

# Software libre

Malcolm Bain  
Manuel Gallego Rodríguez  
Manuel Martínez Ribas  
Judith Rius Sanjuán

XP06/M2114/02160



## Aspectos legales y de explotación del software libre Parte II

### David Megías Jiménez

Coordinador

Ingeniero en Informática por la UAB.  
Magíster en Técnicas Avanzadas de Automatización de Procesos por la UAB.

Doctor en Informática por la UAB.  
Profesor de los Estudios de Informática y Multimedia de la UOC.

### Jordi Mas

Coordinador

Ingeniero de software en la empresa de código abierto Ximian, donde trabaja en la implementación del proyecto libre Mono. Como voluntario, colabora en el desarrollo del procesador de textos Abiword y en la ingeniería de las versiones en catalán del proyecto Mozilla y Gnome. Es también coordinador general de Softcatalà. Como consultor ha trabajado para empresas como Menta, Telépolis, Vodafone, Lotus, eresMas, Amena y Terra España.

### Malcolm Bain

Autor

Abogado inglés. Máster en Relaciones Internacionales y Derecho Europeo de la UB. Especializado en el derecho de las tecnologías de la información y la comunicación. Participa en varios proyectos de investigación relacionados con ellas, bajo el auspicio de la Unión Europea.

### Manuel Gallego Rodríguez

Autor

Licenciado en Derecho por la Universitat Pompeu Fabra y Máster en Derecho de la Empresa. En la actualidad, trabaja como abogado en la firma MALET & GONZÁLEZ DE CARVAJAL de Barcelona, especializado en el área mercantil: Operaciones Societarias, Contratación mercantil y Nuevas Tecnologías.

### Manuel Martínez Ribas

Autor

Abogado en BAKER & MCKENZIE. Coordinador y responsable de proyectos europeos. Conferenciante habitual en Barcelona, Madrid (ESADE, IESE, Abad Oliva, Fomento, Recoletos, Cambra Barcelona, Institut Català de Technologies, Internet Global Conference, Univ. Pompeu Fabra, Univ. Politècnica Barcelona, Univ. Navarra), Bruselas, París (Univ. Paris), Fontainebleau (INSEAD), Darmstadt, Munich, Estocolmo, Amsterdam, Milán, Roma, Londres, Zurich. Autor de varias publicaciones de comercio electrónico.

### Judit Rius Sanjuán

Autora

Licenciada de Derecho y Master en Estudios Internacionales por la Universitat Pompeu Fabra. Beca de La Caixa para estudiar un Master en derecho, ciencia y tecnología en la Universidad de Stanford. Abogada en ejercicio, especializada en la asesoría a empresas del sector informático y farmacéutico, ha colaborado en proyectos de investigación subvencionados por la Unión Europea y es coautora de un estudio sobre Derecho y Comercio Electrónico en España.

Segunda edición: febrero 2007

© Fundació per a la Universitat Oberta de Catalunya

Av. Tibidabo, 39-43, 08035 Barcelona

Material realizado por Eureka Media, SL

© Autores: Malcom Bain, Manuel Gallego Rodríguez, Manuel Martínez Ribas y Judit Rius Sanjuán

Depósito legal: B-13.406-2007

Se garantiza permiso para copiar, distribuir y modificar este documento según los términos de la *GNU Free Documentation License*, Version 1.2 o cualquiera posterior publicada por la Free Software Foundation, sin secciones invariantes ni textos de cubierta delantera o trasera. Se dispone de una copia de la licencia en el apartado "GNU Free Documentation License" de este documento.

## Índice

<b>6. Cláusulas esenciales en las licencias de software propietario y libre</b>	<b>351</b>
6.1. Consideraciones generales	353
6.2. Derechos, prohibiciones y limitaciones	
en las ciencias de software propietario	356
6.2.1. Derecho de uso del software	357
6.2.2. Derecho (prohibición) de copia. La copia de seguridad	365
6.2.3. Derecho (prohibición) de modificación	367
6.2.4. Derecho (prohibición) de distribución	374
6.3. Derechos y libertades en las licencias de software libre	376
6.3.1. Libertades de uso y de copia	378
6.3.2. Libertad de modificación	380
6.3.3. Libertad de distribución	384
6.4. Garantías y responsabilidades	388
6.4.1. Consideraciones generales	388
6.4.2. Distinción entre garantías y responsabilidades	391
6.4.3. Limitaciones y exclusiones de garantías y responsabilidades en las licencias de software	398
6.4.4. Especial referencia a las exclusiones de garantías y responsabilidades en las licencias de software libre	402
6.5. Otras cláusulas	406
6.5.1. Jurisdicción competente y legislación aplicable	406
6.5.2. Pactos de confidencialidad	409
6.5.3. Cláusulas relativas a patentes	411
6.6. Conclusiones	412
<b>7. Licencias de software libre</b>	<b>415</b>
7.1. Aspectos generales de las licencias libres	416
7.1.1. La libertad en el software	417
7.1.2. Criterios de clasificación: apertura y libertad	420
7.2. Unos mitos legales para desmitificar	430

7.2.1. "El copyleft está en contra o no respeta el derecho de autor" .....	430
7.2.2. "El software libre no tiene titulares o propietarios u obliga a ceder sus derechos de autor" .....	432
7.2.3. "No se puede hacer un uso comercial del software libre" .....	433
7.2.4. "El software libre y el software propietario son incompatibles" .....	433
7.2.5. "No se puede integrar o mezclar código libre y código propietario" .....	433
7.2.6. "Todo el software libre es igual, bajo los términos de la GPL" .....	436
7.2.7. "Las licencias libres obligan a publicar sus modificaciones particulares" .....	436
7.2.8. "Nadie es responsable por el software libre, ni tiene garantía" .....	437
7.3. Estudio particular de las licencias de software libre .....	438
7.3.1. Las licencias de software libre con copyleft ....	440
7.3.2. Las licencias sin copyleft robusto, compatibles con la GPL .....	460
7.3.3. Las licencias sin copyleft incompatibles con la GPL .....	465
7.4. Otras licencias de tipo "libre" .....	474
7.4.1. Las licencias de software "seudolibres" .....	475
7.4.2. Las licencias de documentación libre .....	479
7.4.3. Licencias de tipo freeware y shareware .....	485
7.5. Conclusiones .....	485
<b>8. Los efectos prácticos de las licencias de software libre</b> .....	<b>489</b>
8.1. Algunos temas legales relacionados con las licencias .....	490
8.1.1. Elegir una licencia libre .....	491
8.1.2. Las licencias sobre las contribuciones y autoría .....	495
8.1.3. La compatibilidad entre licencias .....	497
8.1.4. Las licencias duales o múltiples .....	498
8.1.5. Cambio de una licencia a otra: problemas y consecuencias .....	499
8.1.6. Licencias libres y la división de software libre ( <i>forking</i> ) .....	500
8.1.7. Resumen de potenciales problemas legales de las licencias libres .....	503

8.2. Licencias libres y otras ramas de derecho .....	505
8.2.1. Licencias libres y el derecho de la competencia .....	506
8.2.2. Licencias libres y licencias de patente .....	509
8.2.3. Licencias libres y secreto comercial .....	510
8.2.4. Licencias y marcas .....	512
8.2.5. Licencias y estándares .....	513
8.3. Los datos personales y la protección de la intimidad .....	514
8.3.1. Introducción y marco legal .....	515
8.3.2. El régimen legal de protección de los datos personales .....	516
8.3.3. Marco legal en otras jurisdicciones .....	528
8.3.4. Datos personales y software libre: la seguridad .....	529
8.4. El software libre y los controles sobre los productos de seguridad .....	534
8.4.1. Seguridad y la sociedad de la información ...	534
8.4.2. Diferentes tipos de protocolos y aplicaciones de seguridad .....	537
8.4.3. Los controles sobre los productos de seguridad .....	539
8.4.4. Conclusiones sobre el software libre y los controles de seguridad .....	548
8.5. Conclusiones .....	548
<b>9. Aspectos relevantes para la creación     de software (estudio de caso) .....</b>	<b>551</b>
9.1. QDR-Soft. La selección de una licencia de software libre .....	554
9.1.1. Introducción .....	555
9.1.2. Modelos de desarrollo y licencias .....	555
9.1.3. La empresa QDR-Soft .....	556
9.1.4. Proyecto actual .....	564
9.1.5. Preguntas y actividades .....	565
9.2. Iván Lasser. Desarrollador .....	567
9.2.1. Introducción .....	568
9.2.2. Iván Lasser .....	568
9.2.3. Controversia .....	569
9.2.4. Preguntas y actividades .....	573
9.3. LibreSolutions. La distribución de software libre .....	574
9.3.1. Introducción .....	574
9.3.2. La empresa LibreSolutions: el concepto .....	577
9.3.3. Conclusión .....	584
9.3.4. Preguntas y actividades .....	585

