Nuevos Modelos de Negocio
basados en
Software Libre

por **Jose Ramón Díaz**

Diciembre de 2004

Tesina del MBA (UNED)
Fundación Guadalquivir

Prólogo:

¿Por qué publico este trabajo? Este trabajo lo he desarrollado como proyecto de fin de master. Me decidí a estudia un master en Dirección y Administración de Empresas porque tenía ganas de "escaparme" un poco de la informática a nivel de conocimientos. Soy Ingeniero informático, y siempre he pensado que la informática es lo más alejado de la vida real que puedes estudiar, no tiene representación en la naturaleza. Pero ese es otro tema que merecería otro trabajo.

En la carrera no aprendes cómo funciona una empresa. Sales al mercado laboral y ves que tienes que dejar de creerte el ombligo de la empresa, por muy en el "filo tecnológico" que ésta se encuentre. No espero dedicarme en un corto o medio plazo a funciones que pudiesen estar relacionadas con estos estudios del master, me encanta el trabajo que hago. Me encanta la informática, pero creo necesario para cualquier persona algunos conocimientos del mundo laboral más allá de sus propios estudios.

Pero bueno, ¿por qué lo publico? Por varias razones, para devolver en cierta manera lo que el software libre me ha dado tantas veces, para poder contribuir con otra visión del tema, para que la gente interesada en el tema tenga algo más que leer si lo consideran de interés (¡gracias!). Y también, para que nos vamos a engañar, para que no quede olvidado en el fondo de un cajón de mi casa y lo rescate dentro de unos años pensando que lo podía haber lanzado al mundo,.. o hasta donde llegue.

www.najaraba.com

A Itsaso, por su comprensión, ayuda y paciencia.

[1] Esta obra está bajo una licencia Reconocimiento-NoComercial-SinObraDerivada de Creative Commons. Para ver una copia de esta licencia, visite http://creativecommons.org/licenses/by-nc-nd/2.1/es/ o envíe una carta a Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Índice:

1. Introducción

En este trabajo se va a explicar cuales son los cambios que se están produciendo en un sector industrial, el del software, debido a un movimiento social y empresarial: el software libre, que va a generar nuevos modelos de negocio.

El software libre está generando una inusitada reacción y debate entre sus seguidores y sus detractores. Los miles de millones de euros que mueven esta industria del software hacen que el impacto potencial de cualquier cambio en la estructura del sector genere una acción y reacción por parte de las numerosas partes implicadas, e incluso con repercusiones en la sociedad en general.

Este trabajo se centra en estudiar cómo afecta el software libre al sector industrial desde la perspectiva de la administración y dirección de empresas. Para ello, primeramente situaremos el contexto del software, sus características como producto, y el mercado que existe alrededor. El software tiene unas determinadas características especiales que le diferencian a la hora de su producción y distribución, y que es importante conocerlas y tener en cuenta para evaluar el impacto del software libre.

Después, en el tercer capítulo, veremos en qué consiste el movimiento de fuerzas que va a hacer evolucionar este sector industrial cambiando los determinantes estructurales de la competencia en el sector. Se explicará la historia del movimiento social y tecnológico que se conoce como software libre.

Detallaremos posteriormente, en el cuarto capítulo, como cambia el sector industrial, en su estructura, en las estrategias competitivas genéricas llevadas a cabo por las empresas, y cómo está marcando un punto de inflexión muy importante en la evolución del mercado. Estudiaremos además el ciclo de vida de un producto software.

El capítulo cinco nos centrará en la problemática de las posiciones que puede tomar una empresa basada en software libre, incidiendo en la diferencia con el modelo comercial tradicional. Explicaremos en qué beneficia o influye a los clientes estos cambios, y las cuestiones básicas que afectan a una organización, basada en software libre o comercial este nuevo modelo de negocio.

Posteriormente veremos desde el lado del cliente, qué opciones tiene y que ventajas consigue en la adopción del software libre frente al propietario. Como caso particular indagaremos en las características de la administración pública como cliente, o incluso productor también de software de código abierto. Además, detallaré algunos de los programas de mayor éxito que se han generado como software libre.

Por último, veremos el conflicto que se ha creado entre el modelo comercial y los partidarios de los nuevo modelos de negocio, y las conclusiones que pueden afectar a las empresas en el capítulo ocho. Existen posiciones en ambos mundos, propietario y libre, muy críticas con la parte opuesta, y explicaremos cuales son las posiciones antagónicas.

A través de todo este desarrollo, se irá explicando cómo funcionan los nuevos modelos de negocio creados. Para situarnos exactamente en el tema, cuando hablamos de modelos de negocio nos estamos refiriendo a las maneras de hacer negocio, que se basan en: [Alonso 2002]

- La descripción de los clientes que tiene la empresa, los problemas que tienen y cómo es el mercado potencial.

- La descripción de los productos y servicios, y cómo van a solucionar los problemas de los clientes.

- La descripción de como se van a proporcionar a los clientes los productos y/o servicios.

- La descripción de cómo se generan los ingresos y beneficios.

Todos estos puntos están implícitos en el trabajo, pero estructurados más desde el punto de vista de funciones u organización de la empresa. De esta manera, intentaré plasmar los conocimientos adquiridos en el transcurso del Master de Administración y Gestión de Empresas.

2. Introducción al Software

Según el diccionario de la Real Academia de la Lengua Española, el software es "el conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora."

Esta definición, en realidad no da idea de la amplitud de la materia que estamos tratando, ni dice mucho de qué características consta. El software no es un producto tangible en sí mismo. Lo tangible es el mecanismo de soporte del software, discos, CDs,...

Un producto software tiene varias características que hacen especial su proceso de fabricación y distribución. Cada producto es único por sus características, pero el coste de reproducción y distribución puede llegar a ser prácticamente nulo. El software se puede reproducir sin error a través de Internet, y puede llegar a cualquier punto en cuestión de segundos.

El modelo comercial, que vamos a llamar tradicional, de venta de software se basa en la venta de licencias de utilización de un paquete software. Este paquete se entrega a la empresa sin posibilidad de modificación. El cliente tiene derecho al uso del software únicamente bajo limitaciones en su distribución, derecho de copias y uso. Las empresas productoras de software suelen vender también un servicio de actualizaciones periódicas y/o un servicio de formación o consultoría.

Por ejemplo, al comprar un programa de gestión, se puede adquirir un número determinado de licencias, tantas como personas que lo vayan a utilizar en la empresa simultáneamente. Además se puede pagar por un servicio anual de actualizaciones o correcciones de errores y un servicio de atención telefónica para la consulta de problemas. Estos servicios generalmente se renuevan anualmente. Incluso hay empresas de Software que venden las licencias de manera anual, y para seguir utilizándolo se debe "recomprar" el producto anualmente.

El mercado del software está muy diversificado. Existen numerosos submercados, puesto que podríamos hablar de software de sistemas operativos, de software de gestión de empresas, de bases de datos, de aplicaciones ofimáticas, de infografía, de diseño arquitectónico,... así hasta prácticamente los cientos y cientos de tipos de aplicaciones que se desarrollan. Hoy en día la práctica totalidad de los sectores industriales existentes cuentan con la ayuda de un software específico para sus labores de gestión, producción, distribución,... para cualquier área de la organización.

Sin embargo, en este trabajo se aborda el mercado del software como algo general. Las empresas del sector comparten por supuesto unas características comunes que estudiaremos en este trabajo. Evaluaremos las características del sector industrial y sus empresas, y cómo un nueva idea en la construcción y procesos de fabricación del software ha venido a modificar las características del mismo.

Este mercado es muy dinámico, y ha experimentado un gran crecimiento en los últimos años. Numerosas consultoras destacan el crecimiento del sector actualmente, tras la crisis que sufrió en el año 2000, la crisis conocida como la de las punto-com, en referencia a las numerosas empresas que se crearon basadas en las expectativas de Internet como fuente de ingresos.

La distribución de las empresas es diferente según el negocio al que se dediquen dentro del mercado. Microsoft es la compañía que domina el sector de los sistemas operativos y aplicaciones ofimáticas. El caso de Microsoft es muy particular porque es un gigante dentro de este mercado. Su posición dominante en el sector de los ordenadores personales, le ha permitido crecer hasta tener una presencia casi monopolística en algunos terrenos. Esto ha llevado a la opinión en buena parte del sector informático a un "todos contra Microsoft", y de hecho se pelea contra él por todos lo medios: demandas anti-monopolio, apoyos a la competencia.

Existen numerosas empresas que venden un producto para un nicho particular en un sector del mercado. Y también un alto número de empresas que ofrecen servicios de consultoría y formación. Por tanto, el mercado se encuentra dominado por una serie de compañías grandes que además son el referente tecnológico, pero se encuentra muy fragmentado en la mayoría de sectores menores.

No quiero acabar esta introducción del software sin destacar algunas peculiaridades de este. Bajo mi punto de vista, la creación de software es algo particular respecto a otros procesos productivos. Cada vez más se habla de la ingeniería del software, pero hay autores incluso que definen su creación como un arte más que como un problema de ingeniería. Los procesos de Ingeniería aplicados a su desarrollo tienden a cambiar esta perspectiva de manera gradual, pero es cierto que no eliminan de manera total el factor humano en la creación de los productos.

Generalmente, en muchos procesos productivos se diseñan y se construyen prototipos antes de comenzar la fabricación en serie. Estos prototipos son productos creados con toda la funcionalidad, que han sido creados de manera más artesanal antes de automatizar la producción. Pero en software, al crear un prototipo, este se refiere a un producto que únicamente contiene partes de la funcionalidad requerida. Después se va completando. Esta diferencia es importante, ya que explica la imposibilidad de automatizar la creación del software. No podemos tener simulaciones reales de los productos software, por que la única manera de simularlos, es construyéndolos.

Es muy importante observar también la indiferencia entre crear una o diez mil copias de un programa, puesto que ni se necesita nuevo trabajo, ni prácticamente nuevos gastos (relativos). En un producto material estándar, cada creación de una unidad suponen unos gastos de material, trabajo,... sin embargo en el software esto es radicalmente diferente.

Un programa software no se puede simular de manera completa, al igual que se haría con las fuerzas que actúan en la construcción de un edificio, porque para poder hacerlo necesitaríamos construir el programa que queremos simular. Esto añade una determinada incertidumbre que la ingeniería del software intenta disminuir, aplicando métodos de desarrollo de ingeniería para minimizar el factor de "arte" que tiene la creación del software.

Más adelante se abordará este asunto que puede ser un punto importante en los procesos de desarrollo aplicados por las empresas o por las comunidades de usuarios del software.

Hasta aquí he presentado la parte más conocida del mercado del software, compuesto por empresas que venden sus productos por un precio que te da derecho a su utilización, pero no a su distribución para que otras personas lo utilicen, ni a su modificación para adaptarlo a las necesidades de cada uno. Las empresas más importantes, como Microsoft, Oracle, Adobe o Autodesk, venden sus productos bajo estas restricciones.

Pero hace años, de hecho en el inicio del software, se creó otra manera de desarrollar y distribuir el código de los programas. Otro punto de vista que choca frontalmente con las prácticas de negocio de las empresas que mueven miles de millones de euros en el mercado. Últimamente este movimiento ha adquirido mucha relevancia y fuerza en algunos sectores y por ello este trabajo va a tratar de aclarar como afecta al sector industrial del software el cambio de reglas que supone.

Se van a estudiar los cambios que produce en la estructura del sector industrial, desde sus componentes. Posteriormente se verá como las estrategias de las empresas deben adaptarse a los nuevos movimientos, así como el conflicto entre los modelos de negocio actualmente más extendidos, y los nuevos que se plantean por el software libre.

3. Presentación del software de Código Abierto

3.1. Un poco de historia

El movimiento de "open source" o Software de Código Abierto (de aquí en adelante también SCA) es una manera muy especial de crear y distribuir el software. Se trata de que los programas se crean bajo licencias que los hacen libres en su modificación y distribución. Esto implica, por tanto, que se distribuyen con su código fuente.

Primeramente, aclaremos los conceptos sobre la creación del software. El software se crea mediante unos lenguajes comprensibles por los técnico que lo desarrollan (código fuente) y posteriormente por un proceso de traducción, compilación se convierten en programas ejecutables por los ordenadores. Estos ejecutables se quedan convertidos en lenguaje binario o máquina, aquel que comprenden los microprocesadores para su ejecución♦. El lenguaje de especificación de ordenes que debe ejecutar la máquina es el utilizado por los programadores de aplicaciones, para desarrollar los programas. Este lenguaje puede ser de mayor o menor nivel según esté más o menos alejado de la semántica que entienden los microprocesadores.

Pero las máquinas solo "comprenden" secuencias de bits, un número: 0 o 1, que se obtienen por la compilación del código desarrollado por los programadores. Estas órdenes codificadas en formato binario forman los ficheros que realmente ejecutamos en los ordenadores.

Figura 1: Proceso de generación de programas

Cuando una empresa produce un programa del que vende licencias a los usuarios, se distribuye el código binario a los clientes. De esta manera, el usuario puede ejecutar dicho programa y utilizarlo, pero no sabe cómo se ha realizado ni cómo está programado. El código fuente pertenece a la empresa. De hecho este código suele quedar como secreto empresarial.

Por tanto, las empresas que distribuyen el código binario únicamente, se reservan la modificación del programa para ellos mismos. Generalmente, actualizan a los clientes, o les venden nuevas versiones del código binario, conforme liberan una nueva versión de los programas. La mayoría del software utilizado hasta hace unos años en la mayoría de los entornos empresariales y particulares era únicamente distribuido en su formato binario, y distribuido bajo restrictivas licencias de su uso.

♦ Este proceso no siempre sucede de esta manera, pero sí en la mayoría de los casos.

Pero se estaba iniciando hacia los años 70 un movimiento en determinados sectores que promulgaban otra manera de crear y distribuir el software. Una manera que proporcionará a todos los usuarios la libertad de modificar el código fuente y su distribución de manera gratuita. En los inicios de Internet, y prácticamente desde los inicios del software, por finales de los años 70, los programadores iniciaron este movimiento como forma de colaboración entre ellos. Creaban de manera altruista aquellos programas que personalmente necesitaban, pero además los liberaban para su utilización por cualquier usuario. Y cuando descubrían un software que les interesaba, pedían el código fuente a su autor, para copiarlo, mejorarlo, o utilizar partes del mismo en sus desarrollos. Realmente el código abierto es la manera en como se empezó a trabajar en Internet, y no una novedad **[Delio 2003]**.

Este movimiento se concretó cuando Richard Stallman fundó el proyecto GNU (acrónimo de "GNU No es Unix[*]") en 1984. Se trataba de crear las herramientas necesarias para poder utilizar únicamente software libre. Stallman no quería que las licencias y *copyright* de las empresas limitasen la capacidad de los usuarios para la utilización del software. Pensaba que los usuarios debían ser libres de la modificación del software y de su utilización y copia, para que la cooperación se impusiese ante el derecho cedido por los usuarios a las empresas.

La idea de una licencia bajo la cual los usuarios fuesen los verdaderos controladores del software se concretó en la licencia GNU-GPL (Licencia Pública General-GNU). Mediante el software distribuido bajo esta licencia se intentan defender las libertades del usuario del software que Richard Stallman defiende como base de la cooperación en la comunidad. La intención de la licencias es además promover la producción de más software libre y elimina la posibilidad la integración con software propietario.

Las cuatro libertades del usuario de software básicas defendidas son las siguientes **[Santos]**:

1) La libertad de ejecutar el programa con cualquier propósito

2) La libertad de estudiar cómo funciona el programa y adaptarlo a sus necesidades. Esto implica por tanto la distribución del código fuente para poder llevar a cabo las modificaciones.

3) La libertad de distribuir copias, gratuitamente o por un precio.

4) La libertad de mejorar el programa, y distribuir versiones mejoradas para que la comunidad pueda aprovecharse de las nuevas ventajas. Obliga a distribuir el código fuente en caso de distribución binaria.

Se promueve de esta manera la cooperación entre los usuarios de software. La primera intención de esta comunidad era que no hubiese que utilizar ningún programa comercial si no se quería. Por tanto, sus esfuerzos se dirigieron a crear un sistema operativo, pieza fundamental para hacer cualquier ordenador. Las necesidades de financiación llevaron a crear la Fundación para el Software Libre (FSF). Esta fundación se financia principalmente de la venta de los programas libres en un soporte físico, aunque también acepta donaciones.

[*] UNIX es un sistema operativo comercial muy utilizado en entornos empresariales, creado en los años 70.

En el año 1991 aparece Linux, un sistema operativo cuyo nucleo fue desarrollado principalmente por Linus Torvalds♦, y que era un núcleo compatible con Unix. En 1992 Linux se libera bajo licencias GPL. Por fin, el movimiento de software libre tenía su sistema operativo listo para servir de infraestructura a cualquier programa libre. GNU/Linux (el nucleo más muchas herramientas complementarias, formando un sistema operativo más completo) es desde entonces una de los mayores éxitos del software libre. Ha ido madurando gracias a la comunidad y es hoy una muy seria amenaza para sus competidores comerciales.

Richard Stallman distingue ente software libre, y lo que después se ha denominado software de código abierto **[Stallman 2004]**. Lo hace para diferenciar el hecho de la libertad que quiere dar a los usuarios del software, puesto que considera que el movimiento de código abierto se ha creado para atraer a inversores y empresas que basan sus conceptos del software en una visión economista, y no en las libertades que él intenta promulgar.

Stallman insiste mucho en las mejoras sociales del software libre. Para él de esta manera la sociedad gana en libertad, de manera que nadie se vea obligado a usar software bajo licencias restrictivas de modificación o copia. Las mejoras sociales que implicaría el total desarrollo del software libre como soluciones globales, incluirían la accesibilidad de todo el mundo al software y el compartir los conocimientos a nivel global.

En este trabajo nos vamos a referir de las dos maneras al software libre, entendiendo por él aquel que se rige por la licencia GNU-GPL. La principal idea será que el código fuente no es solo "abierto", que está disponible para los usuarios, sino que además el software cuenta con las libertades de su modificación, adaptación y distribución de la manera comentada.

El término "código abierto" tiende a enfatizar las ventajas técnicas de tal software, mientras que quienes usan el término software libre, tienden a enfatizar la libertad de control por otros, y las razones éticas o morales. Lo opuesto al software libre es el software cerrado o propietario, comercialmente vendido bajo licencias más restrictivas. **[Wheeler 2003]**

Existen otras licencias más o menos restrictivas en cuanto a las libertades de los usuarios sobre el software, pero la más extendida hoy por hoy, y la que ha posibilitado el desarrollo de las amplias comunidades existentes hoy en día ha sido sin ninguna duda la creada por Richard Stallman.

Podríamos resumir el resto de licencias en dos tipos **[Barahona 2004]**:

- Licencias restrictivas o robustas: Estas licencias intentan garantizar siempre los derechos de los usuarios, y por la tanto bajo esta licencias se logra mantener la libertad del código en todo momento, no permitiendo modificaciones que cambien el código a una licencia privativa o comercial. El principal ejemplo de este tipo de licencia es la ya comentada GPL.

- Licencias permisivas: Estas licencias no imponen prácticamente ninguna condición sobre quien recibe el software, y sin embargo le dan permiso de uso, redistribución y modificación. De hecho, bajo esta licencia se permite la redistribución del software bajo una licencia comercial o privativa.

♦ Linus Torvalds empezó a crear el núcleo de un sistema operativo cuando estudiaba en la Universidad de Helsinki. Solicitó ayuda para la creación de un sistema operativo gratuito en un grupo de correo, y en la primavera de 1991, desarrolla un kernel (núcleo del sistema operativo) basado en Unix para computadoras con microprocesadores Intel. Actualmente es el coordinador de las versiones de Linux.

También existen otros modelos, que se basan en la gratuidad del software, pero estos no deben confundirse con el modelo de software libre. Estos modelos sólo participan de la gratuidad, pero no entregan el código fuente:

- Shareware: El creador del software libera una versión que generalmente deja de funcionar en unos determinados días desde su primera instalación en el ordenador, o que tiene restringidas el acceso a determinadas funcionalidades. Si interesa, posteriormente los usuarios deben pagar una licencia, para poder conseguir un versión estable con las funcionalidades completas.

- Freeware: Se libera el software para su utilización y distribución, sin el código fuente. No se requiere un pago por su uso, tan solo a veces se solicita un "donativo" voluntario para el programador. Existen programas liberados como freeware para usuarios particulares, sin intereses comerciales, y vendidos bajo el modelo comercial de licencias si existe un fin comercial en la utilización del producto.

Y además, es muy importante no confundir gratis con libre. Por ejemplo, el navegador web de Microsoft Internet Explorer es un programa gratuito, pero no es libre. El acceso a su código fuente no está disponible para los usuarios.

3.2. Situación actual

Existe numeroso software actualmente distribuido de manera libre ejecutándose en estos momentos en miles de máquinas. El auge de Internet ha favorecido claramente su extensión, al ser distribuidas de manera sencilla. Lamentablemente, la mayoría de las informaciones que podemos escuchar en un noticiario sobre Internet se producen cuando se ha llevado a cabo algún delito con ayuda de Internet o se usa como medio de distribución de material ilícito. Las otras informaciones sobre el uso real, se escriben en páginas de economía que no llegan a la mayoría de las personas.

Pero Internet es mucho más que eso. Existe un gran movimiento de colaboración entre las personas, dónde numerosas personas comparten sus conocimientos sobre cualquier tema imaginable respondiendo a preguntas de manera desinteresada, especialmente alrededor de las numerosas comunidades de software libre que existen hoy en día. Y existen numerosos foros donde debaten verdaderos expertos en cualquier materia imaginable, y que están dispuestos a satisfacer la curiosidad o dudas de cualquiera.

Centrándonos en el tema que nos aplica, la creación del software de manera desinteresada ha crecido en los últimos años. Los programas creados por personas altruistas y de manera desinteresada son utilizados ya por miles de personas y empresas. Estas personas se agrupan en comunidades con intereses comunes, y a veces se crean empresas que hacen negocios con este software.

