Consensus-based development*, debido principalmente a su desarrollo dentro de la comunidad de usuarios.

- Finalmente, la estrategia en el desarrollo de las extensiones y de las personalizaciones depende del desarrollador que las implemente. Si son proyectos realizados en el seno de la comunidad (mediante los recursos ofrecidos por Openbravo), posiblemente se aproximen al modelo de *Consensus-based development*, mientras que si son desarrolladas por los *partners* dependerán tanto de su estrategia particular como de las características del desarrollo.

Estrategia del *partner*

En el caso de que el *partner* desarrolle extensiones del producto original, la estrategia *Open Source* dependerá tanto de su filosofía empresarial como de las características del producto (por ejemplo, la licencia MPL es más flexible con los módulos propietarios que la GPL).

Estructura de la comunidad

La comunidad de usuarios de Openbravo ERP está definida y estructurada en forma de sistema meritocrático: existen varios niveles de colaboración y cada uno de ellos se define a partir de los conocimientos necesarios para este nivel, la cantidad de contribuciones, las responsabilidades y los privilegios.

En el caso de Openbravo ERP existen tres perfiles diferentes de colaboración (desarrolladores, expertos funcionales y probadores), mientras que en el caso de Openbravo POS sólo existen desarrolladores. Los miembros de la comunidad de usuarios se organizan y se distribuyen por proyectos activos en la comunidad.

Recursos a disposición de la comunidad

Openbravo dispone de diversos recursos (algunos de ellos en más de una lengua) tanto para la comunidad como para los *partners* o para los usuarios en general, entre los que destacan los siguientes:

Web recomendada

Desde el portal web de la empresa se puede acceder a todos los recursos mencionados (http://www.openbravo.com/).

- web corporativa
- área de *partners*
- wiki del proyecto
- portal de la comunidad de usuarios de Openbravo
- *blogs* de trabajadores
- forja de los productos (Openbravo ERP y Openbravo POS)
- *bug tracker*
- universidad
- listas de distribución de correo
- repositorio de código de Openbravo
- servicio de noticias de Openbravo

En general, la comunidad de usuarios tiene a su disposición una guía específica disponible en el wiki que describe cómo colaborar en el proyecto. También dispone de una lista exhaustiva de canales de comunicación a los que puede acceder. Adicionalmente, los mapas de ruta de cada producto completan la guía de recursos para la comunidad de usuarios.

3.4. Posicionamiento y evolución

La empresa nació en 2001 bajo el nombre de Tecnicia. En 2006 obtuvo financiación por más de seis millones de dólares cuando pasó a llamarse Openbravo. Ese mismo año liberó el código fuente de los productos que desarrolla bajo licencias libres.

En mayo del 2008, la ronda de financiación ascendió a más de doce millones de dólares, contando entre sus inversores a Sodena, GIMV, Adara o Amadeus Capital Partners.

A lo largo de los años, Openbravo ha recibido varios premios relacionados con el mundo de la empresa y del software libre, así como subvenciones del programa de fomento de la investigación técnica (PROFIT) del Ministerio de Industria, Turismo y Comercio de España.

Tanto la empresa como la comunidad de usuarios mantienen una tendencia evolutiva positiva, si tenemos en cuenta que el proyecto se sitúa en la actualidad entre los veinticinco más activos de SourceForge con más de un millón de descargas acumuladas a principios de 2009.

Web recomendada

Sobre los proyectos más activos de SourceForge:
http://sourceforge.net/top/mostactive.php?type=week.

Resumen

A lo largo del módulo se han presentado las principales características relacionadas con la creación, gestión y mantenimiento del proyecto de desarrollo de software libre, teniendo en cuenta la participación de la comunidad de usuarios.

En cierto modo, los fundamentos de la producción de software libre no difieren demasiado con respecto a las metodologías de desarrollo de software más tradicionales. Aun así, las características de la apertura del código y la presencia de la comunidad de usuarios moldean el funcionamiento y lo particularizan en muchos aspectos.

En lo que respecta al proyecto en sí, cabe destacar la importancia de identificar, definir y estructurar tanto los aspectos funcionales del proyecto (por ejemplo, la infraestructura, la gestión de versiones o las medidas de coordinación), como aquellos relacionados con el software libre (por ejemplo, la credibilidad, la transparencia o las tipologías de participación).

A estos aspectos se tienen que sumar aquellos factores relacionados con la comunidad de software libre, como la estrategia de gestión de la comunidad por parte de la empresa (estrategia *Open Source*), la metodología y ciclo de vida del producto, la gestión de la calidad y los aspectos legales de las contribuciones de los usuarios.

Finalmente, se ha presentado un caso de estudio representativo de buena parte de los aspectos examinados en los diferentes apartados.

Bibliografía

Bain, M. y otros (2007). *Aspectes legals i d'explotació del programari lliure*. Universitat Oberta de Catalunya <http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/aspectes-legals-i-dexplotacio/Course_listing> [Fecha de consulta: febrero del 2009].

Collins-Sussman, B.; Fitzpatrick B. (2007). *What's in it for me? How your company can benefit from open sourcing code*. OSCON: 27 de julio del 2007 <http://www.youtube.com/watch?v=ZtYJoatnHb8> y diapositivas <http://www.red-bean.com/fitz/presentations/2007-07-27-OSCON-whats-in-it-for-me.pdf> [Fecha de consulta: febrero del 2009].

Crowston, K.; Howison, J. (mayo, 2006). *Assessing the Health of a FLOSS Community*. IT Systems perspectivas (pág. 113-115). <http://floss.syr.edu/publications/Crowston2006Assessing_the_health_of_open_source_communities.pdf> [Fecha de consulta: febrero del 2009].

Fogel, K. (2005). *Producing Open Source Software: How to Run a Successful Free Software Project.* <http://producingoss.com> [Fecha de consulta: febrero del 2009].

Mako, B. (2001). *Free Software Project Management HOW TO*. <http://mako.cc/projects/how-to> [Fecha de consulta: febrero del 2009].

Mohindra, D. (2008). *Managing Quality in Open Source Software*. <http://www1.webng.com/dhruv/material/managing_quality_in_oo.pdf> [Fecha de consulta: febrero del 2009].

Openbravo <http://www.openbravo.com/> [Fecha de consulta: febrero del 2009].

Raymod, E. (1997). *The cathedral and the bazaar* <http://www.catb.org/~esr/writings/cathedral-bazaar/> [Fecha de consulta: febrero del 2009].

Tawileh, A. y otros (agosto, 2006). *Managing Quality in the Free and Open Source Software Community* (págs. 4-6). Proceedings of the Twelfth Americas Conference on Information Systems. México: Acapulco. <http://www.tawileh.net/anas//files/downloads/papers/FOSS-QA.pdf?download> [Fecha de consulta: febrero del 2009].