9.4. MySQL – Progress Software .....	587
9.4.1. Introducción .....	587
9.4.2. Telón de fondo .....	588
9.4.3. MySQL .....	590
9.4.4. Progress Software .....	592
9.4.5. El litigio .....	594
9.4.6. Conclusiones .....	595
9.4.7. Preguntas y actividades .....	595
9.5. Recursos humanos y comunidades de desarrollo .....	597
9.5.1. Introducción .....	598
9.5.2. QDR-Soft .....	599
9.5.3. LibreSolutions .....	603
9.5.4. Conclusiones .....	604
9.5.5. Preguntas y actividades .....	604
<b>10. Aspectos relevantes para la implantación de software (estudio de caso) .....</b>	<b>607</b>
10.1. El Hospital Beaumont .....	609
10.1.1. Introducción .....	609
10.1.2. El Hospital Beaumont y sus sistemas informáticos .....	609
10.1.3. Un cambio de enfoque .....	610
10.1.4. Las aplicaciones nuevas .....	611
10.1.5. La situación actual .....	614
10.1.6. Los aspectos financieros (estimados) .....	615
10.1.7. Conclusiones .....	616
10.1.8. Preguntas y actividades .....	616
10.2. Lorena Nadal .....	618
10.2.1. Introducción .....	618
10.2.2. Lorena Nadal .....	619
10.2.3. Preguntas y actividades .....	619
10.3. Kerberos .....	620
10.3.1. Introducción .....	621
10.3.2. Kerberos .....	621
10.3.3. La distribución de Kerberos .....	622
10.3.4. La versión de Windows 2000 .....	623
10.3.5. El debate .....	624
10.3.6. Conclusiones .....	626
10.3.7. Preguntas y actividades .....	627
10.4. El software libre y la protección de datos .....	628
10.4.1. Introducción .....	628
10.4.2. Preguntas .....	629
10.5. Trusted Computing o la “informática fiable” .....	629
10.5.1. El concepto de Trusted Computing (TC) .....	630

10.5.2. Ejemplos: gestión de derechos y Palladium .....	631
10.5.3. La relación con el software libre .....	632
10.5.4. Preguntas .....	634
10.6. La exportación de productos de seguridad .....	634
10.6.1. Introducción .....	634
10.6.2. Algunos productos de software libre para la seguridad .....	635
10.6.3. Preguntas y actividades .....	636
10.7. El caso SCO .....	637
10.7.1. Introducción .....	638
10.7.2. Telón de fondo: el desarrollo de sistemas operativos .....	638
10.7.3. El desarrollo del software UNIX original y de Linux .....	639
10.7.4. El desarrollo de GNU/Linux .....	642
10.7.5. Los primeros pasos del litigio .....	644
10.7.6. El litigio entre SCO e IBM .....	645
10.7.7. Los alegatos de SCO contra la comunidad Linux .....	648
10.7.8. Otras consideraciones y conclusiones ....	655
10.7.9. Preguntas y actividades .....	656
<b>GNU Free Documentation License .....</b>	<b>659</b>





## 6. Cláusulas esenciales en las licencias de software propietario y libre

Al estudiar las licencias de software, y en particular las licencias de software libre, debemos analizar su contenido esencial: es decir, los derechos que el proveedor concede al usuario sobre el software, así como las limitaciones y prohibiciones que el usuario debe respetar al utilizarlo.

Tanto en el software propietario como en el libre, la licencia de uso es el instrumento legal por el cual el proveedor permite el uso del software a terceros, los usuarios. Pero el esquema de derechos y limitaciones cambia completamente según si se trata de una licencia de software propietario o de software libre. Las licencias de software propietario se basan en conceder al usuario un derecho restringido al uso y copia del software, así como en prohibirle su modificación y distribución, derechos que el proveedor se reserva en exclusiva. En cambio, las licencias de software libre no se basan en proteger derechos de exclusiva del proveedor, sino en conceder y asegurar a los usuarios libertades de uso, modificación y distribución sobre el software. Veremos cómo las licencias de software, propietario y libre, contemplan estos derechos y limitaciones.

Asimismo, analizaremos otro punto importante en estas licencias: las garantías y responsabilidades del proveedor sobre el software. Nos centraremos en determinar la validez o no de las limitaciones de responsabilidad que el proveedor suele imponer en sus licencias, en particular las cláusulas de exoneración o limitación de garantías y de responsabilidades (conocidas en inglés con el término *disclaimers*) en las licencias de software libre.

Por último, trataremos la cuestión de la jurisdicción competente y derecho aplicable en las licencias de uso de software: la posibilidad y los límites que tienen las partes de la licencia en acordar los tribunales de qué país resolverán sus posibles litigios, y bajo el derecho de qué país se registrará la licencia.

Con el estudio de esta unidad de aprendizaje, el lector alcanzará los objetivos siguientes:

1. Obtener una visión general de las distintas motivaciones existentes en las licencias de software propietario y las licencias de software libre, que condicionan el diferente esquema de derechos que se atribuyen al usuario-licenciatario, así como las restricciones y prohibiciones que éste debe respetar.
2. Conocer los derechos, restricciones y prohibiciones habituales en una licencia de software propietario.
3. Tener presente que existen unos derechos mínimos que la Ley de la Propiedad Intelectual concede al usuario-licenciatario, que actúan como un límite a los derechos exclusivos del proveedor-licenciante.
4. Conocer el distinto enfoque de las licencias de software libre con relación a los derechos que se conceden sobre el software: facilitar y asegurar a los usuarios toda una serie de libertades.
5. Analizar las libertades que se conceden a los usuarios-licenciatarios en las licencias de software libre: libertades de uso, reproducción, modificación y distribución.
6. Conocer las limitaciones y requisitos que deben cumplir los usuarios de software libre.
7. Analizar las garantías que el proveedor-licenciante ha de prestar sobre el software; en particular, la continuidad en el buen funcionamiento del software. Asimismo, conocer la responsabilidad del licenciante-proveedor del software.
8. Examinar las limitaciones habituales a dichas garantías y responsabilidades que se incluyen en las licencias de software (cláusulas de exoneración o limitación de garantías y responsabilidades, conocidas como disclaimers), así como sus posibles problemas de legalidad.
9. Conocer, en particular, las cláusulas de exoneración de garantías y de responsabilidades en las licencias de software libre, y analizar su legalidad conforme a la legislación española.
10. Recibir una aproximación a la cuestión, en las licencias de software, a las cláusulas de elección del derecho aplicable a la li-

cencia y de la jurisdicción competente para resolver un litigio entre las partes. Asimismo, conocer sumariamente otras cláusulas frecuentes en las licencias de software: pactos de confidencialidad y cláusulas relativas a patentes (en particular, ciertas cláusulas relativas a patentes en algunas licencias de software libre).

## 6.1. Consideraciones generales

En las unidades anteriores hemos visto que la licencia de software es el instrumento legal, el contrato, por el cual el proveedor del software permite su utilización a terceros, los usuarios.



Cabe recordar que en España y en los países de derecho continental, la licencia se concibe exclusivamente como un contrato. Sin embargo, en los países anglosajones (EE.UU., Reino Unido) es posible y válida una licencia de software a partir de la mera declaración unilateral del proveedor de software.

La **licencia de uso** tiene, por tanto, un contenido esencial:

- Establece los **derechos** que el usuario concede sobre el software al usuario: qué es lo que el usuario va a poder hacer con el software.
- Asimismo, impone unas **prohibiciones y limitaciones** a los derechos del usuario, que éste debe respetar: qué es lo que el usuario no va a poder hacer con el software.

Por otra parte, sabemos que la **legislación sobre propiedad intelectual**, en España la Ley de la Propiedad Intelectual, reconoce al proveedor del software (en particular, el autor o titular de sus derechos de explotación) una serie de importantes derechos en exclusiva: derecho a usar, modificar y distribuir el software. Él es quien decide a quién, cuándo y cómo autoriza a utilizar el software.

### Nota

Podéis ver los derechos de autor que contempla la Ley de la Propiedad Intelectual en España en la unidad 2 de este curso.



De este modo, cualquier utilización del software por alguien que no sea su titular, sin que éste lo haya permitido expresamente, es un uso prohibido por la propia ley.

En este sentido, además de los derechos exclusivos que la Ley de la Propiedad Intelectual contempla a favor del software, el último párrafo del artículo 99 de la Ley de la Propiedad Intelectual establece lo siguiente:

“cuando se produzca cesión del derecho de uso de un programa de ordenador se entenderá, salvo prueba en contrario, que dicha cesión tiene carácter no exclusivo e intransferible, presumiéndose, asimismo, que lo es para satisfacer únicamente las necesidades del usuario.”

Por tanto, la propia legislación da por supuesto que las licencias de uso de software se conceden al usuario (si no dicen lo contrario) con carácter:

- 1) **No exclusivo:** ciertamente, los proveedores de software conceden licencias de uso a los usuarios sin exclusividad. Es decir, no conceden el derecho de usar el software a sólo un usuario, sino a una multitud de usuarios. Si las concedieran con exclusividad, únicamente un usuario podría utilizar el software, mientras que los proveedores persiguen justamente lo contrario, maximizar la divulgación de su software.
- 2) **Intransferible:** el usuario no puede transmitir su licencia a terceros, lo que lleva aparejada la prohibición de vender, alquilar, prestar o regalar su copia salvo que cuente con la autorización expresa del proveedor.
- 3) **Únicamente para satisfacer las necesidades del usuario:** sin autorización expresa, el usuario sólo puede emplear el software para su estricto uso personal, y no para prestar servicios a terceros (como en el caso de las licencias especiales para *outsourcing*).