Podemos ver un ejemplo con el software de servidores web. Un servidor web es un programa básico para la Internet que conocemos actualmente. Es el encargado de servirnos las páginas que pedimos a través de nuestro navegador. La estadística de uso de los servidores web es la siguiente:

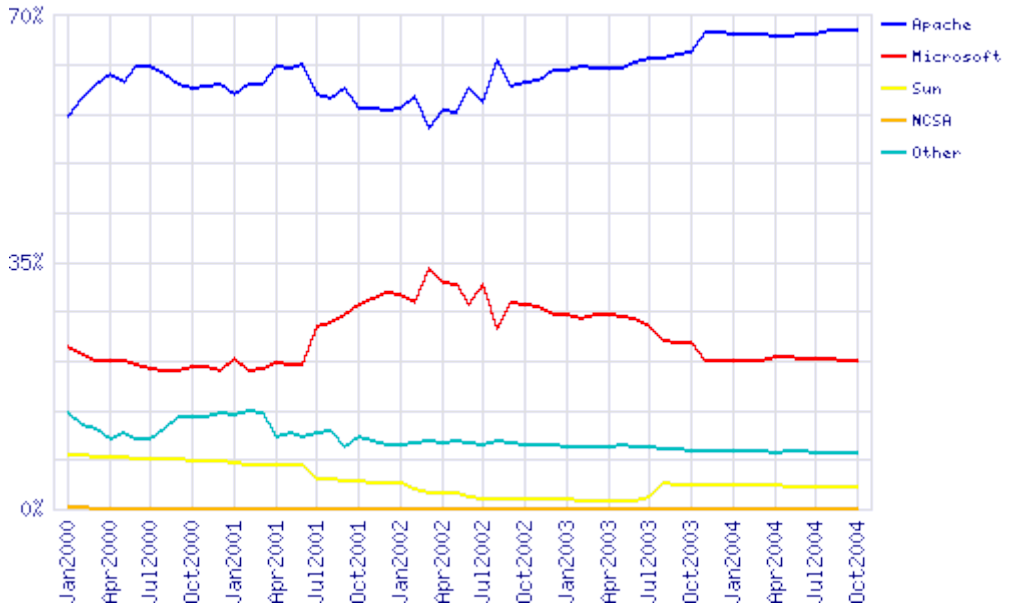

Figura 2: Gráfica de uso de servidores WEB. Netcraft*.

Apache, el servidor web más utilizado en la red, es un software distribuido bajo licencias GPL. Este es uno de los ejemplos más claros en los que la comunidad de usuarios creadores del software ha ganado a las empresas comerciales. Este software se utiliza casi en un setenta por ciento de los servidores actualmente conectados a Internet, casi tres veces más que el segundo competidor, los servidores Web de Microsoft.

Apache (*http://www.apache.org*) es en realidad algo más que un servidor web, es una comunidad de personas dedicadas a crear software de código abierto de manera gratuita. La Fundación de Software Apache (ASF) "*promueve un desarrollo basado en la colaboración y el consenso, ofrece unas licencias de software pragmáticas y abiertas, e intenta crear el software de la mayor calidad.*" Esta fundación tiene numerosos proyectos iniciados, y muchos de ellos consolidados, de software libre, distribuido bajo las licencias GPL o ASL (Apache Software License).

Pero el servidor web Apache no es el único producto de éxito de código abierto: **[Miralles 2003]**

- SendMail: Se trata del sistema líder en administración de cuentas de correo. Una encuesta realizada en el año 2001 atribuía a sendMail la mayor cuota de mercado, con un 42% de todos los servidores de correo, seguido por Microsoft Exchange con un 18% de cuota.

- MySQL: Una base de datos que ostenta el 20% del mercado.Y que según una encuesta de Enero de 2004 ha crecido su cuota un 30% en el año anterior, mientras que las bases de datos de Microsoft (SQL Server) sólo han crecido un 6%. **[Wheeler 2003]**

- PHP: Uno de los lenguajes de programación más utilizados para realizar páginas dinámicas en la construcción de aplicaciones web.

* Reproducido con permiso de www.netcraft.com

- GNU/Linux: Sin duda la gran estrella del movimiento de software libre. Se trata del sistema operativo que más rápidamente ha evolucionado y mejorado de la historia. En un software cada vez más utilizado, sobre todo en servidores en empresas, y especialmente en las máquinas de los servidores web de Internet, donde su cuota alcanza el 70%.

Ilustración 1: La simpática mascota de Linux.

- Navegador Mozilla/Firefox: Ante las continuos fallos de seguridad del Internet Explorer de Microsoft, está este navegador de código abierto del que recientemente se ha lanzado su versión de producción. Ha generado una gran expectación en el mercado, pero todavía está por ver qué cuota de mercado puede conseguir.

Si nos fijamos en el software más utilizado de código abierto, los proyectos de mayor implantación, no son utilizados por un cliente final, sino por empresas que crean sus servicios en Internet, dando acceso a portales de otros clientes o creando sus sitios o aplicaciones, sobre el software libre. Por tanto, los encargados de utilizar estos programas de código abierto suelen ser personas muy al corriente de todos los temas informáticos, y por tanto conocedores del movimiento de software libre. Esto se verá más adelante en profundidad como un punto importante, puesto que una de las dificultades del SCA es llegar al usuario final.

Apache o Linux son unos de los ejemplos más claros de que el software de código abierto puede tener calidad y funcionar mejor que versiones desarrolladas comercialmente por empresas particulares. Pero, por supuesto, que un software sea SCA, no significa que tenga que ser mejor necesariamente que sus rivales comerciales funcionalmente o en calidad, aunque sin duda tampoco lo contrario.

Linux es un ejemplo claro del potencial de las comunidades para la creación del software. Pero creo que es importante observar también que, al igual que el servidor web Apache o el servidor de correo *sendMail*, es un software utilizado por los informáticos para realizar su trabajo. Con esto quiero resaltar el alcance en la sociedad en general de los programas de código abierto. Por el momento el impacto está bastante limitado, especialmente en España, donde la presencia del software libre es prácticamente anecdótica fuera de círculos educativos o puramente en entornos empresariales tecnológicos.

Pero cada vez se oye hablar más de ellos a raíz de las medidas que están tomando numerosas empresas e incluso algunos gobiernos para favorecer el uso de este tipo de software. Hoy existen empresas que han migrado sus sistemas a Linux, y gobiernos que promueven iniciativas para el uso de software libre en su administración, lo que aumentará su difusión cada vez más. El software libre es un movimiento destinado a quedarse, y no hay duda, como veremos más adelante, de que impactará notablemente en el sector.

La pregunta típica que he oído varias veces ante el software libre es: Entonces, si es gratis, ¿cómo se gana dinero con esto? Las personas más integradas con el modelo comercial de las empresas no le ven sentido al movimiento. La contraposición con el software comercial es clara. Existen desarrolladores que no ganan dinero creando software libre, y existen empresas que pagan a sus desarrolladores de software libre y ganan dinero gracias a él, porque se convierten en empresas de servicios más que empresas de venta de productos.

Esto no quiere decir que este nuevo modelo cierre las puertas a las empresas privadas que deseen hacer negocios mediante el software. La cuestión es que se crea un nuevo modelo

de negocio, donde las fuentes de ingreso y clientes van a ser muy distintas a las tradicionales. Conforme vayamos estudiando las empresas basadas en SCA y las características del sector industrial más en profundidad, observaremos que la clave es la migración a una empresa de servicios. Las grandes empresas informáticas hace tiempo que son a su vez empresas de servicios, que proporcionan consultoría, formación y desarrollos a medida. Se debe buscar entonces que más aporta el SCA, como valores añadidos, como por ejemplo, el coste cero de las licencias a los clientes, la posibilidad de la visualización y modificación del código fuente para su adaptación.

Según Eric Raymond (autor de "El caldero mágico") **[Alonso 2002]** la clave del éxito del servidor Web Apache es: "*un ejemplo del que se puede inferir un modelo en el que los usuarios de los programas se benefician de contribuir al desarrollo del código abierto, porque gracias a ello consiguen un producto mejor y a un coste menor del que obtendrían de otro modo.*" Raymond viene a destacar que es posible mejorar el valor que el software aporta compartiendo el trabajo y minimizando costes.

Las empresas basadas en SCA nacientes deben plantearse la estrategia a seguir para asegurar la continuidad y validez del producto que estén creando, teniendo en cuenta la posibilidad de que aparezcan hilos diferentes en su continuidad, al poder publicar cada usuario sus modificaciones sobre el código fuente. La estrategia empresarial puede ser muy complicada de definir si no se controla el producto sobre el que se trabaja, o los servicios que somos capaces de ofrecer si las características modificadas por terceras partes no se controlan.

Una empresa consolidada, que utilice las licencias como fuente de ingresos debe plantearse los cambios estratégicos que le supondrían el pasar su software a código abierto. El movimiento SCA debe ser evaluado en toda su amplitud para asegurar que el impacto en el negocio no va a ser negativo o para descubrir las posibilidades que ofrezca para crear o mejoras las líneas de negocio soportadas. Sin duda, es un cambio que está afectando al sector industrial y las empresas deben plantearse sus repercusiones, si necesitan reaccionar, adaptarse al cambio, o la manera de aprovechar alguna ventaja que les pueda ofrecer.

Actualmente las principales diferencias perceptibles entre los dos modelos de negocio se basan en la contraposición de la venta de un producto contra la venta de un servicio. El coste cero de las licencias de los programas, contra la venta de las mismas, que puede significar un importante planteamiento en las estructuras financieras de las empresas. El hecho de que un programa pueda ser modificado por cualquier usuario elimina la pertenencia del desarrollo del mismo a una sola empresa, y por tanto disminuye el control que tiene sobre él. La continuidad y validez del software no está controlado por una sola empresa, lo que sin duda limita su poder sobre los usuario del software. O desde otro punto de vista, aumenta la libertad de los usuarios con la que lo pueden utilizar.

Las comunidades no suelen ser totalmente caóticas en su funcionamiento. Generalmente existe un líder de proyecto, más por la herencia de su historia, que por tomar posesión activamente del proyecto. Esta persona es la que trata de coordinar los esfuerzos del resto de personas, y decide cuándo liberar una versión al público general. Por ejemplo, Linus Torvalds sigue siendo el líder del proyecto del kernel[♣] de Linux, al ser él su creador inicial.

[♣] Kernel: Es la parte central del sistema operativo. La capa interna más cercana al lenguaje máquina que se ejecuta en un ordenador.

3.3. Un breve apunte sobre la piratería

La cuestión de la piratería es un tema recurrente cuando se habla del negocio del software. Para este trabajo no voy casi a considerar la influencia que tiene en el mercado, para centrarme en el impacto del SCA, pero sin duda es muy alta, y está estrechamente ligada con la estructura del sector industrial. Los productores de software están muy preocupados por ello, e intentan vencerla de forma conjunta formando alianzas.

Por su importancia, quiero dedicarle unas líneas, aclarando la posición de los modelos de negocio comercial y de código abierto.

Actualmente la BSA[♠] cifra las pérdidas en el mercado del software de Europa occidental para los fabricantes locales e internacionales en esta región en más de 8.000 millones de euros, indicando además que el 37% de los programas de software utilizados en la UE es ilegal. La facilidad con la que el software se puede copiar y distribuir obviamente facilita su piratería. En España el índice de piratería alcanza un porcentaje del 44% con unas pérdidas económicas asociadas de más de 421 millones de euros. **[BSA 2003]**

País	Índice de Piratería	Pérdidas Económicas (millones de dólares$)
Alemania	30%	$1.898,9
Austria	27%	$108,6
Belgica	29%	$240,0
Chipre	55%	$7,8
Dinamarca	26%	$164,8
Eslovaquia	50%	$40,3
Eslovenia	52%	$31,9
España	44%	$512,2
Estonia	54%	$14,4
Finlandia	31%	$148,4
Francia	45%	$2.310,5
Grecia	63%	$86,9
Holanda	33%	$576,9
Hungría	42%	$96,1
Irlanda	41%	$70,9
Italia	49%	$1.126,9
Letonia	57%	$16,1
Lituania	58%	$16,5
Malta	46%	$2,1
Polonia	58%	$300,5
Portugal	41%	$65,5
Reino Unido	29%	$1.600,6
República Checa	40%	$105,6
Suecia	27%	$241,1
Región	**37%**	**$9.783,6**

Tabla 1: Piratería de software en la Unión Europea, 2003

[♠] BSA: Bussines Software Alliance. Una alianza entre las empresas más importantes del sector informático para combatir la piratería.

Paradójicamente, la piratería podría ser una ayuda para algunos programas comerciales en la pelea contra el software libre. Existen actualmente miles de personas que utilizan, y tomemos los ejemplos más clásicos y extendidos, Windows y Office de Microsoft sin pagar la licencias, según los datos de la BSA. El hecho de que al final estos programas se obtengan gratis, expande la cuota de mercado de estos programas, aunque no se refleje directamente en las cuentas de ingreso de la compañía.

Si fuese absolutamente imposible poder ejecutar estos programas sin pagar por obtener una licencia, es seguro que muchos usuarios buscarían alternativas más baratas. Si los programas de SCA disponibles les ofrecen una similar funcionalidad y servicio que los comerciales, los clientes ante la posibilidad de tener que pagar licencias podrían pasarse en masa al software libre.

El hecho de que existan mayor número de personas utilizando el software de una compañía, significa que una mayor comunidad de usuarios intentarán usar ese software que conocen y dominan antes de tener que aprender algo distinto.

Obviamente, esto no es tan lineal como lo he expuesto aquí, puesto que la empresa podría reducir el precio, para conseguir más ventas y mantener los márgenes financieros. Pero lo importante que quiero hacer notar aquí, es cómo la piratería en el software ha llegado a ser algo más que ocasional, y que forma parte de las particularidades del sector industrial.

Una empresa no debe fomentar la piratería, pero quizá deba apreciarse el valor que tiene. Intentaré poner un ejemplo más claro. Existen programas que valen millares de euros, de por ejemplo, ingeniería (cálculo de fuerzas estructurales, diseños,...). Si la contabilización de las pérdidas por piratería de la empresa que los desarrolla es la suma de copias piratas que existen, ¿significa eso que todos los que se han pirateado el programa lo compraría si no pudiesen copiárselo? Obviamente no. Puede haber muchas personas que precisamente hayan adquirido ese software porque es pirata, para probarlo o "*jugar*" con él. Por ejemplo, un estudiante puede aprender a manejar el programa si lo obtiene de manera ilícita, pero nunca se gastaría los millones necesarios en comprarlo.

¿Afecta la piratería al software de código abierto? En principio no. Obviamente, un programa que carece de licencias en cuanto a su distribución y modificación, no puede ser "pirateado", puesto que esa acción es absolutamente legal.

La piratería es un hecho ilegal y que debe ser perseguido. La facilidad de crear copias de un software, y el hecho de que siempre se puedan vencer los mecanismos de protección, hace muy difícil su seguimiento. Debe ser perseguida bajo cualquier circunstancia, especialmente aquellas personas que se lucren con la misma. Pero las empresas no deben centrarse en el lado negativo, y tienen que buscar el lado positivo, aunque en su postura oficial quizás nunca lo puedan admitir.

4. Sector Industrial del software

El movimiento de código abierto ha removido en estos últimos años los cimientos del sector industrial del software. En este apartado desglosaremos las principales características del sector industrial y sus empresas, viendo cómo afectan a cada parte la introducción del software libre.

En este sector industrial conviven algunas de las multinacionales más poderosas de la industria, como pueden ser Oracle, IBM o Microsoft, con pequeñas empresas locales enmarcadas en zonas muy concretas, pasando por empresas de mediano tamaño pero que atienden a un mercado internacional. Sin duda el mercado del software es uno de los más globalizados que existen. Internet es la herramienta que ha catapultado esta industria de manera natural, al ser un medio de distribución completo de sus productos para estas empresas. Un pequeña empresa local de software puede vender sus productos en cualquier parte del mundo gracias a la red. La publicidad también se puede realizar a través de Internet, y la distribución y la facturación ya no presentan problemas importantes de seguridad.

Las empresas de software se especializan en algún sector concreto. Las grandes empresas diversifican más sus productos e intentan cubrir el mayor número de sectores posibles. Al ser el software aplicable a cualquier otro sector industrial podemos tener un gran número de empresas en el sector que no compitan entre sí realmente. Hemos de observar que prácticamente no existe en la actualidad ningún sector industrial o de servicios que no utilice en alguna medida un software, de manera que le proporcione mayor productividad o sea la base de la gestión de su negocio.

Algunos de estos sectores más importantes, las principales compañías y el impacto relativo del código libre pueden ser:

- *Sector de los sistemas operativos*: Aquí debemos distinguir dos mundos aparte. El de los ordenadores personales o de escritorio, y el de los servidores. Microsoft es sin duda el gran dominador en los ordenadores personales, siendo *Windows* el sistema operativo más utilizado en el mundo. En el entorno de los servidores hay más competencia actualmente con distintas versiones de UNIX. Es en este segundo entorno donde Linux está penetrando con mayor fuerza.

  Linux ha demostrado su fiabilidad en grandes servidores y aplicaciones de alta disponibilidad, y cada vez es una opción más segura, porque además ya cuenta con el soporte de importantes compañías del mercado, como Oracle o IBM.

  En el mercado de los sistemas operativos para los ordenadores personales la posición de Linux es más débil.

- *Sector de bases de datos*: Oracle e IBM son las grandes dominadoras de este mercado en el entorno empresarial, seguramente con Oracle en primer lugar del ranking, con un 35% del mercado. Microsoft les sigue algo más de lejos.

  Como ya se ha comentado anteriormente, existe una base de datos de código abierto que está teniendo un gran auge los últimos años: MySQL. Está consolidándose como una opción real para muchas empresas. Posiblemente todavía esté muy lejos de las grandes en temas como fiabilidad, capacidad o integridad, pero el hecho es que para muchas aplicaciones se está utilizando porque aporta la funcionalidad necesaria a un coste muy bajo.

- *Sector de aplicaciones de gestión*: Para grandes empresas seguramente SAP sea la herramienta más utilizada. Pero es un mercado muy fragmentado con numerosos competidores, tanto a nivel local como internacional. Generalmente los ERPs de las empresas son más un servicio que una herramienta, puesto que se debe adaptar esta a los procesos internos de la organización.

  Aquí no ha aparecido ninguna herramienta de gestión como software libre que de momento haya destacado. Pero se están desarrollando y pueden en futuro formar parte de la competencia.

  Es de destacar la apuesta que ha hecho la propia compañía SAP por el software libre. Existe una versión de su ERP que funciona sobre MySQL.

- *Sector de aplicaciones ofimáticas*: En este sector tenemos de nuevo a Microsoft como gran dominador. Su suite Office es el producto más usado en todos los ordenadores personales del mundo. Aquí sí ha salido un rival de software libre que es un digno rival, la suite OpenOffice. Esta aplicación fue en principio promovida por Sun, pero tras liberar su código se ha convertido en otro de los mayores éxitos de la comunidad del código abierto.

- *Sector de servidores de aplicaciones*: Los servidores de aplicaciones son las herramientas sobre las que se desarrollan los programas, generalmente se les denomina así a aquellas orientadas a las aplicaciones Internet.

  En este sector hay varios competidores comerciales, y una fuerte competencia del software libre. Existe una empresa dando servicio a un servidor llamado *JBoss* que tiene una importante penetración en los sistemas informáticos a los que va dirigido. Además, el grupo *Jakarta*[2] se encuentra desarrollando otro servidor de aplicaciones de software libre.

- *Sector de herramientas de seguridad*: La seguridad es un tema crítico en la expansión de las aplicaciones sobre Internet. El comercio electrónico debe ser completamente seguro para aumentar su penetración en el mercado, puesto que todavía genera desconfianza en muchos clientes potenciales.

  En este sector existen numerosas empresas, siendo Panda Software, una empresa española, una de las más importantes a nivel internacional.

A pesar de esto, hablaremos del mercado del software de manera global. Por supuesto, todas estas empresas comparten características comunes en su implantación. Y el hecho es, que el software libre afectará allí donde exista a todos los sectores por igual.

Los sectores donde más se ha implantado el SCA son en aquellos dónde los informáticos tienen mayor interés, aquellos dónde encajan las herramientas utilizadas mayoritariamente por los desarrolladores de software. Esto es lógico, un desarrollador de software puede crear herramientas

[2] El grupo Jakarta es un proyecto de la fundación Apache, donde los proyectos se realizan en la plataforma tecnológica de Sun llamada JAVA. Esta plataforma es semi-libre, puesto que aunque es gratuita, SUN se reserva con su licencia el desarrollo de la misma.

para informáticos puesto que posee todos los conocimientos necesarios, pero por ejemplo puede ser más costoso que se reúnan un número suficiente de personas con conocimientos de software e ingeniería aeronáutica para crear un programa libre para cálculo de trayectorias de cohetes.

Sin embargo, el movimiento de software libre es realmente vasto, y no se puede dudar de que al ritmo actual se acabarán cubriendo las necesidades de casi todas las empresas o particulares. De hecho, el movimiento libre anima a que si no lo encuentras como software libre... ¡lo crees tú mismo y lo compartas! Obviamente para esto necesitarás unos conocimientos informáticos que no están al alcance de cualquiera.

El sector industrial se ha visto sacudido por el auge de nuevos modelos de negocio, que cambian la relación con los clientes, los servicios debidos a los mismos y la manera de generar ingresos y beneficios **[Alonso 2002]**. A continuación expondremos las principales características del sector industrial, buscando el impacto que los nuevos modelos basados en SCA tiene sobre ellas.

Veamos ahora la definición del sector industrial y las implicaciones que tiene el software de código abierto en él.

4.1. Determinantes estructurales de la competencia:

Como vamos a determinar, la estructura del sector industrial del software se va a ver sometida a fuertes tensiones y cambios por el SCA. Las empresas que se encuentran en este sector, deben por tanto, prepararse para el cambio que esto va a suponer. De hecho, ya hay numerosos movimientos anticipándose, preparando la empresa para las nuevas posibilidades, nueva competencia y dificultades. Se deben evaluar las oportunidades y amenazas que este movimiento va a generar.

Las empresas existentes deben estudiar en todo momento cómo se encuentra el sector del software para responder con rapidez a las amenazas, o aprovecharse de las ventajas que les pueda suponer. Y no cabe duda que el software libre va a ser una amenaza en muchos aspectos para algunas empresas, y una gran ventaja aprovechable en otros.