#### Nota

Podéis ver las licencias especiales para *outsourcing* analizadas en el apartado 4.5.3.

Las empresas que desarrollan **software propietario** se benefician, precisamente, de los derechos de explotación exclusivos que les reconoce la ley, y basan su modelo de negocio en obtener el máximo

rendimiento económico de su software a partir de la comercialización de copias.



Por ello, las **licencias de software propietario** tienen tradicionalmente un contenido y alcance muy restrictivo de las facultades que sobre el software se conceden al usuario, así como muy protector de los derechos en exclusiva del autor: el uso del software se permite, pero de manera restringida, y al usuario se le prohíbe la modificación y la redistribución del software, derechos que el proveedor se reserva para sí.

Por otra parte, tenemos las licencias de software libre. La licencia es también el instrumento legal para distribuir el software libre, y asimismo, se basa en la existencia de derechos de autor.



Sin embargo, las **licencias de software libre** no se emplean para proteger y beneficiarse el proveedor-licenciante de los derechos exclusivos de explotación, sino para **conceder y asegurar a los usuarios las libertades de uso, modificación y redistribución** (con o sin modificaciones) y redistribución sobre el software.

Además, en las licencias de software libre las limitaciones a los derechos del usuario son mucho menores que las que estipulan las licencias de software propietario y, casi siempre, consisten en “condiciones” para el ejercicio de los derechos. Tales limitaciones se imponen al usuario, no para proteger derechos de explotación exclusivos del proveedor, sino la reputación de éste; así como para asegurar a los usuarios las libertades de uso, modificación y distribución en todo momento, frente a cualquier intento de “apropiación” ilegítima del software.

En definitiva, las licencias de software propietario y las licencias de software libre presentan unos contenidos muy diferentes en cuanto a los derechos, prohibiciones y limitaciones del usuario sobre el software.

A continuación, vamos a analizar con detalle esos distintos esquemas de derechos, prohibiciones y limitaciones.

## 6.2. Derechos, prohibiciones y limitaciones en las ciencias de software propietario

Ya sabemos que las licencias de software propietario acostumbran a conceder al usuario muy pocos derechos sobre el software. En realidad, el proveedor se limita a proporcionar al usuario una copia del software en archivos binarios o ejecutables (código objeto) junto con el derecho de usar el software, un derecho de uso que suele estar sujeto a múltiples y variadas restricciones.

Asimismo, la mayoría de licencias de software propietario prohíben al usuario la modificación o cualquier forma de distribución del software. Como mucho, conceden estos derechos para casos excepcionales y puntuales.

Asimismo, hemos visto que los proveedores de software pueden estipular unas licencias de software tan restrictivas para los usuarios gracias a la legislación sobre propiedad intelectual (en España, la Ley de la Propiedad Intelectual), que les concede importantes y amplios derechos en exclusiva sobre el software.

Sin embargo, la propia Ley de la Propiedad Intelectual contiene algunas excepciones a los derechos exclusivos del proveedor: éste no podrá imponer determinadas prohibiciones o restricciones de uso al usuario “legítimo”, es decir, a quien haya suscrito una licencia válida sobre el software.

No obstante, algunas de dichas excepciones previstas por la Ley de la Propiedad Intelectual a favor del usuario pueden no aplicarse en una licencia si así lo acuerdan las partes. Esto significa que, en la práctica, el proveedor del software (que, recordemos, impone el clausulado de la licencia al usuario como condiciones generales) suele establecer en la licencia todas aquellas limitaciones y prohibiciones que satisfagan sus intereses, incluidas aquellas que deben constar expresamente en el texto de la licencia.

### Nota

Podéis ver las excepciones a los derechos exclusivos del proveedor en el apartado 2.6.

### 6.2.1. Derecho de uso del software



Como es obvio, por la propia denominación de *licencias de uso*, el **derecho a utilizar el software** es la facultad característica y principal que el proveedor concede al usuario. En las licencias de software propietario, el derecho de uso es prácticamente el único que adquiere el usuario.

Al tratar la función jurídica y económica de las licencias de software, citábamos, entre las características especiales del software, que la utilidad y el valor añadido que reporta al usuario se hallan fundamentalmente en la aplicación o resultado técnico que se consigue con su ejecución, esto es, en el provecho que el usuario puede obtener al utilizar el software. Para el usuario puede resultar provechoso modificar el software o redistribuirlo; pero, ante todo, es el uso del software lo que le reporta un provecho, pues le es útil dicha aplicación o resultado técnico que conlleva su uso.

Así pues, el software propietario se distribuye en el mercado con la idea de que se van a adquirir copias por parte de personas que ya saben que únicamente van a contar con un derecho limitado de uso. Pero, a pesar de todas las limitaciones y prohibiciones que se contemplan en las licencias, para esos usuarios poder utilizar el software es “suficiente”.

Cualquiera que pretenda usar un software, para beneficiarse de su utilidad o aplicaciones, necesita la autorización del proveedor del software, que la concede mediante la licencia de uso. El proveedor de software es quien se halla facultado para permitir el uso del mismo software al usuario, en tanto que es el titular exclusivo del **derecho de reproducción** sobre el programa informático; o, al menos, tiene la autorización del titular para conceder licencias: caso de los distribuidores de software.



Concretamente, el artículo 99 a) de la Ley de la Propiedad Intelectual reconoce al autor o al titular de los de-

#### Nota

Ver la función jurídica y económica de las licencias de software en el apartado 4.2.

#### Nota

Podéis ver el caso de los distribuidores de software en el apartado 5.1.

rechos de explotación el derecho exclusivo a realizar o autorizar la:

“reproducción total o parcial, incluso para uso personal, de un programa de ordenador, por cualquier medio y bajo cualquier forma, ya fuere permanente o transitoria. Cuando la carga, presentación, ejecución, transmisión o almacenamiento de un programa necesiten tal reproducción, deberá disponerse de autorización para ello, que otorgará el titular del derecho.”

### Contenido del derecho de uso

Quien adquiere válidamente una licencia de software, pasa a ser “**usuario legítimo**” del software: puede emplear el software para beneficiarse de sus aplicaciones y así realizar la función u obtener el resultado que dicho software le permite, según la finalidad para la que se ha creado.

Entendemos que este **derecho de uso** se compone principalmente de las siguientes acciones:

- a) **Instalación** del software en un equipo hardware.
- b) Ligado a lo anterior, el **almacenamiento** de los archivos necesarios en la unidad de memoria (casi siempre, el disco duro del ordenador) del elemento hardware. Esto conlleva un derecho inherente y necesario a copiar el software (o los archivos necesarios) en el elemento hardware.
- c) **Carga y ejecución** del software en cada momento que decida el usuario, para beneficiarse de sus aplicaciones.

### Contenido mínimo del derecho de uso

Como hemos visto, la Ley de la Propiedad Intelectual contiene algunas excepciones a los derechos exclusivos del titular de los derechos de explotación sobre el software: si éste concede licencias



sobre el software, no puede imponer determinadas prohibiciones o restricciones a los que hayan adquirido dicha licencia, los usuarios legítimos.

En este sentido, son tantas y tan importantes las limitaciones a ese derecho que los proveedores han venido imponiendo en las licencias “propietarias”, que la propia Ley de la Propiedad Intelectual establece una protección mínima a favor del usuario legítimo, estipulando que éste puede llevar a cabo ciertas acciones con el software sin que el proveedor se las pueda prohibir. Con ello se pretende evitar que el derecho de uso pueda desnaturalizarse del todo en una licencia, careciendo del contenido mínimo para el usuario:

- a) El usuario legítimo no necesita autorización del proveedor para **reproducir** el software –esto es, cargar, ejecutar, transmitir y almacenarlo– **cuando sea necesario para la utilización del mismo**, con arreglo a su finalidad propuesta (art. 100.1 LPI).

**Nota**

Entendemos que esto resulta obvio: si el derecho de uso sobre el software es la facultad esencial en toda licencia, el usuario debe tener asegurada la posibilidad de utilizar el software una vez acepte la licencia, durante la vigencia de la misma y mientras cumpla sus términos y condiciones.

- b) Además, mientras ejecuta el programa, el usuario legítimo está facultado para **observar, estudiar o verificar** su funcionamiento, con el fin de determinar las ideas y principios implícitos en cualquier elemento del programa (art. 100.3 LPI).

Este derecho del usuario legítimo a observar y estudiar el funcionamiento del programa mientras lo usa sería también inherente al propio derecho de uso. Además, es consecuencia lógica del artículo 96.4 de la LPI: el proveedor del software no tiene protegidos por los derechos de autor “las ideas y principios en los que se basan cualquiera de los elementos de un programa de ordenador, incluidos los que sirven de fundamento a sus interfaces”.