Veamos cuáles son los impactos del software libre en cada una de las características estructurales del sector industrial:

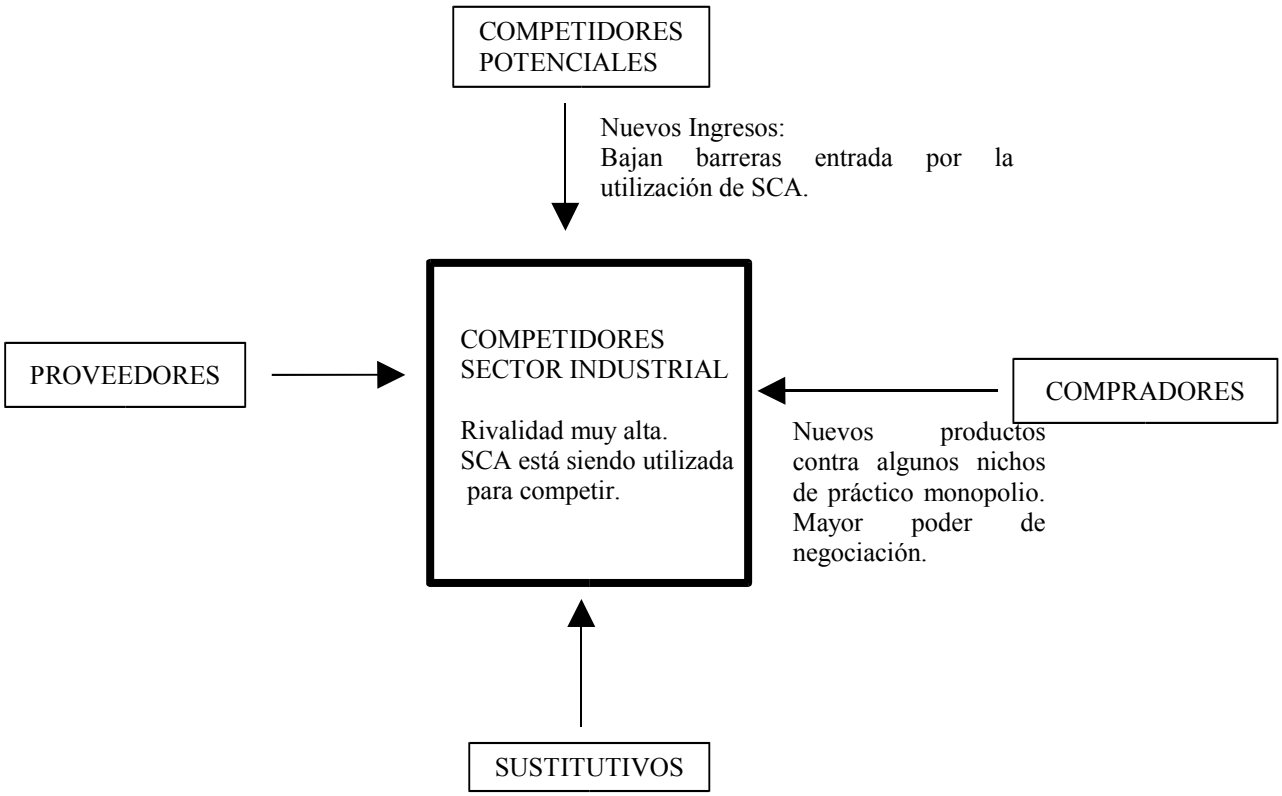

Figura 3: Determinantes de la estructura del sector industrial.

4.1.1. Amenazas de ingreso

Esta característica creo que es una de las más fuertemente afectada por el movimiento de código abierto.

La posibilidad de utilizar de manera legal productos ya desarrollados y en continuo mantenimiento elimina importantísimas barreras de ingreso. Se reduce de manera drástica los requisitos de capital. Una empresa no necesitará partir de cero desarrollando su propio software. No necesitará un determinado tiempo para la creación de un producto o preparación de los servicios añadidos que proporcione. La utilización de software libre va a proporcionar a las empresas una base de la cual partir a coste prácticamente cero.

Las empresa pueden crear sus productos o servicios basándose en SCA, se elimina el costo de las licencias necesarias de otro software del que dependan. Una nueva empresa actualmente puede utilizar todo software libre para su mantenimiento, y para crear productos o dar soporte sobre ellos.

En estas empresas que proporcionen SCA no existe problema de acceso a los canales de distribución gracias a Internet. Esto no es propiamente por el hecho de ser SCA, sino por ser una empresa tecnológica de Software, deberá tener una importante imagen en Internet. Pero poner un portal en Internet es hoy más sencillo que nunca. Ahora se puede crear una imagen empresarial accesible desde cualquier sitio del mundo en cuestión de días, e incluso en cuestión de horas.

La reacción esperada de los competidores ante un nuevo ingreso posiblemente dependa en gran medida del sector de introducción. Las grandes empresas no se van a molestar por pequeñas empresas que inicien una actividad local, basándose en servicios a las empresas que utilicen software libre. En caso de que la nueva empresa logre una diferenciación importante, teniendo en cuenta el rápido crecimiento del mercado, es más que posible que no se espere ninguna reacción contraria importante.

Pero las compañías que ataquen directamente a los productos comerciales ya establecidos, si pueden esperar reacciones más importantes. Aquí podemos ver las empresas de distribución de Linux que se han creado en los últimos años. Estas sociedades establecen la competencia a los sistemas operativos comerciales, introduciéndose en el mercado sin haber tenido que desarrollar el sistema desde la nada.

Como ejemplo de crear nuevas barreras de ingreso podemos ver también las demandas existentes sobre sistemas de software libre como Linux. La compañía SCO, distribuidora de una versión de UNIX, y recientemente adquirida por Microsoft parte de su capital, mantiene varios litigios contra empresas distribuidoras de Linux (RedHat, por ejemplo) porque afirman que Linux utiliza código plagiado de su versión del sistema operativo. Esto puede disuadir a nuevas empresas a entrar en el mercado hasta que quede claro la resolución de estas demandas.

El SCA también limita la experiencia como barrera de ingreso. El hecho de que el SCA proporcione la libertad e incluso la documentación, para poder utilizar unos determinados programas, hace que el tiempo de inicio hasta ser productivos se minimice. La empresa es capaz de conseguir sus primeros objetivos en un menor tiempo. Las nuevas tecnologías nuevas son abiertas y accesibles de manera libre. Además el uso de estándares en el software libre facilita la adopción del mismo, puesto que más personas conocerán sus características.

4.1.2. Rivalidad entre competidores:

En el mundo del software existe una situación algo particular: la posición cuasi-monopolística de Microsoft en algunos sectores, y algunas de sus políticas de competencia, ha fomentado un ambiente de “todos contra Microsoft”. No es el tema de esta tesis desarrollar de qué manera Microsoft ha llegado a la situación en la que se encuentra, pero el hecho es que actualmente se enfrenta a algunas demandas anti-monopolio y al sentido rechazo de numerosas personas en el mundo del software. Sin ninguna duda Microsoft puede ser una de las empresas más afectadas por el SCA. De hecho, ha iniciado algunos movimientos contra el mismo.

Pero la rivalidad entre los competidores empresariales existentes también es muy alta. Da la impresión a veces que se trata de un “todos contra Microsoft”, y grandes empresas como IBM, Sun u Oracle están apostando fuertemente por dar apoyo a los sistemas de código abierto (con Linux como su mayor exponente) para perjudicar a Microsoft, disminuir su dependencia de la misma y aumentar sus cuotas de mercado.

El SCA está generando numerosos movimientos entre las empresas rivales en el mercado. En el mercado del software domina claramente Microsoft en el ámbito de Sistemas Operativos y programas de uso generalista. Esta empresa ocupa más de un 80% en los sistemas de escritorio de los ordenadores instalados. Actualmente parece que empieza a tomarse en serio la amenaza que le puede suponer los movimientos de SCA. Numerosos

dirigentes de Microsoft han hablado contra el SCA, o menospreciando su valor en la industria **[DiarioTI 2004 (3)]**. Estos indicadores llevan a pensar en los movimientos que pueda realizar Microsoft contra este tipo de software: bajada de precios, desacreditación, mejora de imagen pública,...

Las reacciones de Microsoft son muy interesantes, y se podrían clasificar en dos tipos. Por un lado hablan de que la competencia es buena en el mercado, y que no se oponen a la misma. La consejera delegada de Microsoft Ibérica **[Garcia 2002]** dice que "*es una buena noticia [...] A nosotros nos obliga a superarnos y al final, los grandes beneficiados van a ser siempre los usuarios*". Y, añade, pide que el mercado "*se guíe por criterios técnicos y no políticos*". De esta manera intentan destacar sus ventajas técnicas y hacer ver que sus soluciones son las mejores frente a la competencia.

Pero por otro lado, existen otras declaraciones de dirigentes de Microsoft, como Steve Ballmer **[DiarioTI 2004(1)]**, presidente ejecutivo, en las que amenaza veladamente a los usuarios de Linux, con que se pueden encontrar con demandas por la utilización de Linux por el caso de SCO comentado anteriormente. Además dicen que este nuevo modelo de negocio puede acabar con la riqueza que genera el modelo actual. Es decir, que intentan frenar con amenazas y miedos el desarrollo de sus competidores de código abierto.

El hecho es que Microsoft ya está reaccionando. Ha anunciado la liberación de parte de su código fuente de Windows a determinados organismos "*para que puedan compartir la innovación*" **[DiarioTI 2004 (2)]** que producen, y se presenta inevitable una bajada de precios, que ya están haciendo cuando ven perder las ventas. Y es que en los últimos meses han sido continuos los anuncios de empresas importantes o gobiernos anunciando la migración de sus sistemas a software libre. Y aunque el volumen de estos cambios no sea todavía importante para la empresa, está claro que son el inicio de una tendencia que parece acentuarse con el paso del tiempo y la estabilización del software de código abierto.

Pero hay muchas otras empresas en dura pugna por conseguir mayor cuota de mercado en sus respectivos mercados.

La entrada de nuevos competidores basados en SCA puede suponer renovar el dinamismo de las batallas por los clientes. Aunque las empresas creadas a nivel local parece que no encontrarán mucha resistencia de estas grandes empresas, al final en su conjunto representarán una importante cuota de mercado.

4.1.3. Poder negociador de los compradores

Por supuesto, a mayor oferta, mayor campo de elección tienen los consumidores con el que presionar a las compañías. El caso más claro en este punto son los sistemas operativos en los ordenadores personales. Existe prácticamente un monopolio de Microsoft con su sistema operativo Windows, pero Linux, de código abierto, está entrando con fuerza, y da poder negociador a los clientes.

Por supuesto, la creación de nuevos productos con ventajas competitivas sobre los actuales, pueden dar mayor poder a los compradores. Cuantas mayores opciones de compra dispongan, mayor poder negociador dispondrán.

En el mercado de los sistemas operativos, por ejemplo, distingamos dos sub-mercados. El de los sistemas operativos de escritorio o "work-station", y el de servidores centrales o "host". En el primer caso, Windows es el líder indiscutible, y Macintosh era la única competencia que le arañaba una porción significante. Actualmente, Linux empieza a poder presentarse como una baza para los escritorios de los trabajadores (aunque ya veremos más adelante cuales son sus puntos débiles). La estrategia de Microsoft debe ser aprovechar la ventaja de cuasi-monopolio y tecnológica, puesto que mi impresión es que Linux va por detrás, imitando al líder. Pero si Linux ya es una posible amenaza, será la primera alternativa seria que se planteen muchos clientes para cambiar de Windows y no pagar coste de licencias (aunque deberían estudiar todos los costes ocultos, ver punto *1.1 Evaluando los costes de la tecnología*)

En el mundo de los servidores "host" Sun o IBM copaban la mayor parte. Parece que es en este campo donde Linux se ha presentado como una alternativa real y muy competitiva. De hecho, algunos de estas grandes empresas ya han empezado a dar servicio de Linux como plataforma para sus productos. Los clientes se han encontrado con una posibilidad más con sus ventajas de SCA.

El SCA está aumentando la competencia en determinados segmentos del software, especialmente en el mundo de servidores. Esta competencia será buena para los consumidores, porque significará mayores esfuerzos de las compañías por ofrecer mejores productos y servicios.

4.1.4. Poder legislador de los gobiernos:

Los gobiernos se están moviendo últimamente a favor del software de código abierto, donde existen numerosos apoyos para que las instituciones utilicen este tipo de software y promuevan su uso. En este caso, las razones aducidas son más políticas que técnicas: la independencia de las empresas, la portabilidad y reutilización de los programas entre las administraciones,...

Existen numerosos movimientos en las administraciones para la adopción de software libre, un ejemplo es el *Linex*: una versión de Linux distribuida por la Comunidad de Extremadura, para fomentar el uso de este tipo de software. Esta comunidad lo está implantando en sus sistemas, empezando por la red educativa. Esta adopción de Linux ha tenido gran repercusión en los medios, al tratarse de uno de los primeros pasos en una administración pública para la implantación total de soluciones basadas en software libre. Más adelante, veremos las implicaciones de este tipo de software libre para las administraciones públicas.

Sin embargo, en mi opinión, los gobiernos no deberían legislar a favor de un tipo de software u otro en las empresas privadas. Una cosa es que tengan decisión sobre el software que quieren usar en sus sistemas, y otra es que decidan sobre los sistemas en las empresas privadas aduciendo alguna razón social. Al fin y al cabo, si lo hacen, estarán beneficiando a una serie de empresas que dan soporte a los programas SCA, que pueden tener una serie de ventajas, pero dañarán la competitividad necesaria para el desarrollo del mercado, su tecnología y su crecimiento.

Richard Stallman es una gran defensor del software libre como mejora social, puesto que da la libertad a los usuarios en la utilización del mismo. Personalmente no comparto la

opinión que todo el software deba ser libre, y menos aún por tanto, que la legislación deba ayudar a ello.

Creo que una empresa debe tener el derecho de vender sus programas bajo las licencias que crea conveniente. El mercado decidirá si su producto merece la pena como para pagar lo que demanden. Es posible que en unos años solo haya software libre, pero debe de ser decisión del mercado actuando libremente y no algo impuesto por un gobierno.

4.1.5. Poder de negociación de los proveedores:

Los proveedores de las empresa de software son otras empresas de software, o empresas de hardware. Quizá el poder de los proveedores pueda estar bastante limitado en este sector industrial. El hecho de que exista tanta competencia entre las empresas productoras de software, y muchas sean proveedoras de otras, limita el impacto.

Hay un caso especial en el que el código libre puede ser una trampa para su utilización únicamente en el software libre. Una comunidad de usuarios puede desarrollar un software libre para su utilización sobre un sistema operativo comercial, o utilizando una librería[♠] de una empresa privada que no se distribuya bajo la licencia GNU-GPL. Richard Stallman es muy crítico con esta manera de actuar, puesto en realidad no se podría utilizar el software sin haber pagado antes por una licencia del programa, de la librería, que necesita para funcionar.

En este caso, la empresa comercial de software puede ser un proveedor de alguna manera con expectativas de ventas gracias al software libre.

4.1.6. Productos sustitutivos

No parece hoy por hoy que exista un producto sustitutivo al software. Quizás volver a un mundo sin ordenadores ni robots, pero no parece muy probable. Esto, como veremos posteriormente, tiene una alta importancia para la definición del ciclo de vida de un producto software.

4.2. Estrategias competitivas

El análisis estructural anterior del sector industrial debe de ser clave para las formulaciones de las estrategias competitivas de las empresas en el sector. Conocer estos análisis es fundamental para la formulación de los puntos fuertes y débiles de la empresa, ayudando a determinar las oportunidades o amenazas que se presentan.

Las acciones defensivas u ofensivas que tomen las empresas del sector, dependerán en gran medida de su posición actual y su estrategia empresarial futura. Los posibles enfoques generales que pueden tomar:

[♠] Librería: Software que proporciona unos determinadas funcionalidades que pueden ser utilizadas por otro software.

1) **Posicionamiento**:

La empresa buscará los lugares del sector industrial donde las fuerzas del cambio sean más débiles, o pueda crear defensas ante ellas. Para ello es muy importante conocer las capacidades de la empresa, y cómo pueden hacer frente a las causas de la estrategia competitiva.

Por ejemplo, una empresa que produzca aplicaciones de gestión, que tenga un buen departamento de Marketing que conozca al cliente y adapte rápidamente sus productos a sus necesidades, puede potenciar este elemento en particular, puesto que el SCA puede pecar en gran medida de falta de contacto con el cliente, si no hay detrás una empresa que guíe su desarrollo.

Sin embargo un desarrollo de SCA debería buscar inicialmente aquellas posiciones donde el valor añadido por los servicios al producto sea menor, para competir con producto únicamente. Los desarrollos SCA más exitosos hoy por hoy son los utilizados por los propios informáticos. Estos desarrollos atacaron una posición del sector industrial donde, sin darse cuenta, mejor posicionados estaban. Los clientes son expertos en el manejo de estas herramientas, y muchos de ellos colaboran en su desarrollo. Los servicios de apoyo al software se los proporcionaban mutuamente de manera colaborativa.

2) **Cambiando el equilibrio**:

Esta sería una reacción ofensiva para competir contra el SCA. Se trata de alterar los orígenes de las fuerzas competitivas.

En este sentido hay empresas que están intentado tomar medidas mediante las patentes de software. Intentan impedir la utilización de determinadas características del software por los proyectos de SCA. Si una empresa consigue una importante patente de software, las comunidades o empresas que la necesiten para desarrollar el SCA deberían pagar por ello, con lo que de alguna manera se eliminaría las libertades bajo las que se debe distribuir ese software, y se estarían creando barreras de ingreso **[Choike 2004]**.

Debemos tener en cuenta que los programas son tan solo la plasmación formal de las ideas de los programadores, convertidas en órdenes para las máquinas. Si esto se pudiese patentar, estaríamos patentando ideas. Los derechos de copia, o copyright, existentes hasta el momento son suficientes para proteger los derechos de las compañías y los programadores.

En mi opinión las patentes de software son una equivocación que puede perjudicar mucho a la innovación en la industria del software. Se ha demostrado hasta ahora que el modelo comercial puede crecer gracias a la protección de los derechos de autor. Las patentes de software consisten en patentar ideas sobre las que se desarrollan los productos. Las compañías más grandes pueden patentar miles de ideas, para negociar posteriormente con ellas. Las nuevas empresas deberían pagar por usar esas patentes, con lo que realmente sería muy difícil conseguir entrar en el mercado, especialmente para los proyectos basados en software libre.

3) **Tomando ventaja del cambio**:

La evolución del sector industrial se produce de forma natural. En este caso la introducción del SCA parece ser un punto de inflexión en su evolución, pero realmente se ha ido introduciendo poco a poco hasta tomar fuerza. Las empresas deben evaluar el futuro de las fuerzas estructurales y los cambios que pueden soportar.

Una empresa comercial puede crear su software libre si tras el estudio del futuro de las fuerzas competitivas observa que su rentabilidad puede caer si no lo hace. Una empresa líder en un nicho concreto del software, puede decidir cambiar su modelo de ingresos y beneficios y dar el software libre, si con ello paraliza los proyectos nacientes de competencia en SCA. De esta manera se podría esperar que la comunidad se vuelque con su proyecto del que es líder en el segmento, y se las pueda arreglar para ser el sitio de referencia para ese producto. Por supuesto dejará de ingresas beneficios por licencias, pero puede crear sistemas de distribución y servicios.

Otras empresas comerciales, como IBM u Oracle están dando soporte a los principales productos libre, como GNU/Linux. Con ello intentan dañar la posición de su competidor más fuerte: Microsoft. De esta manera aprovechan y refuerzan el movimiento SCA para aumentar la competencia en los sectores donde sus competidores son más fuertes.

4) **Estrategia de diversificación**:

El análisis estructural pormenorizado del sector industrial puede llevar a descubrir aquellos negocios infravalorados actualmente en el mercado. De esta manera la empresa puede diversificar sus productos o servicios antes de que otros lo hagan y ganar terreno.

La mayoría de empresas intentan cubrir varias necesidades de las empresas relacionadas o de los clientes. De esta manera no dependen únicamente de un solo producto que podría crearles problemas. Por ejemplo, la compañía Oracle ha creado alrededor de su motor de Base de Datos numerosas aplicaciones variadas para los clientes: ERP[♣], CRM[♦], herramientas de "Data Mining",...

Clay Shirky, experto consultor en tecnologías de Internet y profesor adjunto de la Universidad de Nueva York escribe **[Shirky 2002]**: "*Hace 30 años, cada departamento de tecnología en este país estaba en el negocio de crear productos propios, y la industria del software creció bajo esa suposición. Ahora, el software de código abierto sugiere un modelo de casi servicio puro, donde la funcionalidad básica no cuesta nada, y todo el dinero se encuentra en la personalización*."

El mercado del software se está orientando a los servicios, y esto lo estaba haciendo antes incluso del despegue del SCA. Cada vez más, las grandes y medianas empresas de software proporcionaban un mayor número de servicios: consultoría, formación, desarrollo de

[♣] Enterprise Resource Planning: Aplicaciones de "Planificación de los Recursos de la Empresa" para gestionar de manera global y centralizada los recursos de la organización.

[♦] Customer Relationship Management: Aplicación de "Gestión de las Relaciones con los Clientes" para gestionar todo el ciclo de vida de las relaciones con los clientes.

aplicaciones... La venta de las licencias de sus productos dejó de ser la única fuente de ingresos para estas empresas.

En principio puede parecer que las empresas que ofrecen SCA como sus productos, pueden presentar una clara estrategia basada en costes, puesto que no pueden cobrar licencias por el uso de los programas que distribuyen bajo la licencia GPL. Sin embargo, la orientación de la mayoría de las empresas hacia los servicios, hace que el dinero que se gasta en licencias una empresa comparado con el costo total del sistema de información sea cada vez más reducido.

Por supuesto que hay empresas que compiten en costes y otras en diferenciación dentro del sector industrial. Indudablemente el tema de la competencia por costes se ve favorecido por la utilización de software libre. La elección de esta estrategia probablemente dependa del tipo de cliente que se quiera o se pueda obtener:

- Los pequeños negocios que buscan mejorar su sistema de información, o la aportación de nuevos valores añadidos a su empresa mediante la tecnología, posiblemente no tengan restricciones en cuanto a la tecnología que pueda ser empleada. En estos casos la empresa que le ofrece una solución tecnológica puede disponer de más libertad para la utilización del software de código abierto y disponer de esta manera un mayor margen de beneficio o una reducción del precio de venta.

- Las grandes empresas que buscan soluciones informáticas de gran tamaño para sus problemas, en los que puede ser que en gran medida descanse su organización, se centrarán más en la diferenciación y la confianza de los productos que en el coste. Además, las empresas que llevan un mayor tiempo funcionando suelen tener de partida algún requisito tecnológico para sus nuevos proyectos, por la existencia de sistemas legados.

El software libre puede llegar a demostrar su fiabilidad y desempeño (y desde luego lo ha hecho ya en muchos proyectos), pero todavía existe en las empresas la reticencia a utilizarlo por la falta de un responsable único de los productos, entre otras cosas. La confianza es un punto muy importante para la toma de decisiones de los directivos en las cuestiones estratégicas.

Una de las ventajas de las empresas comerciales que deberían aprovechar es el contacto con el cliente. Puede parecer una incroguencia, puesto que en el modelo de software libre, el cliente puede incluso modificar a su antojo el programa, pero presumo que no es así. La mayoría de los clientes no estarán preparados para poder hacerlo, o pueden no querer compartir con la competencia el valor añadido que le darían al software. Posteriormente se estudiará con mayor profundidad las diferencias en las estrategias de desarrollo entre los dos modelos. Cada uno cuenta con sus ventajas y desventajas.

4.3. Indicadores del mercado

El mercado del software proporciona numerosas noticias y movimientos empresariales que estudiar. Se producen numerosas declaraciones de altos cargos empresariales, anuncios de productos nuevos o actualizaciones, compras y ventas de empresas,... incluso hay varias demandas antimonopolio juzgándose en estos momentos.