Distinto es que este derecho le resulte siempre provechoso al usuario. Con el mero estudio del software durante su ejecución, muchas veces le resultará imposible acceder a las ideas y principios subyacentes al programa: para ello, puede resultarle imprescindible el código fuente del software (que el proveedor de software propietario no entrega ni tiene obligación de entregar), o el derecho a descompilar el programa (que la Ley de la Propiedad Intelectual permite sólo para obtener la interoperabilidad con otro programa).

### Restricciones al derecho de uso

Si bien mediante la licencia el proveedor permite al usuario utilizar el software, también es cierto que las licencias de software propietario incluyen múltiples restricciones y limitaciones de uso. Estas **restricciones y limitaciones** tienen casi siempre como objeto procurar la comercialización del mayor número de copias del software, para obtener el mayor rendimiento económico posible del mismo.

Entre las restricciones y limitaciones más comunes, podemos destacar las siguientes:

- a) **Uso personal:** sabemos que la propia Ley de la Propiedad Intelectual presupone que una licencia de uso de software se concede únicamente para satisfacer las necesidades del usuario.

Los proveedores de software propietario suelen, efectivamente, reafirmar en las licencias que éstas se conceden para “uso personal”. Ello supone una restricción del derecho de uso, pues impide al usuario emplear el software para prestar servicios a terceros.

Además, algunas licencias de software propietario utilizan la fórmula “uso personal”, “licencia personal” o “edición personal” para llegar a restringir, incluso, la utilización del software por cualquier persona distinta al que ha adquirido la licencia. Si bien esto puede parecer aceptable (dentro de la lógica del software

propietario, en el que se prohíbe al usuario la redistribución con el objetivo de maximizar la comercialización de copias), también puede parecer excesivo y abusivo cuando se trata de impedir el uso del software por una persona allegada al usuario, desde el mismo equipo hardware donde lo tiene instalado el usuario-licenciatarario (el usuario “legítimo”).

Llevar la restricción de “uso personal” hasta su máximo extremo supone que, por ejemplo, si un usuario consumidor instala un software en un ordenador de su casa, ningún otro miembro de su familia podría utilizarlo. Asimismo, en el caso de un usuario que sea empresa, tal restricción no sólo le obliga a instalar el software en un único ordenador, sino también a designar a un empleado por copia de software como el único autorizado para usarlo. En cualquier caso, al proveedor le es muy difícil controlar el cumplimiento de esta restricción por parte del usuario.

Así, las licencias de programas de Microsoft® suelen centrarse en restringir la posibilidad de usar el software en más de un equipo, antes que en restringir el uso a más de una persona. No obstante, algunas de sus licencias sí que se otorgan expresamente con carácter personal, como las del programa Microsoft Visual Foxpro Edición Profesional®:

“Microsoft le otorga a usted **como individuo** una licencia **personal**, no exclusiva, para que haga y use copias del SOFTWARE [...]. Si usted es una entidad, Microsoft le otorga el derecho de designar a un individuo dentro de su organización para que tenga el derecho a usar el SOFTWARE [...].”

Este tipo de cláusulas no es propio únicamente de un software estándar comercializado en masa por una gran compañía desarrolladora. En licencias de uso de un software de aplicaciones más especializado, con destino a empresas, también es frecuente incluir estipulaciones que prohíben a esa empresa licenciataria utilizar el software con fines de *outsourcing*, con un contenido tal y como el que sigue:

“El LICENCIATARIO no podrá utilizar el SOFTWARE para prestar a terceros servicios de procesamiento de datos, tales como el uso

comercial en una oficina de servicios, en un sistema de uso compartido (multiusuario), mediante uso remoto u otras operaciones comerciales similares [...].”

- b) **Uso en un solo equipo hardware:** es muy habitual que el proveedor del software conceda la licencia para que el usuario ejecute el software en un único equipo.

Esta restricción obliga al usuario a adquirir una licencia de software por cada uno de los equipos hardware donde pretenda utilizarlo. Sobre todo, se pretende evitar con ello el uso simultáneo de una única copia del software en diferentes equipos, como puede ocurrir en la red local de una empresa. De este modo, se fuerza al usuario (de hecho, incluso se le exige expresamente), que adquiera una licencia por cada equipo desde el cual ejecute el software.

Las licencias de Microsoft® estipulan expresamente que el derecho de uso se concede para un único equipo, y prohíben su almacenamiento o uso en red salvo que se adquiera una licencia por cada equipo que vaya a usar el software.

Así, la licencia del programa Microsoft® Outlook 2000® dispone que:

“Puede instalar, utilizar, tener acceso, mostrar, ejecutar o de otra manera interactuar con (“EJECUTAR”) una copia del PRODUCTO SOFTWARE o cualquier versión anterior para el mismo sistema operativo, **en un único equipo**, estación de trabajo, terminal, PC de mano, localizador (*pager*), “teléfono inteligente” u otro dispositivo electrónico digital (“EQUIPO”).

“Podrá también almacenar o instalar una copia del PRODUCTO SOFTWARE en un dispositivo de almacenamiento, tal como un servidor de red, que se use sólo para EJECUTAR el PRODUCTO SOFTWARE en sus otros EQUIPOS dentro de una red interna; sin embargo, usted deberá adquirir y dedicar una licencia para cada EQUIPO distinto EN EL QUE SE ejecute EL PRODUCTO SOFTWARE desde el dispositivo de almacenamiento. No podrá compartir ni usar simultáneamente en diferentes EQUIPOS una licencia para el PRODUCTO SOFTWARE.”

Por su parte, la licencia de Windows 98® dispone que:

“Instalación y uso del software. Sólo podrá instalar y usar una copia del PRODUCTO SOFTWARE en el EQUIPO.”

“Almacenamiento y uso en red. No se puede instalar, tener acceso, mostrar, ejecutar, compartir o utilizar simultáneamente el PRODUCTO SOFTWARE en o desde diferentes equipos, incluidos estaciones de trabajo, terminales y otros dispositivos electrónicos digitales (“dispositivos”). No obstante lo anterior, y excepto en lo que se dispone de otra forma a continuación, cualquier número de dispositivos pueden tener acceso o de otra forma utilizar los servicios de archivo e impresión y los servicios de web personales del PRODUCTO SOFTWARE, si se incluyen.”

También puede ocurrir que la licencia de uso permita expresamente su uso en red, con un número máximo de ordenadores que puedan utilizar el software simultáneamente (por ejemplo, cinco ordenadores por cada licencia, a abonar un precio adicional por cada ordenador que exceda de ese número máximo).

Por otra parte, esta restricción no aparece solamente en licencias de un software estándar comercializado en masa. También en aquellas licencias de uso de un software de aplicaciones más especializado, con destino a empresas y negociado con el licenciatario, es frecuente limitar el número de equipos informáticos en los que la empresa licenciataria puede instalar y ejecutar el software. En tales casos, esta limitación suele consistir en una cláusula de la licencia, junto con un “anexo” a la misma, donde se enumeran y describen esos equipos informáticos.

Así, en la licencia de uso para la aplicación “SAP”®, tendríamos una limitación tal como la siguiente:

“Cláusula X. El software sólo podrá ser instalado en la ubicación descrita en el anexo del presente contrato e instalado en las unidades designadas con el sistema operativo que en el mismo se especifiquen.”

“ANEXO [se adjunta a la licencia]”

**Nota**

Podéis ver la licencia especial para *upgrade* en el apartado 4.5.2.

El software sólo podrá ser instalado única y exclusivamente en los siguientes ordenadores ubicados en las oficinas del USUARIO, quedando limitado el uso a su personal autorizado:

Ordenador “servidor” modelo [...]

4 Ordenadores PC modelo [...]

- c) **Uso en un tipo de ordenador determinado:** en ocasiones, la licencia se concede para un determinado tipo de ordenador (PC, servidor, *mainframe*). Si el usuario decide posteriormente utilizar el software en otro tipo de ordenador (por ejemplo, de mayor potencia), deberá adquirir una nueva licencia. Así lo veíamos al hablar de la licencia especial para *upgrade*.
- d) **Uso privado o comercial (profesional):** también puede ocurrir que una licencia se conceda sólo para un uso profesional (es decir, el uso de un empresario o profesional en su actividad económica) o para un uso privado (doméstico, el propio de los consumidores).



Esta restricción suele darse en aquel software cuyas aplicaciones se utilizan tanto por consumidores como por profesionales. No se trata tanto de que el proveedor pretenda distribuirlo sólo a uno de estos colectivos y excluya al resto, sino que más bien lo distribuirá a todos. No obstante, puede conceder distintos derechos a unos y a otros, así como otorgar las licencias a cambio de un precio diferente, según si el destino del software va a ser doméstico o profesional: normalmente, el precio de la licencia será mucho más alto para el uso profesional.

- e) **Uso limitado a ciertos grupos sociales o sectores de actividad:** similar a la restricción anterior, en algunos casos la licencia exige que el usuario pertenezca (o que no pertenezca) a un determinado colectivo, bien para poder emplear el software, bien para concederle derechos distintos a los que se conceden para otros colectivos. Entre estos “colectivos” podemos destacar las administraciones públicas, las instituciones educativas, las organizaciones sin ánimo de lucro, usos militares, etc.