Los numerosos movimientos lo delatan como un sector industrial muy dinámico. Es un sector creciente en el que todas las empresas quieren aparecer como líderes tecnológicas. Ya hemos hablado de las numerosas declaraciones de Microsoft respecto el daño que puede causarles el software libre. Las demandas hacia los usuarios de Linux por parte de SCO son otra prueba de que las empresas comerciales están realmente preocupadas.

Pero no solo Microsoft da indicios de que le preocupa en el sector. También al resto del mercado le preocupa su posición dominante, especialmente en Europa, donde se encuentran juzgándose varias demandas antimonopolio y por abuso de poder en el mercado por parte de la Unión Europea.

Las empresas comerciales actualmente están a favor del uso de patentes del software. Se trata de poder patentar las maneras de desarrollar un software. En Europa existe un debate abierto actualmente sobre este tema **[COM 2002]**, en el que la comunidad de software libre está claramente posicionada en contra. Sin embargo algunas empresas lo solicitan como defensa de sus intereses. Personalmente, no me gustan las patentes de software por que es como patentar "ideas", y no es trasladable al mecanismo de patentes actual, donde patentas inventos. En mi opinión, los "copyright" deberían ser suficiente para salvaguardar los intereses de las empresas.

Las patentes de software pueden ser muy perjudiciales para la competencia incluso entre las propias empresas comerciales. Podría parar en seco la innovación que ha hecho del software uno de los mercados emergentes más importantes en los últimos años.

4.4. Evolución del Sector Industrial

El software libre crea movimientos que afectan a todos los elementos de la estructura del sector industrial. Este hecho va representar un momento importante en la evolución del sector industrial. Es importante por tanto que las empresas del sector asuman que el software libre va a producir un cambio en las mismas bases del sector, de manera que las reglas del negocio van a verse afectadas. Cada empresa debe analizar su posición en el sector, para observar como esta evolución importante en el sector le va a afectar.

Las limitaciones del ciclo de vida del producto como pronosticador de la evolución de un sector industrial se vuelven en este caso todavía más importantes. El ciclo de vida de un producto software es distinto al de un producto estándar. Si hemos admitido que por el momento no hay productos sustitutivos, es de suponer que no hay etapa de declive en su ciclo de vida. Pero hay una importante diferencia entre el software y otros productos. El software no se estropea, no se degrada, no se agota por su uso. Quizás los medios de almacenamiento físico se puedan estropear, pero el hecho de que se puedan hacer copias a coste prácticamente nulo también implica la casi ubicuidad del software.

Pero si observamos más el ciclo de vida de un producto software y el funcionamiento del mercado, podemos llegar o observar otro motivo que puede llevar a la fase de declive del software: la completa satisfacción de los usuarios con el mismo. Puede parecer una incongruencia, pero el hecho de que los clientes estén satisfechos con los productos, va a significar el inicio del declive del producto. Para entender esto, debemos explicar una cuestión importante en el desarrollo de productos software por la mayoría de las empresas: las versiones.

Cuando se lanza un producto software nuevo al mercado se suele encontrar en una fase en la que todavía no está completamente desarrollado su potencial. Las limitaciones de tiempo que impone un mercado tan dinámico y competitivo son muy altas, y las empresas deben restringir las funcionalidades añadidas para poder lanzar el producto cuanto antes. Un buen departamento de marketing y enfoque orientado a la calidad podrían determinar cuáles son las importancias relativas para los clientes de cada funcionalidad.

Pero la limitación de tiempo implica otra cuestión si cabe más crítica, los posibles fallos no solucionados en el producto final. Actualmente se da por hecho que el software contiene "bugs" o fallos, unos más importantes y otros más insignificantes. Las empresas lanzan versiones sucesivamente, de manera que añaden la funcionalidad que faltaba y corrigiendo los fallos encontrados, numerosas veces por los mismos clientes.

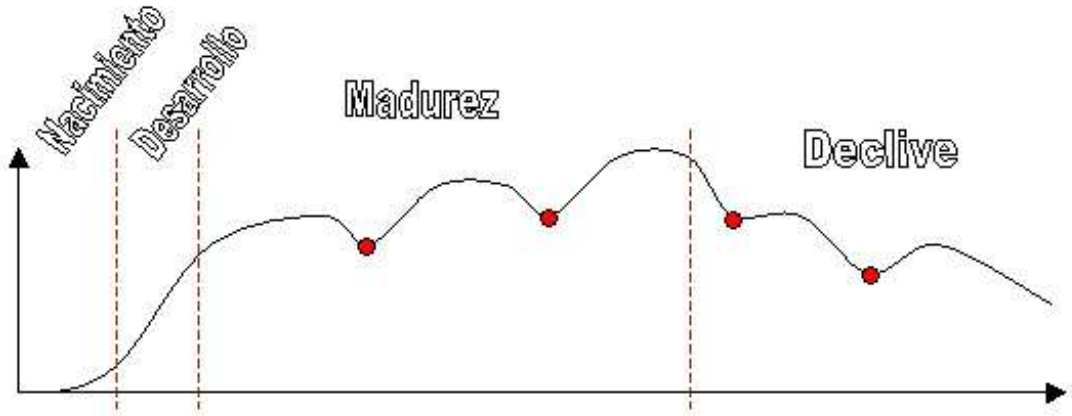

Ilustración 2: Ciclo de vida de un producto software

Pero las empresas se están cansando de actualizaciones forzosas que en realidad no les aportan ningún valor añadido. **[Spolsky 2001]** Cuando un producto está consolidado, y los usuarios obtienen el máximo rendimiento que esperaban de él, es difícil convencerles de que deben actualizarse a la siguiente versión. Mientras se han ido corrigiendo fallos y aumentando su funcionalidad de una manera importante, las versiones eran vendidas sin problemas. Pero después, ¿por qué actualizarse? Cada vez el beneficio comparado con el costo es menor.

Un método que usan algunas compañías es el eliminar el soporte de versiones con más de un tiempo determinado, forzando a sus clientes a la actualización. Pero las protestas de los clientes no tardan, cuando la nueva versión no les aporta nada necesario ni satisfactorio, y el desembolso económico es importante.

La fase de declive es distinta también a los productos tradicionales. Los clientes podrían seguir utilizando un producto durante mucho tiempo, aunque no se actualicen de versiones. Un producto de software obviamente no se dejará de utilizar a no ser que aparezca otro competidor mejor, pero la empresa productora no obtendrá los mismos ingresos que en las fases anteriores.

Se une a esto que los tiempos del software como negocio seguro se han terminado. Cada vez el riesgo es más alto, las expectativas más bajas y los retos más altos. Un ejemplo de esto ha sido la decisión de Microsoft de pagar a sus accionistas 3$ por acción como dividendo

especial. Significa que por primera vez no ha encontrado cómo invertir semejante cantidad de efectivo, puesto que el mercado se ha ralentizado**[Carr 2004]**.

Quizás el mercado del software se encuentre más cerca del sector maduro que de sus inicios espectaculares de rendimiento y ganancias. Los rendimientos ya no son tan elevados, el crecimiento empieza a no ser tan vertiginoso, y los retos para generar un éxito empresarial cada vez son más elevados **[Carr 2004]**.

5. La empresa basada en software libre

Sergio Montoro, CEO de Hipergate* distingue entre fabricantes, vendedores de servicios y clientes **[Montoro 2004]**. Para los primeros, la clave consiste en crear una comunidad de usuarios lo suficientemente amplia como para generar comunidad; para el vendedor de servicios, las ventajas de la utilización de software libre son diferentes (precio, disponibilidad de código fuente así como formación); por último para el cliente final necesita tener confianza en que la empresa que le ofrezca el servicio.

Según esta apreciación obtendríamos dos tipos de empresas basadas en el SCA como núcleo de sus operaciones. La primera serían las empresas que centralizan la gestión de un producto de código abierto, en su línea más importante, distribuyéndolo y dándole servicio de soporte. Las segundas serían aquellas empresas que utilizan programas de SCA como plataforma para crear sus propios productos o servicios que les den valor añadido a los clientes.

O sea, que según la posición en la cadena de valor del software libre podríamos valorar distintos modelos de negocio en los que pueden basar sus ingresos: **[Miralles 2003]**

1) Empresas que distribuyan el software libre: Aunque el software libre es por definición también gratuito y puede ser conseguido de diversas maneras, una empresa puede crear su propia versión y ofrecer servicios añadidos. El ejemplo más claro son las distribuciones de Linux. Este sistema operativo requiere una compleja instalación, y numerosas descargas desde Internet de sus partes. Existen empresas que han creado su propia versión facilitando la instalación y administración de Linux, ofreciendo así prestaciones adicionales con las que hacen negocio. La empresa RedHat es un ejemplo de esto.

   Generalmente, este tipo de empresas también ofrece servicios de formación o consultoría, pues cuenta con personal cualificado.

2) Empresas que den soporte, consultoría o formación al software libre: Obviamente el software libre también necesita de profesional cualificado y de un soporte adecuado que responda ante las necesidades de los clientes.

   Las empresas distribuidoras son las que dominan el mercado de la consultoría y asesoramiento o formación, pero también grandes empresas como IBM o Sun han ampliado sus departamentos de consultoría con estas tareas y software.

   A niveles locales están surgiendo un importante número de empresas donde equipos bien formados en tecnologías de código abierto dan servicios y realizan desarrollos a medida de los clientes.

3) Empresas que se apoyen en programas de SCA: Por ejemplo, determinadas empresas renuevan sus productos para que funcionen sobre plataformas de código abierto para independizarse de empresas proveedoras **[Miralles 2003]**. Además de esta manera pueden aumentar sus márgenes, y competir en aquel mercado dónde sus posiciones son más ventajosas.

* www.hipergate.org: La empresa Know-Gate es el principal patrocinador de este producto, que es de código abierto. Esta empresa es un ejemplo de una empresa suministradora de servicios sobre un producto de software libre. Proporciona servicios de instalación, soporte, *hosting*, consultoría y personalización.

Por ejemplo, *IBM* ha apostado fuertemente por el software libre, ha invertido más de 1.000 millones de dólares en la migración de sus aplicaciones y la formación de sus técnicos, contando en la actualidad con más de 5.000 clientes sobre plataforma Linux.

Otras empresas, como *SAP* y *Oracle* han adaptado también sus aplicaciones para Linux, de manera que también intentar fomentar la competencia contra Microsoft.

E incluso empresas proveedoras de hardware como *Hewlett-Packard* o *Dell Computer* han establecido alianzas con distribuidoras de Linux para ofrecer soluciones integrales a sus clientes.

4) Empresas que utilizan programas SCA como usuarios finales, pero que pueden en cualquier momento aportar modificaciones a la comunidad.

   Estas empresas no generan ingresos por el software libre, pero lo utilizan porque les aporta un mayor valor añadido, o bien por que aumenta sus márgenes operativos. También podrían en cualquier momento aportar un valor añadido al software libre, realizando las modificaciones necesarias para ellas en el software y posteriormente compartiéndolas con la comunidad.

5) Empresas de venta de accesorios para el software libre: Todo este movimiento ha generado un negocio satélite de venta de libros, CDs o documentación.

Por tanto tenemos varias posibles líneas de negocio creadas en base al software libre. Estas empresas basarán sus ingresos en que el software libre funcione de manera correcta y se adapte a los requerimientos de los clientes. Todas las empresas colaborarán por tanto en un desarrollo del código fuente que será abierto incluso para sus clientes como demanda la licencias GNU-GPL.

Pero además, numerosas personas participan en los desarrollos de código abierto por un interés propio débil, al menos visto exteriormente. Las personas que dedican su tiempo a la colaboración en un proyecto de código abierto, pero no pertenecen a ninguna empresa que les proporcione una fuente de ingresos por ese trabajo deben ser muy tenidas en cuenta. De hecho, estas personas son las que han creado las comunidades y han dado el valor que tiene al software libre.

Las comunidades de software libre son dominadas por personas que actúan de manera individual, pero que buscan, en general, el beneficio de la comunidad. Obviamente, cuando un producto avanza, y llega a su fase de producción, genera el interés de las empresas. Entonces el proyecto podría ser dominado por alguna de ellas, pero debería ser crítico el no eliminar el poder de la comunidad sobre el desarrollo del producto.

La posibilidad de modificación libre del software, y la competencia entre empresas, podría conducir a varios hilos de desarrollo incompatibles entre sí. Esto sería un grave problema para el producto y sus usuarios, que se verían desconcertados. Por tanto, el espíritu de comunidad debe estar presente, y debe llevar a la empresa a no descuidar este mismo perfil en sus trabajadores, a fomentar la cooperación.

Las empresas pueden crear el software y distribuirlo como SCA, desarrollar sus productos sobre éste o dar servicios basados en él. La no pertenencia a una empresa del código proporciona

nuevas maneras de crear valor sobre este. Pero las empresas propietarias del software, generalmente "validan" otras empresas para que estas sean "partners" certificados, es decir, que demuestran su conocimientos de las herramientas propietarias y colaboran con ellos directamente. Esto puede dar cierta garantía de eficiencia a los clientes, contra las empresas que den soporte o servicios sobre software de código abierto, a las cuales nadie controla ni establece su posición.

Veamos qué roles podemos describir en los modelos comerciales y de software libre:

- Software Comercial:

  - Empresas de desarrollo, distribución y servicios: En el modelo comercial son estas las empresas que crean los productos, y generalmente, las únicas que pueden modificarlos puesto que el código fuente es de su propiedad. Por tanto, el software está bajo su control y estrategia. Son las únicas responsables del contacto directo con el cliente y de su adaptación del software para que proporcione el máximo valor añadido a los clientes.

  - Empresas de servicios sobre software comercial: Estas empresas crean desarrollos y dan servicios sobre software comercial, que no pueden modificar ni distribuir libremente. Existe aquí el concepto de "partner", que es una empresa certificada por el fabricante para dar servicios relacionados con sus productos.

  - Usuarios de software comercial: Los usuarios finales compran el software y lo utilizan, pudiendo emplear algún servicio añadido de las empresas anteriores. Cuando necesitan nueva funcionalidad, deben esperar a una nueva versión del fabricante. Cuando encuentran un fallo o "bug" deben solicitar una corrección también al fabricante.

  - Empresas clientes de software comercial: Las empresas compran el software y lo utilizan para dar valor añadido a sus productos o servicios. Estas empresas generalmente requieren los servicios de alguien que les respalde en la utilización del software, tanto en formación como generalmente en servicios de consultoría o desarrollo de productos propios. Como usuarios, tienen el mismo problema con los fallos y las nuevas funcionalidades.

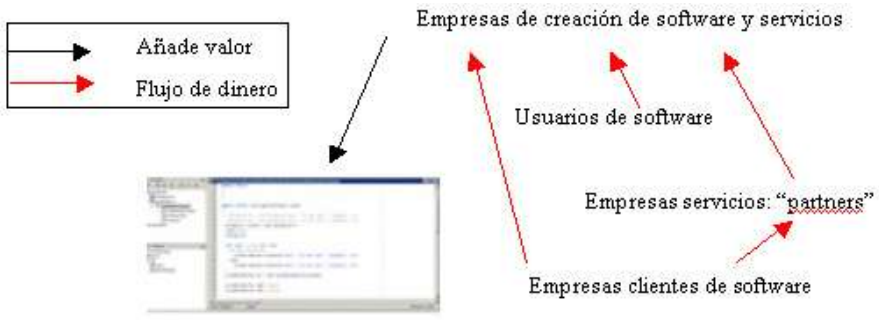

Figura 4: Modelo de Negocio comercial.

En este punto es importante observar que la única posibilidad de mejora del producto viene por parte de la empresa creadora y poseedora del código fuente. Esta empresa debe velar por que el producto o los servicios sean lo que los clientes necesitan. Un buen departamento de marketing y una

orientación estratégica hacia los clientes o la calidad les proporcionará la base para descubrir los requisitos demandados y satisfacer a sus clientes.

Veamos ahora los roles en un modelo de código abierto:

- Software libre:

  - Desarrolladores "libres": Esta es una "especie nueva", génesis del movimiento de software libre. Son personas que de manera altruista* colaboran en la creación de un producto. Muchas veces son iniciadores en solitario de un proyecto por interés personal, lo que luego da lugar a una comunidad de usuarios con los mismos intereses compartidos.

  - Empresas de distribución, desarrollo y servicios: Existen empresas dedicadas a distribuir el software libre en soporte físico. Además generalmente estas empresas crean una línea del software con su marca comercial. El ejemplo más claro de esto son las distintas distribuciones de Linux que existen: RedHat, Mandrake, SuSe.

  - Empresas de desarrollo y servicios sobre software libre: Estas empresas son el equivalente al modelo comercial, pero dan soporte y realizan desarrollos sobre el software de código abierto, para proporcionar a sus clientes las ventajas que de ellos puedan derivarse.

  - Usuarios de software libre: Los usuarios finales del software libre disponen del código fuente para poder corregir los "bugs" o añadirle funcionalidades o cambiarlas libremente.

  - Empresas clientes de software libre: Las empresas que deciden la utilización del software libre tienen las mismas necesidades de crear valor añadido a sus productos, y necesitan también generalmente una empresa que les dé servicio de formación o consultoría. En estos casos, además, disponen del código fuente para modificarlo a sus necesidades.

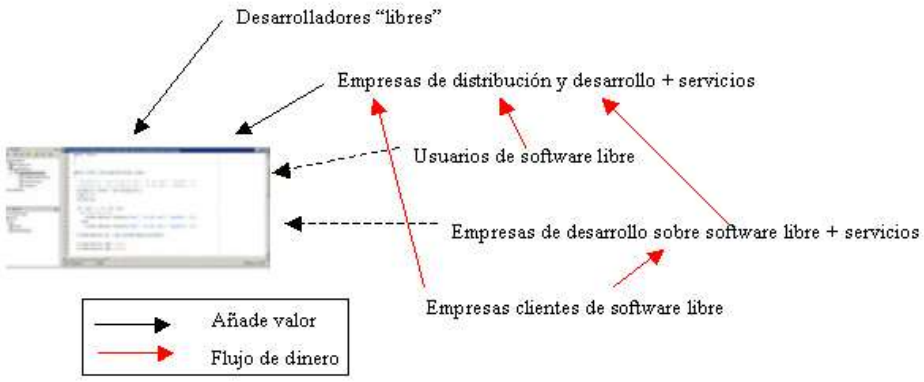

Figura 5: Modelo de negocio basado en software libre.

* No voy a entrar a valorar las motivaciones psicológicas de estas personas, pero se valoran razones tan distintas como generosidad o afán de notoriedad. Lo que es cierto es que numerosas personas dedican su tiempo no sólo a crear el software libre, si no además a dar respuesta al resto de usuarios sobre el software. Los foros en Internet de los programas de SCA están llenos de gente que dedica su tiempo a contestar las dudas de cualquiera **[Lakhani 2003]**.

¿Cuáles son entonces las diferencias más importantes entre los dos modelos? y, ¿cuál es mejor? Sin duda estas preguntas son las que se realizan numerosos directores de departamentos de tecnologías o líderes de proyecto, que deben establecer la estrategia más favorable para la empresa.

Observando las diferencias entre los dos modelos podemos observar que la más importante son las flechas que significan las partes que pueden proporcionar valor añadido al software. En un análisis sencillo, esto es una ventaja clara. Cuantas más partes puedan asegurarse la modificación y mejora del código, más posibilidades habrá de que éste sea mejorado y adaptado a las necesidades reales del mercado.

Pero se debe ver en mayor profundidad si esta ventaja es real, o no tiene por qué ser mejor que la del modelo comercial. En el modelo comercial, la empresa tiene el control sobre el código. Una empresa orientada al cliente, y que haga un buen uso de la gestión de la innovación y la calidad, debe poder garantizar que refleja las necesidades de los clientes y asegura su confianza en el software. El modelo basado en software libre puede crear la aparición del caos en la línea estratégica del producto, que más probablemente, no existirá, puesto que cada parte tendrá sus intereses en que siga una línea de desarrollo u otra.

De todo lo anterior podemos resumir que las empresas en el mundo del software de código abierto pueden generar ingresos por varios métodos, por los cuales existirían clientes dispuestos a pagar un precio:

- Distribuciones con valor añadido como facilidades de instalación o administración.
- Servicios de formación.
- Servicios de consultoría y desarrollos personalizados sobre software libre.
- Servicio de soporte, mediante contratos de asistencia y resolución de errores, o soportes telefónicos u online.
- Ventas de productos relacionados: libros, documentación, objetos de marca,...

Podemos plantear un análisis de Debilidades, Amenazas, Fuerzas y Oportunidades (DAFO) donde concretemos más algunos aspectos que deben tener en cuenta las empresas situadas en este sector industrial, y que se planteen el negocio del software libre: **[Mitre 2001]**

Fuerzas	Debilidades	Amenazas	Oportunidades
I+D realizada por voluntarios	Falta de propietario del producto	Riesgo de fragmentación:	Conectividad por Internet
Lanzamientos de versiones frecuentes	Difícil de originar un proyecto	Necesidad de control de versiones	Estructura de soporte competitiva
Desarrollo y pruebas paralelas	Menor usabilidad*	Falta de aplicaciones compatibles	
Cultura de comunidad, solidaridad			
Accesibilidad del producto asegurada			
CARACTERÍSTICAS DEPENDIENTES			
Masivo número de programadores expertos		Número de distribuidores	
Estructura de líderes aceptada		Nacimiento de muchas nuevas compañías "start-up"	
Código maduro			
Procesos de desarrollo			

Tabla 2: Tabla de Fuerzas, Debilidades, Amenazas y Oportunidades que representa el software de código abierto en el mercado del software.

1) Fuerzas:

- I+D realizada por voluntarios: La labor realizada de manera voluntaria por muchas personas significaría miles de millones de euros para las empresas del sector. Los desarrolladores libres a menudo modifican el código en su interés propio, pero por consiguiente se convierte en una mejora para los demás.

  A veces las empresas del sector de software libre ofrecen recompensas a estos programadores para la mejora de versiones, para mantener los desarrollos activos e interesantes para los clientes.

- Lanzamientos de versiones frecuentes: Generalmente las versiones de los programas de código abierto son liberadas con un menor lapso de tiempo, debido a las frecuentes integraciones que se realizan por una heterogénea masa de programadores.

- Desarrollo y pruebas paralelas: El modelo de desarrollo que utiliza el software libre permite a grupos o personas trabajar de manera independiente, cada uno trabajando en partes distintas del software.

  El desarrollo paralelo de las funcionalidades permite escoger posteriormente la mejor implementación de las funcionalidades. Y el paralelismo en las pruebas

* Usabilidad: La usabilidad son técnicas que ayudan a los seres humanos a realizar tareas en entornos gráficos de ordenador.

aumenta el rendimiento de las mismas, que redunda en un menor tiempo de corrección de fallos.

- Cultura de comunidad, solidaridad: La comunidad permite la realización de un trabajo una única vez, en vez de que cada empresa implemente lo mismo varias veces, con la consiguiente pérdida de trabajo. Desde una vista económica global, se elimina la pérdida asociada al trabajo duplicado.

- Accesibilidad del producto asegurada: Un producto de software libre con el suficiente interés en la comunidad, no podría ser eliminado del negocio en un corto plazo de tiempo. Una empresa comercial podría decidir que no le interesa para su estrategia seguir una determinada línea de productos y discontinuarla bruscamente.

  Además, el hecho de la posesión del código fuente por parte de los clientes permitiría un mantenimiento interno, y se podría continuar la vida del producto indefinidamente.

2) Oportunidades

- Conectividad por Internet: La comunidad se comunica a través de Internet. Las listas de correo, chats, grupos de noticias y sitios web permiten el crecimiento de esas comunidades, permitiendo que exista un desarrollo y mantenimiento las 24 horas del día los siete días de la semana.

- Estructura de soporte competitiva: El software comercial depende de un monopolio, en cuanto a que el código fuente no está disponible. Cuando un producto de software libre supera el umbral crítico de usuarios, el soporte que puede dar cualquier empresa es el mismo, y por tanto la competencia aumenta, lo que beneficiará al mercado.

3) Debilidades

- Falta de propietario del producto: Limita la posibilidad de determinar con garantías cual será el futuro estratégico del producto. Una compañía comercial podría comprometerse a una determinada compatibilidad, y se le podría demandar si no la cumple, pero en un proyecto de código abierto no se puede garantizar nada. Muchos directores de los departamentos de tecnología confían más en las empresas, que en un movimiento dirigido por una comunidad, puesto que pueden poner nombre a las responsabilidades ante los problemas.

- Difícil de originar un proyecto: Se debe lograr una masa crítica de buenos desarrolladores interesados en el proyecto para que este pueda "despegar". Los inicios de muchos proyectos son debidos a una motivación personal de un programador, y debe suscitar suficiente interés para poder crear una comunidad que de respaldo al software.

- Menor usabilidad: Generalmente los proyectos de software libre son desarrollados por programadores que crean las cosas para la gente de tecnología, que como ellos tienen unos conocimientos adquiridos altos de software. Esto lleva a que se despreocupen de la usabilidad, del interfaz con el usuario. No existe la

preocupación por los posibles clientes menos técnicos, ni una estrategia para estudiarlos y planear estrategias de marketing.

4) Amenazas:

- Riesgo de fragmentación: Ocurre cuando se rompe el código fuente en varias líneas de desarrollo. Esto puede ocurrir por que distintas personas tomen un camino distinto en el desarrollo, de manera que cada una crea su propia versión del código base compartido. Esto puede ocurrir por que los desarrolladores no alcanzan un acuerdo sobre el camino a seguir, o por que los intereses de las partes implicadas son muy distintos.

- Falta de aplicaciones compatibles: Los proyectos de código abierto tienen limitaciones cuando aún no son aplicaciones extendidas en la comunidad informática. Hasta que un programa no se reconoce como exitoso, pocas personas crean herramientas para él.

- Necesidad de control de versiones: El código base debe ser puesto en un repositorio común, y cada cambio debe ser integrado y probado antes de lanzar cada versión.

5) Otras:

Según el momento en el que se encuentre el proyecto de software libre, pueden ser una amenaza o una oportunidad, o una fuerza o una debilidad. En realidad estas características también afectarían al análisis anterior, pero las características anteriormente descritas son más intrínsecas a un producto de código abierto, independientemente de su estado y evolución:

- Masivo número de programadores expertos: Los inicios del software de código abierto son controlados por pocos desarrolladores que tienen un interés especial en desarrollarlo. El número no aumenta significativamente, generalmente hasta que el proyecto ha conseguido lanzar alguna versión más o menos exitosa. Es una variable que puede ser una fuerza si el número de desarrolladores es muy amplio, o una debilidad, si carece de los programadores suficientes para controlar el producto.

- Estructura de lideres aceptada: Generalmente los proyectos suelen tener una jerarquía implícita entre sus desarrolladores. La fuerza que pueda ejercer debe ser vista como un arma de doble filo: puede restringir demasiado el desarrollo del proyecto, o llevarlo sin ningún control, de manera que no lleve la mínima línea coherente entre sus versiones.

- Código maduro: Una aplicación se considera madura tras cuatro o cinco años de desarrollo. Es importante evaluar la situación de los proyectos bajo esta perspectiva, para averiguar su situación en el ciclo de vida. **[Spolsky 2001]**

- Procesos de desarrollo: El sistema "caótico" de desarrollo en las comunidades se basa en generar más iteraciones del producto, más integraciones con un mayor número de versiones. Esto puede tener la ventaja del lanzamiento más rápido de correcciones o nuevas funcionalidades, pero se puede perder la visión global del

producto y producir una estructura más inestable por construir el software a base de pequeñas partes superpuestas.

- Número de distribuidores: El número de distribuidores facilitará la integración de las herramientas y la personalización. Pero la falta de los mismos limitará las opciones del software libre para su implantación.

- Nacimiento de muchas nuevas compañías "start-up": El hecho de que la tecnología se puede adquirir a un precio nulo, aumenta las perspectivas de creación de empresas, que no deben invertir prácticamente en el producto. El inicio de un producto de software de código abierto puede ser una oportunidad para colocarse estratégicamente, pero después podría ser una amenaza para las empresas establecidas.

Vamos a ver algunos puntos en los que estos modelos de negocio plantean diferencias a las emrpesas, y cómo una empresa puede generar ingresos con el software libre. La estrategia de la empresa de software debe cambiar para adaptarse a un mercado en evolución, evitar sus amenazas y aprovechar las oportunidades.

5.1. Ventajas competitivas:

Las empresas deben crear ventajas competitivas que les proporcionen mayores beneficios a sus clientes. Cada modelo de negocio tiene sus ventajas competitivas sobre el otro. Cada empresa o usuario debe evaluar cuáles son las ventajas más interesantes para su negocio.

La empresa creadora de software que se plantee sacar sus productos bajo una licencias GPL de manera libre, debe valorar la ventaja competitiva que le supone esto ante la posible pérdida de ingresos por la cuestión de la venta de licencias. Y del mismo modo, la empresa que haya creado un producto software, debe estudiar el rendimiento financiero que puede obtener de las licencias y si guardarse la pertenencia del código le va a facilitar el contacto con el cliente y su fidelización.

Como ya se ha comentado anteriormente, el coste puede ser la primera ventaja competitiva del software libre. Una buena proporción de los clientes de software libre lo serán por que el coste de obtención y uso del software va a ser nulo. Es gratis descargarse de la red muchos programas que pueden competir con sus homólogos comerciales. Existen determinados sectores de la demanda que serán fácilmente atraídos por esta ventaja, donde el precio de adquisición supone un factor decisivo. Por ejemplo, los usuarios particulares, el de la educación, grandes organizaciones que solo requieren un escritorio de trabajo en miles de puestos,...

Pero esta ventaja importante cuenta con unos condicionantes **[Abella 2004]** importantes:

- Desarrollo y aplicación de leyes de competencia y monopolio: Las leyes para el fomento de la competencia deberían obligar a que ningún proveedor realizase practicas comerciales abusivas. A Microsoft se le ha acusado numerosas veces

de utilizar su monopolio "de facto" en los sistemas operativos para aumentar su cuota en otras areas.

- Presión contra el software propietario pirata: Como se ha comentado en el apartado 1.1 Un breve apunte sobre la piratería, la presión contra el software pirata obligaría a los usuarios del mismo a buscar otras soluciones, posiblemente buscando el mismo coste, o sea, nulo.

- Aparición de una masa crítica de usuarios: Cuando el número de usuarios crezca, el interés por el mercado crecerá, pero si hay una demanda pequeña, los fabricantes no crecerán. Este círculo vicioso, se romperá al alcanzar un número de usuarios crítico, dependiente en cada nicho de mercado.

- Regulaciones europeas antimonopolio: Basada en la primera opción, actualmente la Unión Europea está en trámites de obligar a Microsoft a no utilizar su posición dominante en el mercado de sistemas operativos.

Pero esta ventaja muy importante del coste se ve además también muy afectada por dos cuestiones muy importantes: Primera, que en los entornos empresariales el precio de las licencias puede no ser muy importante en relación al coste total de las tecnologías de información, y que los usuarios todavía encuentran dificultosa la instalación de numeroso software libre.

Otra ventaja competitiva del software libre es que generalmente adopta estándares públicos definidos y por tanto facilita la integración entre los sistemas. Esto puede ser especialmente importante en las administraciones públicas, que deben interaccionar con muchos sistemas externos a ella.

El hecho de que el software libre sea proporcionado con el código fuente puede ser su ventaja competitiva más importante. Las empresas no tienen por qué depender ya de un único suministrador de software, e incluso pueden realizar ellas mismas cambios sobre el software para adecuarlo a sus necesidades.

Pero esta ventaja puede tener importantes limitaciones desde el punto de vista de algunas empresas. Muchas de ellas no están preparadas, no tienen el tiempo o no quieren realizar el esfuerzo que supone la comprensión de un programa, porque para ello deberían dedicar unos recursos que quizás no posean o no dispongan de tiempo. La otra limitación es la imagen de marca que tiene un producto de software libre, la desconfianza que pueda generar el que no exista un responsable único del producto, y se confíe en la historia pasada de las empresas comerciales. El hecho de que las empresas dispongan del código fuente de su software no significa que lo vayan a utilizar, ni siquiera a mirar.

Otra punto que puede ser visto como una ventaja competitiva es el respeto a los estándares. Generalmente existen una serie de estándares, definidos por organismo internacionales, para la definición de determinados protocolos en el software. La comunidad de software libre es partidaria de implementarlos siempre como base de su compatibilidad con el resto de los programas desarrollados.

Cuando se desarrolla utilizando software propietario es muy común encontrarse posteriormente con problemas a la hora de migrar de plataformas, puesto que se han podido utilizar implementaciones propias del suministrador, y que cambian de un fabricante a otro.

Esto impide la independencia del suministrador, y numerosas veces las empresas intentan ligar por ello a los clientes con soluciones propias y de las que tienen la única implementación existente.

La calidad de los programas de SCA es referida por numerosos autores de artículos como otra ventaja competitiva más. Este punto necesita un mayor análisis, y es desarrollado en el punto 1.1 Calidad, donde se comparan las implicaciones de la calidad en el software libre.

5.2. Estrategia empresarial

La empresa puede y debe definir claramente su misión y estrategia. Una empresa que dé soporte o servicios sobre software libre generalmente también producirá modificaciones en el mismo, compartidas por toda la comunidad. El problema puede surgir cuando la evolución del software de manera externa a la empresa no sea el esperado o planificado en la estrategia de la empresa.

La estrategia empresarial de una empresa comercial puede estar basada en la confianza de que el producto va a seguir la línea definida por las estrategias y tácticas creadas por el comité de dirección y las líneas de mando. Pero la ventaja de que cualquiera puede aportar código a un producto libre puede convertirse en una trampa para la estrategia de la empresa que apoyaba otra línea de actuación en un proyecto de código abierto.

Un frase muy reveladora de la situación que se produce aquí es la pronunciada por Robert Young, fundador de RedHat[*]: "El competidor también es colaborador". La estrategia empresarial de una empresa basada en código abierto no puede olvidar nunca esto. De hecho, debe contemplar la libertad del software como algo a promover, ya que es la idea principal del movimiento. Cualquier cambio o limitación a las libertades de los usuarios, seguro que sería muy mal recibido en la comunidad, con el consiguiente perjuicio para la imagen de la empresa, y por tanto para sus negocios. Por esto, la empresa debe creer realmente en las ventajas del software libre como mejora para los usuarios, o empezaría traicionando su propia estrategia que debe ser definida defendiendo el movimiento de software libre. Cualquier mejora en el software producida debe ser compartida a la comunidad, y por tanto las estrategias de la empresa no pueden basarse en cuestiones que sean contrarias a las cuatro libertades de los usuarios definidas por la licencia GPL.

La estrategia de comienzo de una empresa basada en software libre puede ser el cubrir un nicho nuevo o poco explotado **[Martínez 1999]**. Pero para ello es crítico la consecución de un número elevado de usuarios y desarrolladores, para conseguir una comunidad suficientemente amplia para llegar a producir una demanda de servicios y productos relacionados con esa tecnología abierta.

Una estrategia que intente cubrir un nicho poco explotado en el software puede llevar a crear un software de código abierto nuevo, o a explotar los servicios que se pueden proporcionar sobre uno ya existente. La diferencia fundamental será la comunidad de usuarios y desarrolladores, que en un caso se debe conseguir que nazca, y en el otro se deben obtener los beneficios de la ya existente.

[*] RedHat es una de las distribuciones de Linux más populares.

La estrategia de una empresa basada en software libre debe tener muy en cuenta dónde se encuentra su posición dentro de la comunidad de usuarios de los productos de código abierto que distribuya o de servicio. Podíamos concretar que una máxima de cualquier empresa basada en código abierto debe ser crear confianza entre la comunidad.

Cualquier sospecha de que una empresa puede estar intentando beneficiarse con prácticas poco éticas que vayan contra las licencias GPL puede hacer que los desarrolladores libres, que trabajan por su cuenta, dejen de hacerlo al no querer verse involucrados en un negocio empresarial. Además, las empresas que obtuviesen servicios de la empresa, podrían retirarnos su confianza si habían escogido el software libre para garantizarse la independencia del proveedor.

Pero ¿deben las empresas comerciales adaptar su estrategia al software libre? Sí, sin ninguna duda. Dependiendo del sector en el que se encuentren del mercado, es posible que el software libre sea una mayor o menor amenaza, o incluso de momento ni siquiera se lo planteen así. Pero como hemos visto en los cambios que se van a producir en las estructuras del sector industrial al que pertenecen, la estrategia empresarial debe adaptarse porque va a significar un cambio en la evolución del sector. Por eso es muy importante que cada empresa ajuste su estrategia, basándose en el análisis de los cambios en el sector:

- Analizando las amenazas que se presentan
- Analizando las oportunidades que puedan significar los cambios
- Analizando sus fuerzas en el sector y en la organización.
- Analizando las debilidades de la organización ante la nueva situación.

Con los análisis correctos, la empresa podrá llegar a discernir la estrategia más adecuada para adaptarse a la evolución del sector industrial.

Una pregunta obligada que debe hacerse es si debe cambiar su modelo de desarrollo a código abierto. Esta es sin duda una cuestión estratégica muy importante. La liberación del código de un software propietario va a tener importantes repercusiones en la compañía y sus clientes. La compañía Sun acaba de anunciar que va a lanzar la nueva versión de su sistema operativo Solaris 10, bajo una licencia de código abierto **[Infoworld 2004]**. La estrategia puede ser obtener el visto bueno de las comunidades de software libre, para que Solaris se convierta en un rival en el mismo terreno que Linux. Si ya su producto copa los primeros puestos en servidores empresariales (ver punto 1.1 Las empresas cliente), el hecho de lanzarlo como código abierto elimina las ventajas estratégicas de Linux, su verdadero competidor en este segmento.

Esta compañía puede perder el control de las versiones futuras de Solaris, pero puede beneficiarse también de la ventaja de contar con un número extra de desarrolladores. De todas maneras, todavía la empresa no se ha pronunciado bajo qué tipo de licencia de software libre va a publicar sus códigos. Es más que posible que sea algún tipo de licencia más restrictiva que la licencia GPL.

Cada empresa debe determinar si la estrategia de adaptación al nuevo modelo de negocio es la más adecuada para su organización. Debe estudiar en profundidad las debilidades, amenazas, fuerzas y oportunidades que se le pueden presentar, y que ya tiene

continuando su política actual en el mercado. Algunas preguntas que debería hacerse para intentar obtener información adicional pueden ser:

- ¿A qué se deben los ingresos? ¿Mayoritariamente a la venta de licencias o a los servicios?
- ¿La competencia de software libre es técnicamente superior?
- ¿Existe una importante comunidad de usuarios de software libre en el nicho de mercado en el que estoy localizado? ¿Continúa creciendo?
- ¿Puede estar pensando la competencia comercial dar el paso a código abierto antes que nosotros para ganarse a la comunidad?
- ¿A qué ritmo ganan cuota de mercado las soluciones de código abierto?
- ¿Cuál es la imagen de la empresa en el mercado?

Obviamente, muchas empresas optarán por seguir con su modelo comercial. Muchos de los productos que se comercializan no tienen rival en el mundo del software libre, o pueden haber necesitado una inversión inicial muy alta, que se debe recuperar por medio de las licencias. Otros productos software pueden necesitar muy poco servicio posterior a los clientes, y por tanto no ser rentable una empresa que únicamente se dedique a servicios sobre su cartera actual de productos.

5.3. Gestión de los RRHH

El desarrollo del software libre puede plantear nuevos retos a la gestión de los recursos humanos de las empresas. Este desarrollo tan especial del código abierto puede provocar que personas que no se conozcan y ni siquiera trabajen en la misma organización (¡e incluso puede que sean de la competencia!), deban coordinarse y colaborar en la mejora del mismo software.

No cabe duda de que esto es algo realmente novedoso. Hasta ahora las empresas incluso firmaban acuerdos de confidencialidad con sus trabajadores, para intentar mantener como propias las ventajas en innovación o investigación y desarrollo. El software libre contraviene absolutamente estas maneras de trabajar y funcionar. La clave ahora es compartir las mejoras y el código. Los trabajadores de una empresa deben colaborar, y por tanto conocer y relacionarse, con personas totalmente ajenas a la organización.

Estos requisitos obligarán al departamento de recursos humanos a cambiar muchas de las maneras establecidas para el trabajo de desarrollo, y especialmente deben potenciar otra serie de habilidades en los trabajadores, como la comunicación y las relaciones sociales. Sin embargo, no deben permitir que las relaciones con el exterior hagan olvidar la misión y la estrategia de la empresa. Los trabajadores deben estar formados para reaccionar de manera independiente, más que como meros seguidores de órdenes de sus superiores.

La misión y la estrategia de la empresa a nivel global debe ser comprendida por cada trabajador, y deben saber cómo hacer que sus actuaciones sigan la línea indicada para que todos colaboren en una dirección y sentidos únicos: los dictaminados por el comité de

dirección. Numerosos desarrolladores van a verse implicados en contactos con el exterior de la empresa. Por ello, se debe implicar a toda la fuerza de trabajo en los objetivos de la organización. De otra manera, se puede dar una imagen cambiante y de poca preparación ante los clientes y el resto de la comunidad.

Si el empleado va a estar muy relacionado con el exterior lo deseable sería que este fuese maduro y estuviese motivado para representar de manera adecuada a la empresa y velar por sus intereses. De otra manera podía ser muy difícil llevar una conducta directiva en todo momento en que el subordinado vaya a relacionarse con el entorno exterior de la empresa. La conducta directiva de los superiores implicaría un mayor control sobre los mismos, y por tanto una mayor necesidad de mandos superiores para controlarlos.

Es muy importante favorecer la comunicación horizontal, porque los cambios pueden sucederse a gran velocidad de manera externa a la empresa, y especialmente el departamento de marketing debería conocerlos de inmediato. El estilo de dirección debe ser muy abierto a la comunicación entre los empleados, sin burocracias exageradas ni jerarquías. La estrategia de mando debería intentar estar basada una conducta de apoyo.

Los coordinadores internos deben tener muy clara la misión de la empresa para orientar los esfuerzos externos en la medida de lo posible a su cumplimiento. Pero la empresa debe tener una postura clara. Cualquier intento de manipulación de la comunidad de desarrolladores libres, creará sin duda una reacción adversa. Se debe fomentar la cooperación a todos los niveles, interna y externa. No se debe olvidar que es la comunidad la que da la importancia al software libre, y cuanto más amplia sea la comunidad de usuarios y desarrolladores será mejor para la situación del producto, y por tanto de los servicios añadidos sobre él.

Y el departamento de recursos humanos será quien deba escribir las estrategias de captación de desarrolladores voluntarios. Para ello, además de que el producto sea útil y cubra sus necesidades y expectativas, se deben utilizar técnicas de habilidades sociales, relaciones públicas, captación del interés,... para mantener los intereses de los colaboradores externos. Por tanto, el departamento de recursos humanos ya no va a estar únicamente centrado en lo que ocurra dentro de la empresa, si no que debe tener en cuenta que también se trabaja con personas que no pertenecen a la misma. Se deben coordinar también para evitar que creen nuevas ramas del producto **[Alonso 2002]**.

La interactuación con demasiada gente externa puede crear problemas, de descoordinación y gran pérdida de tiempo. Para ello se deben poner las herramientas de automatización que sean necesarias. Las más comunes en los desarrollos de software libre son:

- Listas de correo controladas para facilitar la comunicación: Suelen existir al menos un par de listas, una para la comunicación entre los usuarios, y otra para la comunicación entre los desarrolladores.

- Repositorio centralizado de código: El código fuente se centraliza en un repositorio común[♠] donde se integran todos los cambios y líneas de código.

[♠] CVS: Control Versioning System. Sistema de repositorio centralizado para controlar las versiones sucesivas del código fuente. Es un sistema de software libre.

- Herramientas de control de bugs♦: Dónde se controla las mejoras y correcciones que se van necesitando en el software.

El departamento de Recursos humanos debe tener muy en cuenta por tanto que existe gente externa a la organización que va a cooperar con los trabajadores. Una diferencia importante será si somos los depositarios del código fuente, y somos quien administra como líderes el proyecto, o es externo a la empresa y nos limitamos a contribuir a su desarrollo.

En el caso de que seamos los líderes del proyecto deberá existir cierta jerarquía determinando un coordinador del mismo. El coordinador debe resolver los problemas que puedan existir entre los trabajadores y los programadores externos a la empresa. Y aunque los trabajadores deben participar de la manera más independiente posible, como representantes de la empresa en el exterior, será el coordinador el que vigile que se tenga una única línea argumental o de desarrollo.

5.4. Gestión de la innovación

En el mercado del software la innovación tiene una importancia vital. Es muy importante que la empresa este basada en una cultura con sólidos cimientos en la innovación. Es realmente sorprendente cómo el software libre puede ser respaldado por los factores básicos de la cultura occidental descritos por Samuel P. Huntington:

- Igualdad de las personas en cuanto a su dignidad: El software libre iguala tanto al productor como al consumidor en la utilización del software.

- Libertad: Es la base del movimiento, donde se dispone de toda la libertad para cualquier uso, copia o modificación del software.

- Participación: El software libre realmente permite participar en el desarrollo a cualquier persona.

- Solidaridad: El software libre además es gratuito, y facilita la utilización de cualquiera que lo necesite.

- Subsidiaridad: El productor del software no puede tomar ninguna decisión que podría perjudicar a los usuarios, puesto que el software en realidad no le pertenece.

Los principios de creatividad, conocimientos, eficiencia y valores humanos descansan sobre estas bases culturales. Estos principios, que deben estar presentes en todas la empresas, parece que se ven más respaldados todavía en este modelo nuevo de negocio, al cumplimentar los factores básicos de la cultura occidental no solo desde el punto de vista de la empresa, si no además del usuario.