### 6.2.2. Derecho (prohibición) de copia. La copia de seguridad



Los **proveedores de software propietario**, en tanto que titulares exclusivos del derecho de reproducción (e interesados en lograr la comercialización del mayor número de copias posible de su software), acostumbran a prohibir al usuario que haga copias del software.

Hemos visto que, a diferencia de las obras literarias o artísticas, la Ley de la Propiedad Intelectual no permite al usuario-licenciatario de un software hacer una copia privada del mismo, incluso cuando pretenda destinarla a su solo y particular uso, sin intención de distribuirla. Así pues, si el proveedor prohíbe al usuario hacer copias del software, la Ley de la Propiedad Intelectual tampoco le concede tal derecho, salvo que se trate de una copia de seguridad, o la propia copia en el dispositivo hardware de los archivos que sean necesarios para instalar y ejecutar el software.

Por tanto, es el proveedor quien decide cuántas copias del software puede tener el usuario, que normalmente se reduce a una sola: la que le proporciona el propio proveedor. Asimismo, el proveedor suele prohibir las copias sobre la documentación impresa (manuales de uso) que se adjunta con la copia original del software.

#### Nota

Podéis ver la realización de copia de seguridad en el apartado 2.6.3.

#### Ejemplo

En algunos casos, el proveedor atenúa la prohibición de realizar copias, permitiendo al usuario hacer una segunda copia, para su uso exclusivo, en un equipo portátil (ordenador portátil, PDA, etc.).

Así, la licencia del programa Microsoft® Outlook 2000® estipula que:

“El usuario principal del EQUIPO en que se EJECUTE el PRODUCTO SOFTWARE podrá hacer una segunda copia para su uso exclusivo en un equipo portátil”.

**Nota**

La LPI sí permite al usuario realizar copias cuando sean necesarias para: a) el propio uso del software; b) fines de seguridad o respaldo.

**Nota**

Podéis ver la realización de copias de seguridad en el apartado 2.6.3.

No obstante, la Ley de la Propiedad Intelectual contempla dos **excepciones** al derecho del proveedor del software a prohibir al usuario confeccionar copias:

- 1) La primera viene implícita en el derecho de uso que se concede al usuario: el usuario puede **almacenar** –en definitiva, copiar el software –aquellos archivos necesarios en el equipo hardware, cuando sea **indispensable para la ejecución y uso del software** (art. 100.1 de la Ley de la Propiedad Intelectual).
- 2) La **copia de seguridad o respaldo**: ante la posibilidad de que el soporte en que se proporciona el software pueda sufrir daños o de que el usuario sufra un borrado accidental del software instalado, la propia Ley de la Propiedad Intelectual permite al usuario realizar una copia de seguridad para que pueda volver a utilizar el software sin dificultades (art. 100.2 LPI). El usuario no podrá distribuir esta copia de seguridad a un tercero; y ni siquiera podrá utilizarla sino en el caso de que la copia original se destruya o quede inutilizada.

Nos remitimos a lo estudiado en relación con la realización de copias de seguridad”. Basta recordar que el usuario sólo puede hacer la copia de seguridad “cuando resulte necesaria para la utilización del programa”.



El usuario podrá realizar una copia de seguridad cuando la copia original se encuentre instalada o preinstalada en el propio equipo hardware, o bien cuando el soporte (por ejemplo, el CD-ROM) de la copia original deba emplearse siempre para cargar y ejecutar el software en el equipo. Sin embargo, cuando el soporte de la copia original no sea necesario para ejecutar el software una vez se ha instalado en el equipo hardware, se entiende que el propio soporte original constituye la “copia de seguridad”.

**Ejemplo**

Vemos reflejado este criterio en licencias como la de Windows 98®, cuya cláusula relativa a la posibilidad



del usuario de hacer copias del software dispone que:

“\* Copia de seguridad o de respaldo. Si el fabricante no incluyó una copia de seguridad o de respaldo del PRODUCTO SOFTWARE con el EQUIPO, puede hacer una única copia de seguridad o de respaldo del PRODUCTO SOFTWARE. Sólo puede utilizar la copia de seguridad o de respaldo con fines de archivo. Utilidad de copia de seguridad o de respaldo. Si el PRODUCTO SOFTWARE incluye una utilidad de copia de seguridad o de respaldo de Microsoft, puede usar la utilidad para hacer esa única copia de seguridad o de respaldo. Tras hacer esa única copia de copia de seguridad, la utilidad de copia de seguridad o de respaldo se deshabilitará definitivamente. Excepto en lo dispuesto expresamente en el presente CLUF, no puede hacer de ninguna otra manera copias del PRODUCTO SOFTWARE, incluidos los materiales impresos que acompañan al SOFTWARE.”

### 6.2.3. Derecho (prohibición) de modificación



Los proveedores de software propietario, en tanto que son también titulares exclusivos del derecho de transformación, acostumbran a prohibir al usuario la realización de modificaciones al software, sobre todo en las licencias de software comercializado “en masa”.

Por tanto, al usuario-licenciatario de software propietario le suele estar vedado traducir, adaptar o arreglar dicho software.

### Derecho (prohibición) de modificación y código fuente

Para hacer modificaciones al software, el usuario no sólo necesitaría contar con la autorización del proveedor del software, sino que éste,

además, le debe proporcionar un medio técnico imprescindible: una copia del “código fuente”.



El proveedor del software propietario, como medio para asegurarse de que el usuario cumple con la prohibición de no modificarlo, se limita a entregar la copia del software en versión “código objeto”; es decir, la que permite ejecutarlo. Pero no proporciona al usuario la versión en “código fuente”, que permitiría modificarlo, y que trata de preservar en secreto.

En tanto que el proveedor puede prohibir al usuario que realice modificaciones en el software, no tiene la obligación legal de proporcionarle la versión del programa en código fuente. Es más, el proveedor tiene el derecho a no revelar el código fuente a los usuarios, encontrándose protegido por las normas que amparan los “secretos industriales”.

Incluso, la Ley de la Propiedad Intelectual permite un supuesto muy limitado de derecho del usuario a realizar operaciones de ingeniería inversa con el software licenciado (consistentes en operaciones de descompilación para obtener interoperabilidad con otros programas), fuera del cual dichas operaciones se han de considerar ilegales por vulnerar el derecho exclusivo a la transformación.



Por tanto, y como ya habíamos apuntado, la prohibición de modificar el software que las licencias propietarias imponen al usuario, junto con la negativa del proveedor a proporcionarle el código fuente, son esenciales para los intereses del proveedor de software propietario.

Con ello, el proveedor impide que otros puedan plagiar el programa y crear uno derivado con prestaciones idénticas o mejoradas, para comercializarlo en el mercado aprovechándose del trabajo desarrollado por él; o que empeoren el software y perjudiquen su

reputación. Por otra parte, el proveedor también se reserva así la exclusividad para prestar servicios de actualización o mantenimiento sobre su software.

### Limitaciones a la prohibición de modificar el software

La Ley de la Propiedad Intelectual (art. 100) contiene unas excepciones muy limitadas, en las cuales el proveedor del software no puede prohibir al usuario hacer modificaciones sobre el mismo.

Sin embargo, además de tratarse de supuestos excepcionales, en la práctica el usuario tiene muy difícil beneficiarse de tales excepciones (y el consiguiente derecho a modificar el software), porque el proveedor del software no tiene la obligación de entregarle el código fuente del programa:

- a) **Uso legítimo y corrección de errores:** el artículo 100.1 de la Ley de la Propiedad Intelectual establece que el usuario legítimo no necesita autorización del proveedor para la transformación del programa, incluida la corrección de errores, cuando sea necesario para la utilización del mismo, con arreglo a su finalidad propuesta. Sin embargo, esta norma contempla asimismo que las partes puedan pactar lo contrario en la licencia ("salvo disposición contractual en contrario").

Por tanto, lo que podía ser una norma interesante e importante para el usuario, en virtud de la cual podría modificar el programa para mejorar su uso (en especial, en caso de errores de funcionamiento del software) y, quizás, encontrarse legitimado para exigir al proveedor la versión en código fuente del programa, queda en la práctica vacía de contenido: el proveedor suele imponer en las licencias de software propietario la prohibición expresa de cualquier modificación, ni siquiera para corregir errores.

El propio proveedor es quien se encargará de solventar el error: a su coste o al del usuario, según si el software se halla o no en garantía, o bien sujeto a un servicio de mantenimiento.

- b) **Descompilación para la interoperabilidad:** de una forma bastante condicionada y limitada, el artículo 100.5 de la Ley de la

**Nota**

Podéis ver las tareas de ingeniería inversa o descompilación en el apartado 2.6.2.

Propiedad Intelectual (así como las leyes sobre propiedad intelectual equivalentes en el resto de estados miembros de la Unión Europea) permite al usuario efectuar tareas propias de la llamada “ingeniería inversa”, que en el ámbito de la programación se conocen como *descompilación*.

Según analizamos, con dichas operaciones y a partir de los archivos ejecutables en código binario, el usuario puede conseguir una versión aproximada del código fuente, la cual permite modificar y adaptar el software.