Es decir, estos factores en los que se basan los planteamientos estratégicos de creatividad, conocimientos, eficiencia y valores humanos no son solo valorados desde el punto de vista de la empresa. Ahora se da la vuelta a la relación empresa-cliente, y se observan desde el punto de vista del usuario de software. Esto es lo que Richard Stallman planteaba: la libertad de los usuarios que además debe acarrear incluso mejoras sociales.

♦ Bugzilla: Sistema para administrar, visualizar e introducir los fallos o mejoras pendientes en un programa. Es un sistema de software libre.

¿Significa que las empresas comerciales intentan omitir factores elementales de la cultura en su relación con el cliente? Sinceramente no lo creo. Simplemente el punto de vista es distinto. No creo que una relación comercial entre un productor de software y un usuario, la venta de un código binario mediante unas licencias, suponga ninguna trasgresión de los derechos de ninguna de las partes. Pero el nuevo modelo de negocio implica simplemente la visión desde el otro lado, desde el lado del usuario. Y ello ha llevado a cambios muy importantes en el sector industrial.

El modelo de negocio basado en software libre facilita sobremanera la capacidad de innovación mediante la copia. De hecho, la empresa que adquiere la innovación mediante la utilización de software libre, ni siquiera debe hacer esfuerzo en realizar la copia, no supone un trabajo de análisis de producción para alcanzar la producción igual que la original. Simplemente es software el libre, y por lo tanto también lo es la innovación que represente.

La innovación de ruptura puede venir de distintos departamentos que los encargados del desarrollo. En este punto, la empresa comercial puede tener ventajas sobre una comunidad de usuarios no organizada. Los distintos departamentos de la empresa colaborarán en buscar mejoras de ruptura, que no tienen por qué venir únicamente desde el perfil técnico. Marketing o el departamento financiero, por ejemplo, pueden (y deberían) colaborar en distintas ideas que puedan significar desde su punto de vista una innovación.

En las comunidades de usuarios muy distribuidas, generalmente las innovaciones provienen del esfuerzo de una sola persona. Estas innovaciones pueden ser brillantes, pero por lo general tendrán ventaja los equipos bien organizados de las empresas. Por esto, las empresas basadas en software libre deberían realizar un esfuerzo para encaminar a la comunidad a una organización dónde se fomente la innovación, o realizarla internamente con conexiones al exterior.

En el software de código abierto la innovación incremental es la más aplicada. Los desarrolladores van mejorando un producto, añadiendo cada uno su porción de innovación, pero es más difícil que se planteen un cambio radical al producto.

En las comunidades de software libre la innovación descansa únicamente en los desarrolladores de forma individual, pero las empresas pueden innovar compartiendo visiones distintas: desde el área comercial, desde la alta dirección o desde el área financiera. Estas areas pueden aportar, y de hecho es necesario que lo hagan, un fuerte empuje para facilitar la innovación empresarial.

Se suceden mayor número de iteraciones para obtener un producto terminado, debido a la externalización de los recursos. Puede tener un coste que hay que gestionar (repositorios, listas de correo,... en definitiva, es muy importante el sistema de información empresarial)

Las redes son la mejor explicación del funcionamiento de las comunidades de software libre. Los intercambios entre las personas y organizaciones involucradas en un desarrollo de software código abierto no sigue ninguna jerarquía, puesto que no existe una súper-organización, al menos explícitamente, que englobe a todas las partes.

La importancia de las 3 Cs: Crear, Colaborar y Compartir, en la innovación es importante. Esto se realiza ya incluso con proveedores, clientes e incluso competencia. Se puede ver a distintas empresas sin acuerdos escritos entre ellos colaborando entre sí. La licencias GPL en alguna manera les obligaría a ello, puesto que deben publicar sus cambios y

mejoras. De esta manera, la misma naturaleza de la licencia en cierta manera fomenta la innovación basada en el trabajo de un grupo. Por lo menos de manera más clara se fomenta la creación y el compartir, por que quizá sea en la colaboración como manera de innovar donde se encuentren mayores problemas. Todos colaboran creando el mismo producto, pero es más que posible que cada trabajo vaya en su dirección, la que le interese al programador voluntario, o a la empresa que lo desarrolla.

Actualmente la gestión del SCA es muy descentralizada, o al menos en teoría, pero se debería poner más atención a los requisitos de los clientes.

5.5. Organización empresarial

Las comunidades de software libre pueden ser muy complejas y vastas. La empresa que tome el SCA como parte fundamental de su inversión, debe desarrollar mecanismos para adaptarse a la realidad externa a ella. Como hemos hablado, la comunidad forma una red, donde cualquier persona puede aportar algo al código fuente.

Esto va a obligar a las empresas del entorno del software libre a ser muy dinámicas, con mucha comunicación entre los propios desarrolladores que conocen a fondo el estado del producto y el equipo de marketing, que necesitará estar informado de las últimas novedades. Aquí aparece la adaptación mutua como forma favorita de organización, puesto que en el modelo caótico que se nos presenta puede será la mejor y más flexible alternativa. La empresa debe estar poco jerarquizada y confiar mucho en sus trabajadores.

Las circunstancias complejas y cambiantes del modelo de negocio basado en software libre, donde no se controlan todas las variables que afectan a un producto, hace que la adaptación mutua sea posiblemente la que mejor responda como mejor forma organizacional. Esto además será especialmente crítico cuando el producto esté en su fase de nacimiento y más tensiones existan para llevarlo hacia una línea de desarrollo u otra. En esta fase la organización debe responder estableciendo sus líneas maestras para el seguimiento de la estrategia de la empresa. Por tanto la organización también debe asegurar que los mandos intermedios que existan hacen conocer a todos los trabajadores sus metas empresariales.

No olvidemos de todas maneras que una estrategia alrededor del software libre siempre debe estar muy centrada en lograr la máxima calidad del producto, o de lo contrario la comunidad dejaría de confiar en una empresa que puede anteponer sus intereses a los del producto de software libre. Por tanto, asegurando la calidad del producto y la transparencia de la empresa en conseguirla, se alcanzará una de las estrategias claves: conseguir la confianza de la comunidad de software libre. Todos los trabajadores deben centrarse en ello, y debe garantizarse mediante la organización empresarial adecuada.

Tradicionalmente el software se ha desarrollado bajo una jerarquía bastante estricta, dónde las personas implicadas podían jugar estos roles:

- Jefe de proyecto: Quien gestiona el equipo y se asegura de su buena marcha.

- Analistas: Que obtienen los requerimientos del cliente y realizan el diseño del software a implementar. A veces se dividen en analistas *senior* y *junior*, y se dividen la responsabilidad de tratar con el cliente o la iniciativa de tomar decisiones estructurales del software.

- Programadores: Como último escalafón seguirían los diseños de los analistas creando el código fuente. También se suelen distinguir entre *senior* y *junior*, principalmente por su experiencia.

Ahora estamos hablando de que hay que proporcionar mayor confianza a los trabajadores, de que son estos quienes deben relacionarse con el mundo exterior. En este caso, los verdaderos conocedores del código fuente son los programadores. Cada vez más se escucha que se debe dar mayor importancia a los programadores, valorarlos más.

Podemos asegurar que la clave de una empresa son sus personas. Las técnicas de organización, de calidad, de producción,... cada vez se orientan más hacia el valor de la persona. Por supuesto, los procesos son también importantes, pero "la clave del éxito son las personas". La organización empresarial de una empresa de software, ya sea comercial o basada en software libre, no debe olvidarlas.

Los roles tradicionales limitan la capacidad de las personas. Nadie quiere ser programador como rol tradicional, porque se limita a realizar un trabajo a veces carente de iniciativa en los proyectos más burocratizados. Además, es un trabajo menos valorado. Una organización que motive a las personas intentará que cada uno rinda al máximo de sus posibilidades, centrándose en los equipos como medio fundamental para la creación de software. En la creación de software esto es todavía más importante, puesto que ya hemos hablado que no se pueden establecer procesos idénticos, porque no existen los mismos problemas. Cada producto de software es único, aunque tengan las mismas funcionalidades.

Ya existen formas organizacionales de los equipos de desarrollo de software diseñadas para eliminar las barreras jerárquicas, para mejorar la comunicación y disminuir la burocracia. Un ejemplo de esto es la metodología de gestión de proyectos de software conocida como "*eXtreme Programming*[♦]" o XP. Por supuesto, esto puede ser aplicable a empresas productoras de código abierto y las que no lo son.

La metodología XP devuelve la importancia al programador de aplicaciones, y le da la iniciativa y el valor que pierde en las metodologías tradicionales. Supone que a los programadores les gusta el trabajo que hacen, y lo intentan hacer de la mejor manera posible.

Las comunidades de código abierto eliminan también en gran manera los roles tradicionales. Cualquiera puede encontrar un problema o ver que mejora quiere realizar, analizarla, implementarla y compartirla. De una manera muy parecida que el "*eXtreme Programming*" propone, las comunidades realizan su código.

¿Cómo deben adaptarse las organizaciones a esta manera de trabajar? Deben hacer uso de una organización muy adaptable, flexible y que confíe totalmente en sus trabajadores.

Es importante destacar de nuevo el impacto en la organización de la empresa del hecho que no se controle el producto completamente si este se distribuye. La organización debe ser muy ágil para absorber cualquier cambio importante en un producto. Puede ser de repente que aparezca otro proyecto en la comunidad que aglutine a los desarrolladores externos del nuestro, e incluso puede que este nuevo proyecto sea un software competencia o sustitutivo del mismo.

♦ Programación eXtrema.

¿Qué pasa si de repente los desarrolladores externos con los que hemos trabajado deciden empezar otro proyecto de similares características pero partiendo de cero? La repercusión en la organización puede ser muy fuerte. Por tanto esta debe ser extremadamente en reaccionar, y estar lista para hacerlo de manera adecuada. Las decisiones no se pueden alargar o perder entre las jerarquías de mandos intermedios. Debe llegar pronto a quien diseña el fundamento de la estrategia empresarial.

La forma de comunicación con el exterior también estará determinada por la estructura de la organización. Posiblemente esta comunicación se realice a dos niveles importantes: los trabajadores que colaboran con los desarrolladores externos, y el departamento de marketing. Por tanto, es muy importante que fluya la comunicación entre estas dos partes. El departamento de marketing debe conocer en todo momento el "estado del arte" del producto desarrollado y las tendencias del mercado externo sobre el mismo.

Por tanto, la principal diferencia a la que debe responder la estructura organizacional de una empresa basada en software libre de una que no, es la falta de control del producto, el hecho de que los agentes externos sean tan importantes en el producto o servicio ofrecido.

El modelo muy descentralizado de desarrollo del producto, sus características, obliga a adaptar la organización a este hecho. El resto de la empresa debe responder a las necesidades que esto pueda implicar, porque los cambios pueden suceder de manera interna o externa, y la organización debe estar preparada para adaptarse a los cambios.

5.6. Estrategias de desarrollo.

El desarrollo de los nuevos productos de SCA implica nuevas estrategias en el desarrollo o **estrategias de producción** de software debido a la posibilidad de que personas o empresas ajenas participen en el desarrollo del producto. Aquí se repite la idea de que "el competidor también es colaborador".

La producción del software no puede funcionar de la misma manera que en un entorno empresarial cerrado, puesto que habrá aportaciones al producto que resulta imposible controlar, ni se tendrá ningún poder sobre ellas. Hemos visto cómo el valor añadido de los productos con los que trabaja la empresa puede ser creado por terceras partes. Esto puede tener varias consecuencias importantes:

- Velocidad de desarrollo: La velocidad de desarrollo de un producto puede aumentar al contar con recursos externos.

- Menor carga de trabajo: La carga de trabajo a repartir puede ser menor si se cuenta con recursos externos.

- Desinformación parcial: El hecho de que partes del producto sean desarrolladas por personas externas, pueden llevar al desconocimiento del producto parcialmente.

Estas características se ven limitadas por la proporción del peso del desarrollo de productos contra el peso de los servicios. La empresa que desarrolle productos, para luego distribuirlos, como software libre, contará con las ventajas de que conoce mejor el producto para dar servicios sobre él.

El software libre va a permitir a las empresas que den servicios basándose en él, un conocimiento mucho mayor del producto para responder mejor a los requerimientos de los clientes. Quizás la empresa de servicios no participe en el desarrollo de manera general, pero el hecho de poseer el código fuente crea la posibilidad de realizar las modificaciones necesarias para cualquier cliente que lo solicite, y la investigación en profundidad del comportamiento del software. La empresa que da servicios sobre programas comerciales, sólo puede reclamar al fabricante cuando tiene problemas con el producto. Por tanto, se debe contrastar independencia con responsabilidad (del fabricante).

Genéricamente se han comparado los dos modelos de desarrollo distintos como el modelo bazar **[Raymond 1998]** y el modelo catedral. Su propio nombre indica bastante bien en que se basa la distinción. El modelo bazar es aquel donde el software crece de una manera caótica. Cada uno realiza sus aportaciones, modificaciones, y cualquier cuestión que le plazca. Este desarrollo vendría a significar un caos más o menos controlado, donde la comunicación entre las partes es la pieza fundamental donde se ciñe todo.

El modelo catedral es aquel dónde existe una línea clara en el desarrollo, dónde un arquitecto decide donde deben poner las piezas los trabajadores. La idea es que en las empresas existe una jerarquía más o menos fuerte, que indica las labores a realizar, de manera ordenada.

Estos dos modelos obviamente no son rígidos. De hecho, las empresas más complejas y grandes irán hacia un modelo cada vez más basado en el modelo catedral. Las técnicas de calidad cada vez dan más importancia al trabajador, y por tanto más importancia a sus decisiones.

Pero la empresa debe adaptar el modo de trabajo de sus desarrolladores al modelo de software libre. El modelo bazar con el que crean las comunidades el software libre, no es apropiado en un ambiente empresarial **[Martínez 1999]**. La estrategia de desarrollo de la empresa debe ser aplicada de la manera apropiada.

El producto sobre el que se da soporte puede estar soportado por la misma empresa o por otra. De todas maneras, debemos estar al tanto de las listas de distribución, de los cambios en el producto (servidores de código compartido) y de las listas de fallos o “bugs” que vayan surgiendo. La descentralización del desarrollo puede ser una ventaja, pero también una incomodidad para los desarrolladores.

El desarrollo del software libre, basado en comunidades, tampoco es tan anárquico como ha podido aparecer hasta ahora. La realidad es que en todo proyecto existe un líder que tiene la capacidad de tomar decisiones sobre el desarrollo, y las líneas de trabajo. Por ejemplo, en Linux, Linus Torvalds es el líder, quien decide cuando se lanzan nuevas versiones del kernel o que estrategias generales seguir en su implementación.

Además, en principio cualquiera no puede generalmente añadir modificaciones al repositorio de código compartido. Normalmente los desarrolladores se “ganan” ese derecho cuando han hecho varias sugerencias de modificación del código en las listas de correo de los desarrolladores, corrigiendo fallos o añadiendo alguna funcionalidad.

Las empresas que proporcionen servicios sobre código abierto es muy importante que tengan sus desarrolladores buena imagen en la comunidad de software libre, y derechos para poder hacer sus modificaciones. Aunque también podrían plantearse su propia línea de

desarrollo, esto puede llevar a quedarse separado del mercado, si no se es lo suficientemente grande en el mercado para atraer a más clientes.

Más específicamente sobre el modelo de desarrollo del código abierto, la consultora Forrester califica el modelo bazar como el modelo científico de desarrollo: **[Schadler 2004]**

Características	Beneficio
Revisión por pares	• La meritocracia de desarrolladores da a los mejores programadores las posiciones líderes. • Solo las mejores ideas y código se lanzan en las versiones finales. • Un "líder"(generalmente el fundador) es el "dictador benevolente"
Resultados publicados	• Como el código es público, los fallos son corregidos rápidamente • Muchos equipos pueden trabajar en módulos diferentes simultáneamente. • Las mejoras se soportan totalmente por el código existente.
Programadores voluntarios	• Internet mantiene la comunidad unida-virtualmente ilimitada. • Los programadores están altamente motivados por que escogen su propio trabajo.
Versiones lanzadas por ingenieros	• Se lanzan las versiones cuando los ingenieros piensan que el sistema está listo para producción, no cuando conviene al depto. de ventas.

Tabla 3: El desarrollo de SCA utiliza el método científico

Esta consultora destaca los beneficios en el modelo de desarrollo del código abierto, que se han probado con éxito en proyectos como Linux, Apache o PHP. Las empresas comerciales deben estudiar sus propias estrategias de desarrollo para verificarlas contra esta tabla, pero sin perder de vista su gran ventaja: el contacto con el cliente, su conocimiento y su habilidad para satisfacer sus necesidades.

Si el modelo empresarial de desarrollo, más jerárquico y definido, se observa en una empresa que no aporta las ventajas que puede aportar este modelo científico, la organización deberá replantearse una adaptación de este modelo al suyo, sin perder las ventajas que le proporciona un desarrollo guiado por y para el cliente, seguramente a través del departamento de marketing.

5.7. Distribución

La distribución de los productos software puede resultar muy sencilla. Internet es la clave. La distribución ya no presenta problemas de logística, transporte,... gracias a las redes de telecomunicaciones. Pero también puede ser un problema para algunos clientes, el hecho de no "ver" o conocer quién vende. Se necesita una buena imagen de empresa, bien posicionada, para que la gente se atreva a probar tu producto.

Para ello se debe "conseguir que nuestro web se convierta en un portal" **[Alonso 2002]**. De esta manera obtendremos una imagen de marca y un lugar donde atraer los clientes, informarles y obtener datos de ellos. El portal se convierte en Internet en el punto de contacto con los clientes, por lo que debe ser cuidado hasta el mínimo detalle, convirtiéndolo en lugar de visita frecuente por lo clientes, por su sencillez, información e interés.

La distribución de los productos software se puede realizar por medio de la venta de CDs, DVDs,... que implican una cierta logística, pero la orientación hacia los servicios de las empresas, debe minimizar esta cuestión, posiblemente externalizándola.

Los servicios ofrecidos dependerán en gran medida de su tipo. Pueden ser servicios de consultoría presencial, atención telefónica o formación. Cada tipo de servicio implica una logística diferente, aunque por supuesto en cualquiera el principal cometido es la satisfacción y fidelización de los clientes (y los empleados).

Actualmente, las empresas que dan soporte a Linux, por ejemplo RedHat, proporcionan una distribución con una versión concreta del sistema operativo libre, que es sobre la que dan soporte. Pero en realidad, restringen algunas de las ventajas del código abierto, puesto que si se modifica alguna parte del código, se pierde el soporte contratado. Por una parte tiene la lógica de que una empresa no puede hacerse responsable de los cambios que realicen los clientes, pero por otro lado, se restringen los beneficios de la modificabilidad por el cliente y la corrección rápida de bugs, así como se minimizan los beneficios del intercambio del código.

5.8. Calidad

En este punto es crítico determinar si el hecho de que el software sea SCA significa que será de mayor calidad o no. Las empresas que se planteen la adquisición un software deben evaluar la calidad del producto a adquirir para que les reporte un mayor valor añadido a sus procesos productivos. La calidad de ese software puede determinar algunas cuestiones críticas en su uso, como tiempos perdidos por fallos o pérdida de valor por características que no implemente y fuesen necesarias para la empresa.

Ante todo, ¿qué es un software de mayor calidad? Lo definiré como un software que proporciona mayor valor añadido para quién lo utiliza, teniendo en cuenta que sus posibles fallos[♦] van a mermar ese valor proporcionado, así como la dificultad en su uso.

Se pueden definir una serie de atributos que debe tener el producto de software, que son comúnmente aceptados para que tengamos un software de calidad **[Glass 2003]**:

- Fiabilidad: El software hace lo que se supone que debe hacer.
- Portabilidad: El software puede ser fácilmente transportado a otra plataforma.
- Eficiencia: El software hace lo que hace de la mejor manera posible, economizando tiempo y espacio[♣].
- Usabilidad: El software es fácil y cómodo de usar.
- Testable: El software es fácil de probar su funcionamiento.

♦ Un fallo en un programa software se conoce popularmente como “bug”

♣ En el software generalmente existe una dicotomía entre estas dos variables: un software que sea muy rápido ocupará mas espacio(memoria), y uno que ocupe muy poco espacio será más lento.

- Entendible: El software es fácil de comprender por las personas que lo van a mantener.

- Modificable: el software es fácil de modificar por las personas que lo van a cambiar.

Si entendemos la calidad como una lista de atributos, los anteriores serían los que debería cumplir. El grado en que un software cumpla esos atributos determinarían su nivel de calidad. ¿pero qué modelo de negocio va a proporcionar mayores valores de estos atributos por sí mismo?

En teoría, ninguno tendría que ser mejor que otro para aumentar el valor de estos atributos solo por el hecho de que el software sea desarrollado bajo un modelo u otro. Tanto los programadores y desarrolladores de ambos modelos pueden crear el mismo producto con la misma calidad. Hemos visto que la clave está en las personas. La Gestión de la Calidad Total insiste en esa idea. Cualquiera de los dos modelos, funcionando correctamente, pueden generar software de calidad.

Pero, y me atrevo a poner un pero, la realidad puede hacer que algunos de esos atributos sean más "queridos" por un modelo que el otro. Mi experiencia me inclina a pensar que el modelo de negocio comercial tiene más ventajas en la usabilidad y la fiabilidad; y que el modelo de código abierto se centra en la modificabilidad, entendibilidad y portabilidad. En estos atributos veo posibles diferencias a la hora de desarrollar los productos. Por supuesto, no quiero decir que se ignoren los otros atributos, solo destaco las visiones que he tenido de los diferentes productos.

Las empresas del modelo de negocio comercial focalizan la atención en el departamento de marketing. Esto les acerca más a los clientes, cuyas necesidades son estudiadas de manera sistemática y organizada. Se conoce mejor lo que buscan en un producto software y se esfuerzan en desarrollar aquellas características que más valoran. La usabilidad es muy controlada por que supone el contacto de las personas-cliente con el producto, cuidando y diseñando la interfaz del producto para que tengan una experiencia de uso satisfactoria.

La usabilidad suele ser un punto débil en los programas de software libre, puesto que los desarrolladores que la crean son generalmente expertos informáticos. De esta manera, no cuidan con exageración la usabilidad, puesto que ellos no tienen problemas en entender el sistema.

Las comunidades de software libre se esfuerzan en su comunicación, el código desarrollado por una persona, que sabe que debe ser leído y entendido por otras, es más modificable y entendible. Las comunidades suelen tener reglas de estilo de código fuente muy estrictas. Además la heterogeneidad de las personas pertenecientes a las comunidades, nos lleva al hecho de que se esfuercen en la portabilidad, puesto que cada persona puede tener una plataforma de ejecución y desarrollo diferente.

En las empresas se suele insistir mucho en los plazos de entrega, y ello suele llevar a una programación del software más caótica, dónde la comunicación se olvida, y se trata de terminar cuanto antes el trabajo. La modificabilidad y entendibilidad se resienten.

Insisto en que las tendencias anotadas anteriormente son solo opiniones obtenidas de la observación del mercado. Además todo puede cambiar: las empresas comerciales son tan

grandes que empiezan a funcionar como una comunidad en sí mismas, o las comunidades se organizan de tal manera que se plantean nuevos retos o surgen empresas que lideran los cambios de manera más organizada.

Y en realidad esta definición de atributos se queda corta para poder conseguir la satisfacción del cliente. Para ello se necesitan algunas cosas más: que el producto satisfaga los requerimientos, que sea entregado en los plazos previstos y a un coste apropiado, especialmente si nos referimos a los servicios.

Por tanto, estamos basando las empresas de software libre en los servicios. Servicios que deben estar basados en productos de código abierto de calidad, pero también estos tienen su propia noción de calidad. Podríamos verlo como un todo, el producto implica unos servicios, y los servicios se harán mediante unos productos utilizados en la implementación.

Anteriormente hemos comentado las diferencias en los modelos de desarrollo. Los procesos productivos de software de una empresa o de una comunidad de código abierto son diferentes, sobre todo si nos ceñimos a la forma más tradicional de roles en los proyectos empresariales.

Pero, ¿qué implicaciones tiene todo esto en la calidad? El desarrollo de un producto basado en SCA tiene un enfoque claramente basado en el producto. Pero en este enfoque desaparece un elemento fundamental que es la estrategia de marketing basándose en el usuario, para crear un diseño que satisfaga sus requerimientos. Por otro lado, las empresas comerciales puede definir el enfoque que quieren proporcionar a su ingeniería de la calidad para obtener los mejores resultados. Generalmente, las empresas comerciales se orientan de manera más clara hacia el cliente, al que intentan conocer y satisfacer.

Este punto se ha comentado varias veces anteriormente. Los desarrolladores de software libre, en general desarrollan lo que necesitan, teniéndose en cuenta ellos mismos como clientes. Con la aparición de cada vez más empresas de servicios sobre software libre, sin duda esta cuestión irá cambiando, pues podrán aportar al desarrollo su visión y conocimiento de los clientes que utilicen dicho software, aportando otro enfoque de calidad, basado en el valor o incluso un enfoque trascendente.

Por tanto, parece crítico para un proyecto de software libre, y para su orientación hacia la calidad de manera más global, que existan empresas de servicios que aporten una visión más extendida que la que tienen únicamente los desarrolladores. Por ejemplo, y tomando el caso del ya comentado sistema operativo Linux: nadie duda de que el producto funcionaba muy bien hace ya unos pocos años, se puede decir que tiene un enfoque de calidad basado en el producto, pero que entonces le faltaban tres factores importantes para su lanzamiento al público general, según los factores enunciados por Garvin:

- **Conformidad**: No se comprueba que el producto satisfaga los requisitos mínimos de seguridad, fiabilidad,... Los clientes necesitan que alguien les asegure que se cumplen.

- **Estética**: Generalmente los programas de software libre son realizados a la medida de los desarrolladores que los han creado, no se preocupan del aspecto gráfico ni su usabilidad.

- **Calidad percibida**: Los clientes no tienen referencias fiables sobre las características totales del producto. No saben realmente como funciona o desarrolla sus servicios.

Cuando varias empresas comerciales, en las que el mercado tiene depositada suficiente confianza, empiezan a apoyar un software libre pueden proporcionar la conformidad al producto, al dar su visto bueno, y mejorar la calidad percibida de los clientes al tener referencias válidas del producto.

Por ejemplo, en mi opinión es fundamental para el despegue actual de Linux el apoyo de empresas como Oracle o IBM. Estas empresas certifican sus productos sobre esta plataforma de código abierto, y los clientes, que ya tienen confianza en estas grandes empresas, la trasladan también al software libre.

A modo de resumen, y basándome en el artículo de Ted Schadler **[Schadler 2002]**, presento una tabla con las principales diferencias entre el software de código abierto y el comercial. Este consultor de la prestigiosa consultora *Forrester*, presenta una tabla que he ampliado en algunos puntos:

	SOFT. LIBRE	SOFT. COMERCIAL	IMPACTO
Calidad	Calidad por motivación, controles, eficiencia técnica.	Calidad vigilada, usabilidad, mayor atención al cliente.	Calidades diferentes
Proceso desarrollo	Método científico	Top-Down*	Los proyectos de SCA pueden beneficiarse de cientos de desarrolladores.
Estrategia de futuro	La comunidad, dirigida por un líder.	Decisiones estratégicas con el cliente como centro: Marketing y niveles directivos.	La portabilidad aumenta en SCA, pero a expensas de funcionalidades importantes.
Innovación	Incremental	Incremental, pero a veces aparecen innovaciones de ruptura.	El SCA es bueno para versiones estables, pero poca innovación de ruptura.
Licencias	Generalmente gratis	Pago por licencia.	El costo sobre SCA es considerablemente menor.
Soporte	Numerosas empresas libres de darlo.	Empresa productora o "partners" certificados.	La competencia beneficia al SCA.
Estándares	Estándares por definición	Estándares seguidos como último recurso, o con trampas.	El SCA acelera la adopción de estándares.

Tabla 4: Principales diferencias entre software libre y propietario.

A raíz de todo esto surge la idea más clara de que el SCA se convierte en software como servicio **[Baranda 2002]**. Habrá empresas que creen software libre que tendrán servicios de consultoría, formación, y soporte. Los ingresos por el software dejarán de obtenerse del software como producto para pasar a software como servicio.

El hecho de que las empresas se deban orientar cada vez más a los servicios debe cambiar la manera de pensar en las variables de medición de la producción como las únicas variable válidas de medición de la rentabilidad de la organización. Ahora se debe tener más en cuenta la fidelización y rentabilización de los clientes.

Las empresas de servicios deben conseguir alargar la vida media de los clientes. Con el software libre se debe ser consciente de que existirán muchas empresas capaces de dar servicio sobre el mismo producto, y que incluso los clientes podrían decidir prescindir de los servicios externos para realizar internamente las tareas necesarias. La no restricción del código fuente permite esa libertad

* Método Top-Down: (Arriba-Abajo): Hace referencia al modo de creación de software basado en las especificaciones del cliente, de las que luego se obtiene un diseño y se codifica desde los puntos más genéricos hacia los particulares.

que aumenta la competencia. Por tanto, el logro de fidelización de clientes debe ser la máxima prioridad para las empresas de servicios basados en código abierto.

También la fidelización de los empleados, y en este caso incluso de la comunidad de desarrolladores y usuarios debe ser una actividad constante. La comunidad, como ya se ha destacado varias veces, va a jugar un rol muy importante en el éxito o fracaso del producto, y por tanto de los servicios creados alrededor suya.

Luego existirán dos tipos de utilización del software libre, dependiendo de su utilidad. Unas empresas crearán sus producto y/o servicios sobre la infraestructura que les proporcione el software libre, y el resto de empresas serán usuarias directamente del software de código abierto, que directamente les proporcionará le valor añadido que necesitan. Estos clientes pueden no necesitar mas que algo de formación de los productos, y no van a contratar ni consultorías ni servicios, puesto que la herramienta ya se ajusta a sus necesidades.

Por tanto, existirán dos tipos de productos:

- de infraestructura: los sistemas operativos, los servidores de aplicaciones,... son productos que no son un fin en sí mismos, si no que son usados para crear aplicaciones de valor añadido sobre ellos. En estos casos los servicios serán importantes, y las empresas que los den adecuadamente pueden conseguir un sector del mercado importante.

¡

- para el usuario final: las aplicaciones ofimáticas, una aplicación de diseño industrial, un navegador web,... son aplicaciones que no necesitan una posterior personalización (generalmente) ni requieren apenas servicios. En este caso la empresa que cree un producto que necesite una importante inversión en investigación y desarrollo, debe plantearse hasta que punto podría prescindir de los ingresos por licencias.

6. Los clientes de código abierto

6.1. Evaluando los costes de la tecnología

Para determinar cuál va a ser el coste de la elección de una determinada tecnología, la consultora Gartner acuñó el término TCO (*Total Cost of Ownership*: Costo total de Propiedad) donde se define el método para evaluar los costes reales para la implantación de una tecnología en la empresa **[García, Romeo 2004]**.

Otro método que puede ser usado es el cálculo del ROI (*Return of Investment*) centrándose más en los beneficios que vaya a aportar posteriormente el empleo de la tecnología. Pero numerosas veces en las instalaciones o cambios tecnológicos importantes, no llega a quedar claro cuál es el beneficio monetario que producen realmente, al repercutir en cuestiones abstractas como el método de trabajo de las personas. Sin embargo, el ROI puede dar realmente la diferencia importante entre dos tecnologías distintas. Si un producto más caro aumenta la productividad de los trabajadores, tendrá un mayor ROI que otra tecnología que no lo haga, aunque esta sea más barata. El ROI ya depende de cada caso particular, puesto que el retorno de la inversión dependerá de los efectos que produzca, y por ello explicaremos aquí los costes derivados de la tecnología.

El TCO no determina los costes de manera que se pueda decir que una tecnología es más barata siempre que otra, si no que también tiene en cuenta los factores propios de la situación que se analiza. Por tanto, estos costes deben ser calculados para cada situación dónde debamos tomar la decisión de elegir una tecnología u otra.

Costes directos	
Software	Coste de adquisición / Licencias
	Actualizaciones y añadidos
	Costes de propiedad intelectual
Hardware	Coste de adquisición
	Actualizaciones
Costes de soporte (Interno)	Instalación y configuración
	Mantenimiento
	Resolución de problemas
	Otros (libros, revistas,...)
Costes de soporte (externo)	Instalación y configuración
	Mantenimiento
	Resolución de problemas
Costes de personal	Gestión del proyecto
	Ingeniería y desarrollo de sistemas
	Administración de sistemas
	Formación

Costes indirectos	
Costes de soporte	Soporte entre trabajadores
	Aprendizaje casual
	Tiempo gastado en formación
	Factor futz*
Tiempo perdido	"Downtime" del sistema

Tabla 5: Elementos del TCO para la evaluación de la tecnología. [Mitre 2001]

Los **costes directos** son los que se derivan del hecho de la compra de la tecnología y su proceso de implantación. Los costes del software son claramente favorables al software libre, donde no existen gastos en licencias, o estos son mínimos comparados con el software propietario.

El coste de hardware en principio no debería ser menor en un caso o en otro de por sí. Sin embargo, el ejemplo de Linux muestra que el hardware necesario comparado con las soluciones propietarias de Microsoft es menor. Se necesita menor gasto en hardware si vas a instalar un sistema Linux para alcanzar la misma eficiencia que la solución basada en Windows.

Los costes de soporte, tanto internos como externos, serán muy parecidos en ambos casos. El trabajo a realizar internamente o que será externalizado a compañías de consultoría será el mismo. Suponiendo que los trabajadores desconozcan por completo las soluciones que se estén estudiando para su implantación, la formación también sería equivalente. Pero obviamente para estudiar el coste de soporte interno es muy importante estudiar a la perfección los conocimientos de los empleados, puesto que pueden tener una experiencia anterior en un sistema u otro. Los costes externos de soporte disminuirán en función del grado de utilización por la sociedad de ese software, puesto que al ser libre, numerosas empresas se crearán dando soporte y la mayor competencia disminuirá los precios de los servicios. Se debe evaluar además en el software libre como son sus comunidades de usuarios: una amplia comunidad puede aportar un importante soporte de manera gratuita.

Los costes de personal tampoco representan una diferencia importante entre los dos modelos de software. La diferencia puede venir en la dificultad de conseguir trabajadores especializados en una u otra plataforma, dependiendo de la experiencia requerida y el mercado que tenga n producto u otro.

Los **costes indirectos** son los generados por la pérdida de trabajo real obtenido por la compañía por la influencia de la nueva tecnología. Cuando los trabajadores necesitan ayuda para conocer cómo realizar algo con la nueva tecnología, y se lo preguntan a un compañero o lo investigan ellos mismos; cuando están en un tiempo de formación; o cuando se encuentran realizando tareas personales no relacionadas con el trabajo se está perdiendo un tiempo que va a generar un coste indirecto. Este coste es muy difícil de calcular debido a su ambigüedad y difícil control, y es el mismo para cualquier tecnología independientemente del modelo de negocio en que se base.

El ejemplo más claro del factor "*futz*" es el uso del correo electrónico por parte de los trabajadores para comunicarse temas de índole personal en el trabajo, o el problema del "*spam*", correo no deseado que inunda actualmente los correos electrónicos.

* El factor futz es considerado por la consultora Gartner como un costo indirecto. Se produce cuando los trabajadores utilizan la tecnología para su provecho personal en horario de trabajo.

Otro coste indirecto es el que genera el tiempo perdido por los fallos del sistema tecnológico. Este se debe evaluar en función de la calidad del software a elegir, no depende directamente de que sea software libre o no. Pero, por ejemplo, en el caso de los sistemas operativos, según la mayoría de estudios, Linux es más estable que Windows, y los usuarios de este último experimentan mayores problemas, con mayores pérdidas de tiempo implicadas.

6.2. Los usuarios domésticos

Para los usuarios domésticos, entendidos como personas que utilizan el software para su propio beneficio, el software libre puede tener una serie de ventajas. Todas las ventajas que proclama el software libre pueden ser aplicadas al usuario particular, pero se debe contrastar con las ventajas del software comercial.

- Ahorro: Para un particular puede ser muy importante este factor. El hecho de ahorrarse las licencias del sistema operativo, del software de ofimática y algún que otro programa de dibujo o de gestión puede inclinar la balanza favorablemente hacia el software de código abierto.

- Legalidad: Los usuarios que utilizan programas comerciales "pirateados" dejarían de cometer un acto ilegal al pasarse al software libre.

- Disponibilidad: Existen multitud de programas desarrollados bajo la licencia GPL, que pueden cumplir las expectativas del usuario normal. Existen miles de proyectos de software libre, aunque la mayoría de ellos sean abandonados pronto o sea prematuro el empezar su utilización.

- Participación: El usuario puede darse la satisfacción de colaborar en un proyecto en el que puede formar parte de forma activa y ser valorado dentro de una comunidad.

Otros temas como compatibilidad, seguridad o estabilidad pueden ser más discutibles enfrentando los dos tipos de software. **[Ordovas 2004]**

6.3. Las empresas cliente

Las empresas que utilicen código abierto como infraestructura para sus sistemas de información o tecnológicos, buscarán los posibles beneficios o ventajas que les reporten estos.

Las empresas ya no buscan un producto, si no una solución a su problema. Generalmente se busca una solución completa al adquirir un producto, atendiendo a los servicios posteriores a la compra como pueden ser formación, consultoría o atención telefónica. Las empresas buscan un respaldo general en sus suministradores, puesto que de ellos puede depender en gran medida su situación futura.

Muchas empresas quieren tener como socios a sus proveedores estratégicos, y el proveedor de tecnología sin ninguna duda lo es, de esta manera se aseguran un mayor control sobre sus acciones y una mayor garantía de futuro. Los acuerdos estratégicos de alto nivel con las empresas suministradoras aportan confianza en la continuidad del modelo de negocio.

Ante esto se debe evaluar que ventajas o desventajas proporciona el SCA. Y aquí hay implicaciones a corto plazo, como pueden ser la calidad del producto; y a largo plazo, como puede ser el respaldo estratégico que proporcione el suministrador.

Las ventajas del software libre pueden ser proporcionar independencia de las empresas suministradoras, puesto que la libertad del código implica incluso que se puede realizar un auto mantenimiento, y que cualquier empresa del mercado puede hacerse cargo de suministrar las futuras versiones. La afinidad del software libre con los estándares que deben seguir los productos también facilita esta independencia para posibles cambios de plataforma. **[Senf 2003]**

Esto también tiene la desventaja de que se puede no ver un responsable claro del producto. La empresa debe valorar los acuerdos estratégicos contra la independencia del suministrador. Generalmente la mayoría de empresas no están a la altura de firmar acuerdos estratégicos con los grandes proveedores de software, y preferirán la ventaja de tener una relativa independencia de los mismos.

Otra cuestión a tener en cuenta es que el mantenimiento de las aplicaciones puede ser muy rápido, al ser accesible tan pronto como un desarrollador corrige un fallo. Incluso, puede ser obtenida la corrección dentro de la propia empresa, al poseer el código fuente.

En mi opinión el SCA existente hasta ahora tiene una diferencia cualitativa en cuanto a los usuarios finales: el diseño de la interacción persona-computador. En la mayoría de los casos, el SCA más utilizado no tiene una interfaz demasiado amigable, que pueda llamar la atención de personas sin habilidades especiales en el campo de la informática. Este aspecto está muy cuidado en las empresas, que priman muchas veces el aspecto gráfico y la facilidad de manejo a la funcionalidad.

Esta cuestión impide un despegue mayor de los programas de SCA. El hecho de no seguir claramente una estrategia orientada al cliente, puesto que a los programadores de SCA "no les preocupa", limita este área de contacto con el cliente.

Y es en realidad en esta definición de la interfaz del software donde la empresa contacta durante mayor tiempo con el cliente. Las empresas comerciales intentan tener una estrategia clara, o deberían intentarlo, de mejora continua con los contactos con los clientes en la "usabilidad" del producto.

Podríamos exponer una serie de cuestiones para la evaluación por parte de una empresa, al intentar determinar el si el software libre puede ser una buena opción para implementar la estrategia tecnológica: **[Senf 2003]**

- ¿Existe en la empresa el nivel suficiente de experiencia para poder gestionar el mantenimiento y la aplicación de software de código abierto?
- ¿Puede ser un problema la limitación del soporte en la implementación y mantenimiento?
- ¿Es crítica la posible falta de documentación en los sistemas?
- Comparada con las soluciones propietarias para los requerimientos de la empresa ¿ las capacidades del software libre son mejores, peores o similares?

- ¿La funcionalidad y usabilidad del software libre es la necesaria para los usuarios finales?
- ¿Encaja la solución de software libre con los requerimientos empresariales ya definidos?
- ¿Cuáles son las implicaciones a corto y largo plazo de elegir la arquitectura y plataforma de código abierto?
- Con los planes actuales, y la estrategia futura, ¿alcanza la solución de software de código abierto las necesidades de negocio en estabilidad, rendimiento y escalabilidad?
- ¿Cuáles son los costos de administrar, mantener y adaptar el código abierto?
- ¿Las consideraciones legales del software libre son compatibles con los objetivos tecnológicos y cuestiones legales de la empresa?

Obviamente, algunas de estas cuestiones son importantes para la decisión de cualquier tipo de software, aunque sea comercial. Pero el desarrollo potencialmente incontrolado, sin una estrategia clara, y la posible falta de servicios, documentación, etc... en algunos proyectos, hace necesario un énfasis mayor.

Las empresas que adopten el software libre deben plantearse involucrarse en el desarrollo de los productos. De esta manera se pueden asegurar cierto control en las líneas de desarrollo del software, y sus versiones.

En el entorno empresarial es donde mayor implantación existe del software libre, seguramente por que ha sido introducido por los propios técnicos, que han ido observando la evolución de los productos y veían que podían confiar en ellos. Algunos de ellos más conocidos se conocen como la pila de productos LAMP: (Linux, Apache, MySQL y PHP).

	Producto
Sistema Operativo	Linux
Servidor WEB	Apache
Base de datos	MySQL
Lenguaje de programación de páginas dinámicas	PHP

Tabla 6: Pila de productos LAMP.

Como se observa en la tabla anterior, mediante estos productos disponemos de una plataforma completa sobre la que podemos crear nuestras propias aplicaciones en Internet, todo con software libre. De hecho, este modelo es usado en muchas empresas, que consiguen una potente infraestructura a bajo coste.

El mercado de los sistemas operativos es sin duda uno de los que mayor interés tienen para estudiar el comportamiento de los programas de código abierto. Veamos una tabla de porcentajes de uso en EEUU. y Reino Unido de sus principales cien empresas:

FORTUNE 100		
Operating System	**Sites**	**Users**
Solaris 8	32	Viacom, J&J,Boeing, Sears, Bank of America
Windows 2000	26	ExxonMobil, Merrill Lynch, Bank One
Linux	12	Disney, GM, Merck, Verizon, American Express
Solaris (other)	7	AT&T, Cisco, Citigroup, Du Pont
Windows Server 2003	5	TXU, Microsoft, Intel, EDS
Solaris 9	4	Wachovia, Procter & Gamble, Lockheed Martin
AIX	3	IBM, Lowes
HP-UX	3	Home Depot, Wal Mart, Hewlett-Packard
Windows NT/98	1	Kroger

FTSE 100		
Operating System	**Sites**	**Users**
Windows 2000	37	Hanson, Sainsbury, Unilever, Bank of Scotland
Linux	16	Johnson Matthey, Marks & Spencer, Granada
Solaris 8	14	C&W, Pearson, Reuters
Windows NT/98	8	Lloyds TSB, Next,
Windows Server 2003	7	Carnival, United Utilities, Exel
Solaris (other)	4	BT, WPP Group,
AIX	3	Barclays, Safeway
FreeBSD	2	Severn Trent, Xstrata
HP-UX	1	Vodafone

Tabla 7: Sistema operativo usado en las 100 primeras empresas de EEUU y Reino Unido*

Como se puede observar, los sistemas operativos propietarios (Solaris, de Sun y Windows, de Microsoft) son los más utilizados. Pero Linux ocupa un tercer puesto muy cerca de los mismos. Además, en los últimos meses se está produciendo una destacable migración hacia Linux por parte de empresas importantes **[Netcraft 2004]**.

Por tanto, ya hemos visto numerosas cuestiones que una empresa debe tener en cuenta para la elección de un modelo de negocio u otro. Básicamente primeramente debe plantearse la utilidad y el beneficio que le reportará cada producto a evaluar. Posteriormente debe plantearse la confianza que puede depositar en la situación del mercado de cada producto, sus proveedores, su expansión y su futuro.

Para muchos directivos es también una cuestión de confianza. Todavía no ven "con buenos ojos" al movimiento de software libre, aunque esto está cambiando rápidamente como lo demuestran las cifras de adopción de productos que hemos comentado anteriormente como Linux, Apache o MySQL. El evaluar si un producto de código abierto está lo suficientemente maduro como para poder utilizarlo con el menor riesgo es muy importante **[Senf 2003]**. Determinar si es el momento adecuado para la utilización de un producto de software libre puede depender de factores externos como son:

* Reproducido con permiso de Netcraft.

- Disponibilidad de soporte y su coste.
- Madurez del código y las pruebas sobre el mismo.
- Evolución de las versiones y planes de desarrollo del código abierto.
- Disponibilidad de herramientas auxiliares para el desarrollo y mantenimiento.
- Aceptación e inclusión por parte de la industria del software o hardware.
- Disponibilidad de soporte para la integración con el resto de sistemas.