Como excepción al derecho en exclusiva de modificación del proveedor del software, el usuario no necesita la autorización de éste para la “reproducción del código y la traducción de su forma” si ello es indispensable para “obtener la información necesaria para la interoperabilidad de un programa creado de forma independiente con otros programas”.

Es decir, la descompilación está permitida si tiene como objetivo posibilitar que un programa funcione en conjunción con otros, de modo que el usuario aproveche al máximo la utilidad conjunta de todos ellos.

**Nota**

Por lo que respecta a los requisitos y restricciones que el usuario debe respetar para poder beneficiarse del derecho de “descompilar” el programa con fines de interoperabilidad, nos remitimos a lo visto en 2.6.2. En suma, recordemos que el usuario sólo podrá recurrir a la descompilación cuando sea indispensable para conseguir la interoperabilidad y únicamente para tal fin:

- Si el proveedor no le facilita la interoperabilidad de otra manera.
- Debe limitar la descompilación a aquellas partes del programa necesarias para conseguir la interoperabilidad.

- Debe abstenerse de emplear la descompilación para propósitos distintos, en particular el desarrollo de un programa similar o cualquier infracción de los derechos de autor, etc.

### Licencias de software propietario con derecho de modificación (limitado)

Si bien las licencias de software propietario acostumbran a prohibir que los usuarios modifiquen el software, y el proveedor-licenciante asegura tal prohibición manteniendo el código fuente en secreto, podemos destacar dos tipos de licencias “propietarias” que, de algún modo, sí que permiten un acceso limitado del usuario al código fuente y, por ende, prevén un derecho limitado a modificar el programa:

- a) **Licencias de software propietario adaptables a las necesidades del usuario:** ya vimos que hay licencias de software propietario (software complejo, destinado a usos empresariales o profesionales) que, aun siendo el software estándar, permiten ciertas adaptaciones del mismo para adecuarlo a necesidades particulares del usuario.

#### Nota

Recordemos que dichas adaptaciones pueden consistir tanto en **parametrizaciones** (también llamadas **extensiones**), cuando no se modifica el código fuente del programa; como en **customizaciones** (también llamadas simplemente **modificaciones**), cuando sí que se modifica el código fuente.

En estas licencias, puede preverse que dichas adaptaciones las efectúe tanto el proveedor-licenciante (licencias “llave en mano”), como el propio usuario-licenciario.

Cuando el proveedor permite al usuario hacer tales adaptaciones, éste pasa a tener –siquiera de forma limitada– un derecho de transformación sobre el software licenciado. Y, si el usuario puede

#### Nota

Podéis ver la licencia “llave en mano” en el apartado 4.5.1.

hacer “modificaciones”, también se contempla una posibilidad limitada de acceso al código fuente del programa.

De contemplarse estos derechos a favor del usuario, la licencia de uso regulará asimismo a quién pertenecen los derechos de autor sobre esas customizaciones o parametrizaciones: normalmente, se preverá que tales derechos de autor correspondan al usuario, pero el proveedor licenciante se reservará ciertas prerrogativas.

Así, de corresponder al usuario los derechos de autor sobre las adaptaciones que desarrolle del software, se contemplará una “licencia automática” a favor del proveedor, sobre las adaptaciones desarrolladas por el usuario; o bien el compromiso de éste a no distribuir las a terceros sin contar con el beneplácito del proveedor (en forma de autorización a la distribución, o derecho preferente del proveedor a adquirir una licencia sobre las adaptaciones).

#### Ejemplo

Un ejemplo de este tipo de licencias lo encontramos en las licencias de uso de la aplicación informática “SAP”®, cuya cláusula relativa a modificaciones y extensiones dispone que:

“El CLIENTE podrá hacer modificaciones y extensiones al software [...] siempre que sea para su uso, en la/s unidad/es designada/s y de conformidad con lo establecido en la presente cláusula.”

“En el caso de que el CLIENTE lleve a cabo una modificación o extensión en el software sin la participación de SAP, el primero ostentará sobre dicha modificación o extensión todos los derechos de propiedad intelectual, sin perjuicio de los derechos que sobre el software le corresponden a SAP.”

“Asimismo, el CLIENTE se compromete expresamente a negociar con SAP, en primer lugar y en los más estrictos términos de buena fe, la transferencia de dichas modificaciones o extensiones a SAP. Igualmente, el CLIENTE

acuerda expresamente que hasta que no tenga lugar la renuncia expresa de SAP a tal derecho a negociar la transferencia referida, las modificaciones o extensiones del cliente únicamente podrán ser usadas en conexión con las actividades propias del negocio del CLIENTE, no pudiendo ser distribuidas, licenciadas, sublicenciadas, vendidas, transferidas ni de cualquier otro modo cedidas a terceros.”

- b) Por otra parte, están las **licencias de uso “semilibres” o “seudolibres”**: se hallan en un estadio legal intermedio entre las licencias “propietarias” y las licencias “libres”. Corresponden a iniciativas procedentes de grandes compañías desarrolladoras y distribuidoras de software propietario, tales como por ejemplo “Microsoft” o “Sun”.

En la unidad siguiente analizaremos con detalle cada una de estas iniciativas y licencias “seudolibres”. En este apartado, basta señalar que estas licencias permiten a los usuarios tener un acceso limitado al código fuente y efectuar así modificaciones en el software.

Sin embargo, en ningún caso consisten en licencias de software libre, por cuanto:

- Suelen establecer discriminaciones según los usos a que se destine el software: por ejemplo, se permite la modificación para usos no comerciales y se prohíbe para usos comerciales.
- Limitan o excluyen la posibilidad de distribuir esas modificaciones: por ejemplo, se contempla una licencia exclusiva y automática a favor del proveedor-licenciante inicial sobre las modificaciones desarrolladas por el usuario.

#### Nota

En la unidad 7 veremos las principales iniciativas que han surgido y que han dado lugar a licencias de software “seudolibres”, tales como:

- Microsoft Shared Source Initiative

- Sun Community Source
- Apple 1.x
- Aladdin Free Public License

#### 6.2.4. Derecho (prohibición) de distribución

Los proveedores de software propietario prohíben a los usuarios distribuir a terceros su derecho de uso sobre el software y, por ende, la copia del software licenciado. Entre los derechos de explotación exclusivos, aquéllos tienen el exclusivo derecho a realizar o autorizar **“cualquier forma de distribución pública, incluido el alquiler del programa de ordenador original o de sus copias.”**



Como hemos visto, la Ley de la Propiedad Intelectual presume incluso que, si no se dice nada en contra, las licencias de uso sobre el software se conceden al usuario con carácter “intransferible” y para “satisfacer únicamente sus necesidades”. Por lo tanto, aunque no se diga nada expresamente en la licencia, el usuario, en principio, no puede transmitir su copia del software a terceros; sólo en aquellos casos en los que el proveedor lo autorice de forma explícita.

En todo caso, ni el usuario podrá transmitir a terceros la copia del software que le proporcione el proveedor, ni hacer nuevas copias y distribuirlas a terceros. Tal como analizamos, los proveedores de software propietario prohíben al usuario hacer copias del software, ni siquiera para uso privado.

También analizamos en la unidad 4 que el titular de un software propietario está interesado en retener en exclusiva el derecho de distribución sobre el software. Cuando se pretende obtener un beneficio económico de la comercialización del software propietario, el modelo de negocio se funda en la distribución de copias, por lo que cuantas más copias se comercialicen, mayores beneficios obtendrá el proveedor.





Bajo este derecho exclusivo a la distribución del software, los proveedores de software suelen prohibir al usuario ceder temporalmente o transmitir a terceros la copia del software que han adquirido por medio de la licencia. Y prohíben la transmisión de la copia por cualquier título: el usuario no puede ni vender, ni regalar, ni alquilar ni prestar la copia del software a otra persona.

Además, hay que recordar que el derecho exclusivo de distribución “se agota” con la “primera venta en la Unión Europea de una copia por el titular de los derechos o con su consentimiento”.

Ante esta posibilidad de agotarse el derecho de distribución, el proveedor del software bien se cuida de dejar claro en el texto de las licencias de uso que no “vende” ninguna copia de software al usuario. Si así fuera, ello le haría perder su derecho exclusivo de controlar la distribución de las copias, y entonces el usuario podría distribuir su copia libremente (si bien, no podría hacer copias sucesivas para distribuir las), al menos en la Unión Europea. Por tanto, el proveedor sólo “vende” el soporte del software (el CD-ROM), pues respecto al software sólo concede al usuario el derecho a usar el software, **prohibiéndole que transmita su derecho de uso**.

No obstante lo anterior, existen supuestos de software propietario en los que sí se permite su copia y redistribución del mismo, como es el caso del *freeware*, el *shareware* o el software de evaluación: esto se debe a que el proveedor no pretende tanto obtener un beneficio económico directamente de la copia que proporciona al usuario, como promocionar la versión completa del software u obtener licencias definitivas de pago. No obstante, este software continúa siendo propietario (no libre) porque, entre otras razones, no se permite su modificación ni se facilita el código fuente del mismo.

Por otra parte, en los supuestos de software estrechamente vinculado con el funcionamiento de un dispositivo hardware (sistema operativo, *drivers*, software preinstalado, etc.), los proveedores de software pueden llegar a permitir una redistribución limitada del software, vinculada necesariamente a una transmisión del hardware.