Posteriormente, cuando se ha decidido la adopción de un software de código abierto, se debe evaluar la involucración en el desarrollo del código con recursos propios. Esto será una necesidad cuando se quieran cubrir necesidades particulares para atender un nicho del mercado o requerimientos especiales. Desde luego, sólo la posibilidad de poder hacerlo, es una ventaja posterior del software de código abierto que debe ser evaluada "a priori".

6.4. El SCA en la administración

Un segmento especial del mercado donde existen ahora numerosos movimientos es en las administraciones públicas. Las administraciones pueden obtener importantes ventajas del código abierto para compartir sus trabajos entre las mismas. El uso de software libre les puede permitir que los desarrollos sean comunes a varias administraciones y colaboren en su desarrollo. Esto puede tener un fuerte impacto en las empresas de servicios, y seguramente presenten una fuerte resistencia a estos movimientos.

El software libre se adapta perfectamente a la administración pública, porque podemos contar con dos requisitos en el desarrollo dentro de una administración: el hecho de que no hay intereses comerciales, y que cada administración debe adaptar los programas a sus particularidades. Por tanto, el código fuente abierto es una premisa básica **[Schmitz 2002]**.

En la Unión Europea se realizó un estudio de viabilidad para compartir datos y aplicaciones entre las administraciones. La idea es que todas las administraciones utilizan un tipo de software muy parecido, y que este se puede desarrollar en código abierto para que pueda ser compartido. El programa IDA **[IDA 2004]** (Intercambio de Datos entre las Administraciones) intenta fomentar la interoperabilidad de los datos y sistemas entre las distintas administraciones que existen en la Unión Europea.

En el boletín de Septiembre de 2004 del programa IDA, se comenta un estudio que fue publicado a principios de ese año por el Comité de Regiones Europeas "que proporciona un valioso vistazo a las mayores innovaciones de proyectos de *e-governmet*[♣] a nivel regional y local". En concreto uno de las conclusiones de este estudio es que "los productos basados en software de código abierto pueden ayudar a reducir los riesgos de la sobre-dependencia de los productos y proveedores comerciales, mientras que a su vez puede significar importantes ahorros en los costos del software."

[♣] e-governmet: "Gobierno electrónico". Se trata de acercar las administraciones al ciudadano a través de las nuevas tecnologías para facilitarle el acceso a los recursos o la realización de las tareas administrativas.

El programa IDA intenta establecer un conjunto de "buenas prácticas" para el desarrollo del e-governmet y de su acercamiento al ciudadano. En esta tarea se plantean el software libre como una buena manera de trabajar y compartir los conocimientos. Cualquier impulso desde la administración europea al software de código abierto, elegido por que responde de la mejor manera a sus necesidades, tiene una gran importancia al significar posiblemente una importante inversión en su desarrollo e implantación.

En España se han realizado varios estudios en la Comisión de Ciencia y Tecnología sobre el software libre y su implantación en la administración. Se han concretado en entrevistas con los mayores expertos del mundo en la materia para hacerse una idea de la situación actual.

Algunas Comunidades Autónomas han ido un paso más allá y tienen ya sus propuestas en el uso del software libre en sus administraciones. La comunidad de Extremadura, como ya se ha comentado antes, es pionera en este sentido. **[Ibermática 2003]** Se escogió un paquete ofimático y un sistema operativo libres, se mejoraron y se empaquetaron como una distribución particular. Ese paquete se devolvió a la red, y además se utilizó para instalar en cientos de ordenadores que iniciaban el proyecto de la "Red Tecnológica Educativa". Ahora, la administración se plantea el migrar sus sistemas a esta distribución de software, mantenida y soportada por la propia administración.

El sector de las administraciones públicas es muy importante para las compañías de software comerciales. El volumen de negocio que se maneja es muy significativo para empresas como Microsoft. Por tanto, se esfuerzan en no perder cuota significativa, por lo que intentan llegar a acuerdos comerciales que les obligan a bajar los precios.

7. Conflicto entre el modelo tradicional y el modelo basado en código abierto.

Las tecnologías de la información han dejado de ser una utilidad en las empresas, para convertirse en base fundamental para el negocio, proporcionando valor añadido y una inversión esencial. El debate de cómo sacar el máximo rendimiento a esas tecnologías siempre ha existido. Ahora, la irrupción del software libre como una alternativa real desde hace unos años, ha avivado este debate. La industria del software se ve enriquecida con las nuevas ideas.

El modelo de software comercial, con Microsoft como el mayor representante, pide a la comunidad de clientes que *se guíen por criterios técnicos y no políticos* **[Garcia 2002]** para la elección del software en las empresas. Hasta ahora, el mercado de las tecnologías ha tenido las condiciones adecuadas para que las compañías invirtiesen y creciesen. El modelo comercial ha permitido el rápido crecimiento de un sector que además ha innovado, invirtiendo gran parte de sus beneficios en esa innovación.

La innovación aparece como punto clave en el enfrentamiento entre estos dos modelos de negocio. ¿Qué modelo aporta mayores facilidades para la innovación y el desarrollo de las tecnologías? Cada uno lo realiza en distintas maneras. Sin duda, el software comercial ya ha demostrado que mediante la inversión de la riqueza generada se puede crear una gran fuente de innovación, con grandes presupuestos para la investigación y el desarrollo. El software libre basa su innovación más en la creatividad de las partes que componen la comunidad, y en lograr que esta comunidad sea cada vez más amplia y fuerte.

Pero no poca gente ve problemas de competencia con el nuevo modelo. El consejero delegado de Microsoft Ibérica, Francisco Román, aunque reconoce la *'transparencia y sentido de la comunidad'* que aporta el código abierto y el *'enriquecimiento del marco competitivo que impone'*, señala que su sistema de licencias *'corre el riesgo de contaminar toda una industria y hace inviable un modelo comercial'*.

La acusación de hacer inviable un modelo comercial puede ser cierta, entendida de manera que puede acabar con el modelo comercial *existente actualmente*. Pero pueden surgir otros modelos comerciales menos implicados con el sistema de licencias que se utiliza por la mayor parte de compañías, un sistema basado en servicios más que en productos.

Es muy importante distinguir la discusión técnica de la política. Las empresas deberían guiarse únicamente por motivos técnicos para la elección de sus proveedores de sistemas de información. La discusión política debería estar en otro ámbito, aunque la administración pública debe ser neutra en el apoyo a estas tecnologías. ¿por qué favorecer a unas empresas u otras? Al final, no olvidemos que tras el código abierto en un entorno empresarial, existirá alguna empresa dando unos determinados servicios. La externalización de los procesos de tecnología en muchas empresas, hace inviable a corto plazo la utilización de software libre soportado internamente, sin el soporte de una empresa externa.

Un punto muy importante para la confianza en el código abierto, es la amplitud de uso del propio programa. Los inicios serán más difíciles, y la credibilidad del sistema será más amplia cuanto más amplia sea su comunidad de usuario. En el modelo comercial se deposita la confianza en una empresa, de la que se conoce su credibilidad y respuesta ante las necesidades de los clientes.

Existen fanáticos del modelo basado en código abierto anunciando que todo el software debería estar desarrollado así, por cuestiones de ética y solidaridad. Creo que son exagerados, no observo ningún problema en que las empresas que desarrollan software le puedan poner un precio a su trabajo. No debemos olvidar que a menudo el desarrollo de un producto puede exigir importantes inversiones iniciales que deben ser recuperadas posteriormente. El modelo de licencias ha demostrado que es exitoso en crear beneficio, no sólo para la empresa en particular, si no para el mercado, por que potencia la innovación. Por ejemplo, Microsoft asegura que en España, por cada euro que factura ella, las empresas que colaboran con ellos ganan 12€, generando estos beneficios desarrollando productos y servicios sobre su plataforma **[García 2003]**. No parece una exageración estos datos, teniendo en cuenta la multitud de empresas que crean valor añadido sobre los productos de desarrollo de Microsoft, y lo extendido de su plataforma.

Numerosos seguidores del código abierto dan razones éticas para el desarrollo masivo del software libre, como Richard Stallman, que incluso promueve que todo el software debería ser libre **[Stallman 2004]**. Pero esto va más allá del razonamiento debido en una empresa, donde se ligue el planteamiento estratégico y las bases culturales y éticas. De ninguna manera parece reprobable moralmente el modelo comercial del software, puesto que no se aprovecha sino excepto del trabajo y la inversión para generar beneficios. Es una cuestión más personal la visión del software libre como modelo ético, que corresponde a cada persona decidir y escapa a un análisis empresarial.

Pero el problema de confrontación que existe entre los dos modelos de negocio, puede venir muy influenciado por el sentimiento anti-Microsoft que existe en numerosas comunidades de código abierto. La posición dominante le ha llevado a veces a abusar de ella con prácticas monopolísticas. Esto ha generado mucha crispación en determinadas personas que ven el software libre como una manera de terminar con el dominio de Microsoft. Esta posición me parece demasiado simplista.

Actualmente se están buscando acuerdos de colaboración entre empresas comerciales y proyectos de código abierto. Por ejemplo, IBM patrocina numerosos proyectos de software libre y Oracle se compromete en su publicidad a hacer su base de datos “irrompible” sobre Linux. En realidad el apoyo de las grandes empresas ha sido el disparador de la carrera por la utilización de software libre. En otro caso, determinados productos hubiesen seguido teniendo un cierto éxito en comunidades de informáticos, pero nunca hubiesen saltado a mercado del gran público o empresariales.

De todas maneras es importante observar que gran parte de los gastos en tecnología de las empresas no se van únicamente al pago de licencias, si no a la personalización y mejora de los productos ya existentes. Cada empresa debe evaluar cual va a ser el impacto de una “liberalización” del código de sus productos, y si es realmente interesante para su estrategia y objetivos empresariales.

8. Conclusiones

El software de código abierto está removiendo los cimientos del sector del software y parece que cada vez se va a extender más. Todas las grandes empresas se están posicionando alrededor de este movimiento. Existen grandes empresas, como IBM que le proporcionan un impulso muy fuerte, y otras como Microsoft, que parece que se posiciona contra él.

Ninguna empresa puede quedarse indiferente ante las perspectivas de que la competencia en su nicho pueda ser de otra índole totalmente diferente a la que llevaría a cabo otra empresa de carácter comercial. Este hecho supone una novedad tan importante en el mercado, que como ya se ha dicho, cambian los modelos de negocio.

Los seguidores del software libre a menudo lo defienden con el argumento de que debería ser un derecho para los usuarios el poder obtener y modificar el código fuente. Ciertamente, me resisto a creer que sea un "derecho natural". Los derechos sobre las copias tienen suficiente legitimidad como para ser una forma de distribución y licencias válida. Este es un punto clave, por que si se considerase de manera general que el software libre es la manera correcta, y subrayo, la manera *correcta*, no ya únicamente mejor, de crear el software, los gobiernos estarían obligados a legislar hacia su implantación. Pero ambos modelos deben coexistir, porque forman parte del mismo ecosistema de software mucho más amplio **[Smith 2003]**. El modelo comercial no supone la infracción de derechos de los clientes, ni el software libre es la solución a los problemas reales con los que se encuentra el desarrollo de la ingeniería de la informática.

Un punto fundamental para la creación de riqueza en una economía capitalista es la defensa de la propiedad privada y, por extensión de esta, de los derechos intelectuales. Las empresas, o un particular tienen el derecho de ceder el fruto de su trabajo bajo el modelo de negocio que más le interese o le convenga. La obligación de un modelo u otro por la legislación supondría una merma en los derechos de propiedad de los individuos.

Veamos a modo de resumen qué representa el nuevo modelo de negocio visto desde el punto de vista que se presentaba en la introducción

- La descripción de los clientes que tiene la empresa, los problemas que tienen y cómo es el mercado potencial.

  Los clientes ahora van a disponer del código fuente del software. Seguirán requiriendo en la mayor parte de los casos servicios de mantenimiento, pero podrían independizarse en cualquier momento y auto-mantenerse. Los clientes verán como se multiplican los suministradores de servicios por la libertad del código.

- La descripción de los productos y servicios, y cómo van a solucionar los problemas de los clientes.

  Los productos dejarán de proporcionar ingresos directamente por su posesión intelectual, puesto que no se tendrá derecho sobre ellos. Los servicios pueden ser proporcionados por cualquier empresa, puesto que el código fuente es libre.

- La descripción de como se van a proporcionar a los clientes los productos y/o servicios.

Los productos serán accesibles gratuitamente, y se pueden contratar o no servicios sobre ellos a las empresas existentes.

- La descripción de cómo se generan los ingresos y beneficios.

  Este será la mayor diferencia con el modelo tradicional. Aquí la venta de licencias por la utilización del software desaparece. El software es libre, y por tanto gratuito. Los servicios serán la principal fuente de ingresos.

La revolución al mundo del software ha llegado, mejor dicho, llegó, y está para quedarse. Pero esta revolución debe ser pacífica. El mercado debe decidir libremente qué opción le interesa más. A veces un producto comercial satisfacerá mejor las necesidades, y otras veces se valorará de mayor importancia la accesibilidad del código fuente.

A corto plazo, la dinamización del sector que se produce, vista como una competencia sana, logrará que las empresas comerciales bajen sus precios ante la presión de los nuevos competidores basados en el nuevo modelo de negocio. Es difícil saber que ocurrirá en el medio y largo plazo. En mi opinión, determinado sectores se orientarán a un modelo de negocio basado puramente en servicios. Las empresas consolidadas que ya han rentabilizado sus productos mediante el cobro de licencias podrán adaptarse a la nueva situación, reestructurando sus modos de trabajo, inversiones y organización. Pero puede ser preocupante la disminución en la investigación y desarrollo que se produzca en estos sectores, al disminuir los ingresos previstos por la innovación de ruptura que se lograría con la reinversión de los beneficios.

Sin duda el código abierto, o software libre, tiene una serie de ventajas frente al código propietario que lo hacen muy interesante como una opción para las administraciones públicas. Este campo es donde veo en un futuro cercano una mayor implantación de software libre a escala general. Las iniciativas de compartir software y datos entre las administraciones europeas podrían ahorrar millones de euros a los contribuyentes. Los problemas resueltos en una administración podrían ser fácilmente extrapolables a la comunidad entera.

Otros sectores serán más inaccesibles al código abierto, generado de manera altruista. Aquellos sectores que no tengan interés más que para un reducido número de gente, no sean capaces de crear una comunidad importante, y necesiten de una inversión en tiempo o recursos importante no verán crecer la competencia de código abierto. Estos nichos del mercado seguirán con pocas empresas, o una, ofreciendo un producto desarrollado que siempre irá por delante del software generado de código abierto, puesto que dispondrán de la experiencia para ello.

Y determinadas parcelas del mercado del software se verán en sana competencia con productos y servicios basados en software libre. Pero no olvidemos que las empresas que den servicios o valores añadidos sobre software libre, no pueden dejar de cumplir las reglas del mercado: ser rentables. Al final, además de las comunidades de desarrolladores que trabajan de manera desinteresada, también tendremos a empresas que intentan hacer negocio con los productos desarrollados.

La cuestión es: habrá en un futuro empresas creando el código abierto, o la gente que ahora se dedica de manera altruista a hacerlo seguirá haciéndolo. De alguna manera, la gente que no gana dinero creando este software es por que ya tiene otro medio de vida. Cuando grandes empresas se creen alrededor de este software, que no me cabe duda de que se seguirán creando ¿seguirá esta gente colaborando? ¿o en realidad se volverá en algún sentido al modelo comercial, pero sin el dinero de la venta de licencias?

Cualquier empresa de software existente debe plantearse como le afecta este nuevo modelo de negocio. Cualquier proyecto de compañía, debe averiguar cómo sacar el mayor rendimiento a este modelo de negocio. Existen empresas que desarrollan un modelo mixto, con productos comerciales sobre productos de código abierto, o viceversa. Si conseguimos unir lo mejor de ambos mundos, podremos obtener las ventajas competitivas de los dos. Pero esto sería merecedor de otro trabajo distinto.

[3] Esta obra está bajo una licencia Reconocimiento-NoComercial-SinObraDerivada de Creative Commons. Para ver una copia de esta licencia, visite http://creativecommons.org/licenses/by-nc-nd/2.1/es/ o envíe una carta a Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

9. Anexo: Bibliografía

[Abella 2004] Libro blanco del software libre en España 2004. A. Abella, J. Sánchez, M. A. Segovia. 2004 (http://www.libroblanco.com/)

[Alonso 2002] Modelos de Negocio Software Libre: Red Hat / VA Linux. Fco. Javier Alonso. Junio 2002 (http://curso-sobre.berlios.de/curso/trab/jalonso/ModelosNegocio.PDF)

[Barahona 2004] Software libre: licencias y propiedad intelectual. Jesús M. González Barahona, Joaquín Seoane Pascual, Gregorio Robles. Julio de 2004

[Baranda 2002] El software como servicio. O de como producir programas libres y no morir en el intento. Baranda. Octubre 2002. Revista TodoLinux nº 25.

[BSA 2003] Bussiness software Alliance. (http://www.bsa.org/espana/antipiracy/Internet-Piracy.cfm)

[Carr 2004] El crecimiento de la industria del software disminuye. Nicholas G. Carr. – traducido por Pilar Acosta/NYT. Noviembre 2004. (http://www.calipso.com/noticia.php3?nt=93)

[Choike 2004] Patentes de Software. Informe en Choike. (Una iniciativa del Instituto del Tercer Mundo: http://www.item.org.uy/) (http://www.choike.org/nuevo/informes/1064.html)

[Com 2002] Propuesta de DIRECTIVA DEL PARLAMENTO EUROPEO Y DEL CONSEJO sobre la patentabilidad de las invenciones implementadas en ordenador. Bruselas, 20.02.2002 - COM(2002) 92 final

[Delio 2003] El código abierto no es algo nuevo en la red. Michelle Delio. Wired News. Enero 2003 (http http://us.terra.wired.com/news/linux/0,1274,21105,00.html)

[DiarioTI 2004(1)] Ballmer insta a descartar Linux debido al riesgo de demandas. DiarioTI.com Noviembre 2004 (http://www.diarioti.com/gate/n.php?id=7723)

[DiarioTI 2004(2)] Microsoft comparte código. Diciembre de 2004 (http://ingenieroseninformatica.org/revista/noticias/mostrar_completa.php?id=359&inicio=10&verSec=4)

[DiarioTI 2004(3)] Microsoft: “Linux amenaza nuestros márgenes de ganancia”, Bill Gates. DiarioTI.com. Octubre 2004 (http://www.diarioti.com/gate/n.php?id=7485)

[García 2002] El modelo de Software Comercial. Rosa María García. Consejera delegada de Microsoft Ibérica. (Online: http://www.coit.es/publicac/publbit/bit142/rosa.pdf)

[García, Romeo 2004] Análisis financiero (cap.7) Juantomás García, Alfredo Romeo (http://www.lapastillaroja.org/capitulos_liberados_pdf/la_pastilla_roja_capitulo_7.pdf)

[Glass 2003] Facts and Fallacies of Software Engineering. Robert L. Glass. 2003 ISBN 0-321-11742-5

[Ibermática 2003] El despegue del Software libre. Desayunos Tecnología y conocimiento. Ibermática. (http://www.ibermatica.com/ibermatica/whitepapers/despeguesoftwarelibre)

[IDA 2004] Intercambio de Datos entre Administraciones, Programa europeo. (http://europa.eu.int/ida/) (http://europa.eu.int/ida/servlets/Doc?id=2007)

[Infoworld 2004] Sun to launch next version of Solaris. INFOWORLD. Noviembre 2004 (http://www.infoworld.com/article/04/10/29/HNsolarislaunch_1.html)

[Lakhani 2003] Why hackers do what they do: Understanding motivation and effort in Free/Open Source Software Projects. Karim R. Lakhani, Robert G. Wolf. 2003. (freesoftware.mit.edu/papers/lakhaniwolf.pdf)

[Martínez 1999] La empresa basada en Sw. libre. Juan Antonio Martínez, Abril 1999 (http://es.tldp.org/COMO-INSFLUG/es/pdf/La_empresa_ante_el_software_libre.pdf)

[Miralles 2003] "Software" de código abierto, de una iniciativa altruista a una fuente de negocios. Francesc Miralles, Guillermo Armelini. Euskotek. (número 22- 4º trimestre 2003) (http://www.rpte.net/euskotek/numero_22/Euskotek22.pdf)

[Mitre 2001] A Bussiness Case Study of Open Source Software. Mitre Corporation. Julio 2001. (http://www.mitre.org/work/tech_papers/tech_papers_01/kenwood_software/)

[Montoro 2004] ¿Es negocio el software libre? Sergio Montoro. Mayo 2004 (http://www.baquia.com/com/20040527/art00003.html)

[Netcraft 2004] Solaris remains TopChoice Among Fortune 100. Netcraft. (http://news.netcraft.com/archives/2004/11/16/solaris_remains_top_choice_among_fortune_100.html)

[Ordovas 2004] Historia y cultura del Software Libre. Jorge Bernal Ordovas, Héctor Blanco Alcaine, Isaac Clerencia Pérez (http://pulsar.unizar.es/gluz/manual-sl/ 2004)

[Raymond 1998] La catedral y el bazar.Eric S. Raymond. Eero 1998 (ttp://es.tldp.org/Otros/catedral-bazar/cathedral-es-paper-00.html)

[Santos] Repercusión económica del Software Libre. Roberto Santos. Hispalinux (http://www.hispalinux.es/~rsantos/repercusion-socioeconomica-del-software-libreV02.pdf)

[Schadler 2004] Open Source rewards outweigh the risks. Ted Schadler. 2004 (http://www.brodeur.de/de/magazin/index_449.htm)

[Schmitz 2002] Compartir Software de código abierto. Estudio de Viabilidad IDA. Patrice-Emmanuel Schmitz, Sébastien Castiaux. Junio 2002

[Senf 2003] Is Open Source for Your IT Strategy? 10 essential Questions. David Senf and Warren Shuiau. IDC (http://www2.cio.com/analyst/report1489.html)

[Shirky 2002] Clay Shirky's Writings About the Internet (http://www.shirky.com/)

[Smith 2003] El futuro del software: ofrecer al mercado la posibilidad de elegir. Bradford L. Smith, Consejero General de Microsoft. Marzo 2003 (http://www.microsoft.com/spain/sharedsource/Articles/Future.mspx&e=747)

[Spolsky 2001] Los Buenos Programas Toman Diez Años. Acostúmbrense. Joel Spolsky. Código maduro. 2001. (http://spanish.joelonsoftware.com/Articles/GoodSoftwareTakesTen.html)

[Stallman 2004] Software libre para una sociedad libre. Richard M. Stallman. ISBN: 84-933555-1-8. Editorial Traficantes de Sueños.

[Wheeler 2003] How to evaluate Open Source Software / Free Software Programs. David A. Wheeler. Noviembre 2003 (http://www.dwheeler.com/oss_fs_eval.html)