#### Nota

Podéis ver las unidades 2 y 4.

**Ejemplo**

De conformidad con lo comentado, Microsoft® permite al usuario, en sus licencias para Windows® o programas de entorno Office, una única “transferencia” del software. El usuario podrá transmitir todos sus derechos sobre el software de manera permanente y como parte de la venta o transferencia definitiva del equipo hardware. El usuario no podrá quedarse con ninguna copia del software, y el tercero (comprador del equipo) deberá aceptar los términos y condiciones de la licencia.

**6.3. Derechos y libertades en las licencias de software libre**

Como ya hemos dicho, las licencias de uso constituyen también el instrumento legal habitual para distribuir el software libre. Sin embargo, mediante las licencias de software libre el proveedor del software no pretende preservar al máximo sus derechos exclusivos de explotación que le reconoce la legislación sobre derechos de autor (en España, la Ley de la Propiedad Intelectual).



Al contrario, mediante la licencia, el proveedor de software libre permite expresamente a los usuarios **usar**, **modificar**, así como **redistribuir** el software, con o sin modificaciones. Las licencias de software libre no suponen que el proveedor renuncie a su condición de “autor” o titular del software, pero sí que significan una puesta a disposición, generalizada a favor de la comunidad de usuarios, de sus derechos de explotación sobre el software.



Recordemos que, según hemos visto, en España y en el resto de países de derecho continental, los derechos morales no se transmiten ni puede renunciarse a ellos,

por lo que el software sigue teniendo un “titular”. Con la licencia de software libre, este titular cede a la comunidad de usuarios (mejor podríamos decir, “comparte con la comunidad de usuarios”) los derechos de explotación. Por esta misma razón, tampoco es posible poner voluntariamente un software bajo el dominio público.

No obstante, incluso en los países de derecho anglosajón (en los que no existe la figura de los “derechos morales” de autor respecto del software, y sí que se puede poner un software voluntariamente en el dominio público), que es de donde proceden las licencias de software libre, también se ha querido dejar bien claro que el autor originario del software libre no renuncia a su condición de tal.

La propia legislación sobre derechos de autor, como la Ley de la Propiedad Intelectual, concede en exclusiva al proveedor del software el derecho a hacer o **autorizar a terceros** la reproducción (instalar y ejecutar), copia, modificación y distribución del software. Por tanto, y como hemos visto, para que un usuario pueda beneficiarse de las libertades del software libre, necesita el permiso explícito (la licencia) del proveedor: de lo contrario, se entendería que utiliza el software de forma ilegítima.

Por otra parte, las **condiciones y restricciones** que se imponen a los usuarios en las licencias de software libre son mucho menores que las recogidas en las licencias de software propietario. Además, estas “limitaciones” a las libertades no pretenden proteger derechos exclusivos del proveedor, sino que consisten fundamentalmente en condiciones para el ejercicio de dichas libertades, con el objeto de:

- a) **Proteger la reputación** del autor del software.
- b) **Asegurar las libertades** de uso, modificación y distribución a los usuarios en todo momento; en particular, evitar e impedir cualquier intento de “apropiación” del software libre.

**Nota**

Podéis ver el apartado 5.1.3

En este apartado, podemos recordar la definición de la Free Software Foundation sobre “licencias de software libre”. Según la FSF, las licencias de software libre son aquellas que permiten y aseguran a los usuarios el ejercicio de las cuatro libertades siguientes:

- Ejecutar el programa, con cualquier propósito (libertad 0).
- Estudiar cómo funciona el programa y adaptarlo a las propias necesidades (libertad 1).
- Distribuir copias (libertad 2).
- Mejorar el programa y publicar las mejoras a los demás (libertad 3).



Para el ejercicio de estas libertades, en especial las libertades 1 y 3, el usuario necesita disponer del código fuente del programa. Las licencias de software libre contienen, efectivamente, el compromiso del proveedor-licenciante a proporcionar el código fuente a los usuarios; o, al menos, ponerlo a su disposición.

A continuación, y de forma paralela al apartado 6.2, vamos a analizar cómo se otorgan a los usuarios los derechos de uso, copia, modificación y distribución en las licencias de software libre. Las escasas restricciones y prohibiciones que estipulan estas licencias, con relación al ejercicio de estos derechos, hacen que podamos calificarlas como auténticas libertades para el usuario.

### 6.3.1. Libertades de uso y de copia

El proveedor de software libre permite a los usuarios todos aquellos actos que, como titular exclusivo del derecho de reproducción sobre el software, en principio sólo le corresponderían a él.



El usuario tiene libertad completa para utilizar y copiar el software cómo, cuándo, cuánto y donde estime oportuno.

## Libertad de uso

El usuario puede emplear el software libre sin restricciones: instalarlo en su equipo hardware, almacenar los archivos necesarios y ejecutarlo cada vez que desee para beneficiarse de sus aplicaciones.

Asimismo, el usuario puede estudiar libremente el software libre: no sólo durante su carga y ejecución (como ocurre en el software propietario), sino que también puede disponer del código fuente para su estudio.

Lo que caracteriza a la libertad de uso en las licencias de software libre, a diferencia de las licencias de uso de software propietario, es que las de software libre no deben incluir restricciones al derecho de uso. De este modo, el usuario puede utilizar el software:

- a) Para cualquier propósito o finalidad. Por tanto, no cabe limitar el empleo del software al “uso personal” del usuario. Asimismo, el software libre puede emplearse tanto para finalidades privadas o profesionales, sin que quepa estipular discriminaciones por razón del colectivo o grupo al que pertenezca el usuario.
- b) En los equipos hardware que estime oportuno, sean las que sean sus características técnicas.
- c) Pertenezca al colectivo que pertenezca.

### Ejemplo

Destacamos la libertad de uso que establece la **GNU-GPL**, estableciendo en su cláusula 1, segundo párrafo, simplemente (y de forma contundente) que “el acto de ejecutar el programa no está restringido”.

Asimismo, dos directrices de la OSD prohíben expresamente que las licencias impongan ciertas restricciones en los usuarios:

- La directriz 5 impone la “no discriminación con respecto a personas o grupos”.

- La directriz 6 impone la “no discriminación con respecto a campos laborales (sectores de actividad)”.

Estas directrices se explicarán con detalle en la unidad 7. En este momento, basta decir que, según la OSD, las licencias que se adecuen a su definición no deben contener restricciones de uso: en particular, no pueden excluir el uso del software a colectivos determinados o a finalidades determinadas.

### Libertad de copia

El usuario de software libre puede efectivamente hacer cuantas copias del software desee, sin tener que limitarse únicamente a aquellos archivos necesarios para la ejecución del software en el equipo hardware o a una sola copia de seguridad.

Esta libertad de copia está estrechamente vinculada con las libertades de uso (el usuario puede utilizar el software en cuantos equipos de hardware desee) y de distribución (el usuario puede proporcionar copias del software, con o sin modificaciones, a terceros).

#### 6.3.2. Libertad de modificación



Con las licencias de software libre, los usuarios pasan a tener también el derecho de transformación sobre el software licenciado. Por tanto, los usuarios pueden **traducir** el software, **adaptarlo** a sus necesidades, **corregir errores** o **combinarlo** con otros programas.

Así, el usuario que desarrolle un software derivado a partir del software libre licenciado pasará a considerarse como el autor de dicho software derivado, al que la propia Ley de la Propiedad Intelectual (art. 96.3) concede derechos de autor (morales y de explotación) exclusivos.

Es en el ámbito de la libertad de modificación donde empieza a surtir efecto la existencia de cláusulas *copyleft*.



Recordemos que las licencias libres *copyleft* (en particular, la GNU-GPL) prohíben al usuario añadir restricciones a la licencia de un software libre derivado (desarrollado a partir del software libre original) para redistribuirlo a terceros, aparte de las que contenía la licencia del software originario. De este modo, el software derivado también debe redistribuirse como software libre.

En cuanto a la libertad de modificación que las licencias de software libre conceden al usuario, la incidencia de una cláusula *copyleft* es la siguiente:

- a) Si la licencia de software libre es *copyleft*, el usuario que desarrolle un software derivado sólo se quedará en exclusiva con los derechos morales sobre el mismo. Sin embargo, si desea distribuir el software derivado, la licencia del software libre originario le exige ceder los derechos de explotación (uso, modificación y distribución) sobre las modificaciones con el mismo grado de libertad –como mínimo– contemplado en dicha licencia del software libre originario.
- b) Si la licencia de software libre no es *copyleft* (como las licencias de tipo BSD, Mozilla, etc.), el usuario puede decidir si también redistribuirá el software derivado como software libre o, en cambio, puede optar por distribuirlo como software semilibre (con más restricciones) e incluso como software propietario en algunos casos.

## Disponibilidad del código fuente



Para permitir que el usuario pueda ejercitar efectivamente esta libertad de modificación, el proveedor-licenciante no sólo debe proporcionar al usuario el soft

### Nota

Las cláusulas del *copyleft* se han visto en la unidad 2 y se analizan en detalle en la unidad 7.

### Nota

En la unidad 7 veremos estos diferentes supuestos pormenorizados.

ware en código objeto o ejecutable: también debe proporcionarle el código fuente o, por lo menos, ponerlo a su disposición a precio de coste (coste de reproducción de la copia).

Tal y como lo definen la GNU-GPL y las directrices OSD, el código fuente es la forma preferida para que un programador haga modificaciones en el software.

Las licencias de software libre, así como las directrices OSD (también conocidas por el título de su primera versión, directrices Debian de código abierto), incluyen el compromiso del proveedor a proporcionar al usuario el código fuente o, como mínimo, a que éste pueda conseguirlo fácilmente si lo quiere, a precio de coste. Los medios y formas de proporcionar el código fuente se analizan con detalle en la unidad 7.

#### Nota

En este apartado basta destacar que el proveedor tendrá principalmente las dos opciones siguientes:

- 1) Acompañar al software (en código objeto o ejecutable) con una copia del código fuente completo, en formato electrónico.
- 2) Comprometerse por escrito a proporcionar copia del código fuente, a petición del licenciataria, al precio de coste de la copia. Dicho compromiso debe durar durante un tiempo (por ejemplo, un mínimo de 3 años en la GNU-GPL, 12 meses en la licencia Mozilla).

### Límites a la libertad de modificación en el software libre

Hemos visto que la libertad de modificación en las licencias de software libre tiene un alcance muy amplio. No obstante, en el ejercicio de esta libertad, los usuarios deben observar:

- a) Una **condición**, impuesta (si bien de distinta manera) por las distintas licencias de software libre y las directrices OSD: el usuario



que modifique el software debe respetar el anuncio de *copyright* del autor original y advertir qué archivos ha modificado.

Recordemos, a modo de repaso y de ilustrar esta condición, que la GNU-GPL señala en su cláusula 3 que el usuario “debe hacer que los ficheros modificados lleven anuncios prominentes indicando que han sido modificados y la fecha de estas modificaciones”.

Las directrices OSD van incluso más allá, y aunque permiten que el programa sea modificado por el usuario y éste pueda distribuir las modificaciones:

- La directriz 4 autoriza al autor original a impedir que los usuarios distribuyan versiones modificadas del código fuente. Los usuarios no podrán distribuir directamente el código fuente modificado; sin embargo, sí que podrán distribuirlo si acompañan los archivos fuentes originales con “archivos parche” separados.
- Asimismo, el autor original puede exigir que los trabajos derivados tengan un nombre distinto o un número de versión distinto del software original.

Por lo que respecta a las licencias tipo BSD, una de las pocas restricciones que contempla es, precisamente, mantener el “anuncio de *copyright*” al redistribuir el software, tanto el código fuente como en binario. Además, prohíbe emplear el nombre del autor “para respaldar o promover productos derivados de este programa sin un permiso previo por escrito.”

Esta condición tiene por finalidad proteger la reputación del autor original ante posibles problemas en el funcionamiento del software a raíz de una modificación.

- b) Un límite impuesto por la Ley de la Propiedad Intelectual, cual es el derecho moral a la integridad de la obra. En España (y en los países de derecho continental, que reconocen este derecho moral), los usuarios deben abstenerse de realizar aquellas modificaciones que puedan menoscabar la reputación del proveedor.

**Nota**

Podéis ver el apartado 5.1.3.



No son sólo las licencias, sino el derecho moral del autor a la integridad de su obra (reconocido por la Ley de la Propiedad Intelectual), lo que impide a un usuario hacer modificaciones en el software que puedan empeorarlo o que disminuyan sus aplicaciones, por no hablar de la introducción de un virus informático.

### 6.3.3. Libertad de distribución



Los usuarios de software libre tienen una libertad más, impensable en las licencias de software propietario (salvo supuestos excepcionales, como el del *freeware*): la libertad de distribuir copias del software a terceros, con o sin modificaciones.

Esta libertad es amplísima, por cuanto el usuario va a poder distribuirlo:

- a) Gratuitamente o a cambio de una remuneración económica; de forma temporal (alquiler, préstamo) o indefinida.



Como dijimos en el apartado relativo al “precio” en las licencias de software libre, el proveedor de software libre tiene derecho a pedir una contraprestación económica. Sin embargo, lo más habitual es que distribuya las copias gratis, o a cambio de un precio mínimo o residual (para resarcirse de ciertos gastos, como el de realizar la copia, entregarla en un soporte físico, etc.): no tiene sentido cobrar un precio alto por la copia de un software libre, cuando los usuarios podrán, a su vez, distribuir sucesivamente las copias que quieran.

No obstante, ello no impide que existan modelos de negocio alternativos a la comercialización de copias de software; basados en software libre, en los que el ma-

#### Nota

Podéis ver el apartado 5.3.2, relativo al “precio” en las licencias de software libre.

yor precio de la solución informática no radica en la licencia, sino en los servicios de consultoría y/o mantenimiento adscritos a ese software libre (por ejemplo, los productos y soluciones Red Hat, entre muchos otros).

- b) Proporcionando una copia del software en código objeto y/o con el código fuente; redistribuyendo el software originariamente licenciado o con modificaciones.

### Límites a la libertad de distribución en el software libre

Sin perjuicio de la libertad tan amplia que los usuarios tienen para distribuir el software libre licenciado, así como la libertad de distribuir el software derivado que desarrollen, deben asimismo respetar ciertas condiciones y límites:

- a) Como ya hemos dicho, las distintas licencias de software libre (GNU-GPL, las de tipo BSD, etc.) y las directrices Debian imponen al usuario que redistribuya el software (con o sin modificaciones), debe respetar el anuncio de *copyright* del autor original y, en su caso, advertir qué archivos ha modificado.
- b) Asimismo, el usuario que redistribuya el software debe conservar las advertencias sobre garantías y responsabilidades que contenía la licencia originaria.
- c) Como veremos en el apartado siguiente, las licencias de software libre, más que garantías, incorporan un “repudio de garantía” (cláusula de exoneración o *disclaimer*): una cláusula por la que el proveedor del software manifiesta proporcionar el software sin ofrecer garantía alguna, y rechaza la posibilidad de ser responsable frente a los daños que pudiera causar al usuario un error o fallo de funcionamiento del software.
- d) Dejando para el apartado siguiente la discusión sobre la validez legal de estas cláusulas, las licencias de software libre imponen al usuario que, si redistribuye el software, conserve en su licencia dicho “repudio de garantía”.

- e) Las licencias de software libre también suelen contener ciertas limitaciones a la redistribución del software cuando dicha redistribución puede entrar en conflicto con una patente: por ejemplo, abstenerse de redistribuir el software, excluir la libre distribución en aquellos países donde esté en vigor la patente que entre en conflicto con la licencia (GNU-GPL); o bien, advertir que el software libre se encuentra bajo una demanda de violación de patentes por un tercero, identificando a ese tercero (licencia Mozilla).

### Referencia al *copyleft*

En este apartado destacaremos que el *copyleft* afecta sobre todo al derecho del usuario de software libre a distribuir programas derivados que él haya desarrollado a partir del software original.

Si la licencia de software libre es *copyleft* (como la GNU-GPL), el usuario sólo puede redistribuir el software original, o incluso el software derivado que haya desarrollado, de forma libre también. Concretamente, si el usuario quiere distribuir el software original, o el software derivado que haya podido crear a partir del original, debe hacerlo bajo la misma licencia de uso por la que adquirió sus derechos sobre el software original.

Por lo tanto, el usuario que pasa a ser proveedor-licenciante de un software derivado debe permitir a los nuevos usuarios el libre uso, modificación y distribución del software; con el mismo grado de libertad –como mínimo– contemplado en la licencia del software libre originario.

Si no lo hace, se entiende que este usuario está infringiendo la licencia original, por lo que puede perder los derechos de uso sobre el software original objeto de la misma.



Por ello, y para asegurar que dichas libertades persisten al redistribuirse el software, la GNU-GPL permite al usuario distribuir el software original o el derivado, siempre que lo acompañe con una copia del código fuente en formato electrónico, o bien se comprometa –

#### Nota

En la unidad 7 analizamos en detalle la figura del *copyleft*, a la cual ya nos hemos referido en la unidad 2.

durante el plazo de 3 años mínimo— a ponerlo a disposición del nuevo usuario, a precio de coste de realización de la copia.

Para el caso en el cual el usuario redistribuya el software con usos no comerciales y el licenciente del software original no le hubiera proporcionado el código fuente, sino que simplemente se hubiera comprometido a ponerlo a su disposición, basta con que el usuario (ahora nuevo licenciente) proporcione al nuevo usuario la información que al respecto recibió del licenciente del software original (esto es, cómo disponer del código fuente).

Por otra parte, si la licencia de software libre no es *copyleft* (como las licencias de tipo BSD, Mozilla, etc.), no obliga al usuario que desarrolla un software derivado a distribuir este software derivado como software libre. El usuario deberá respetar el “grado de libertad” que le otorgue la licencia para distribuir el software derivado, la cual le puede permitir redistribuirlo como semilibre o incluso a través de licencias de software propietario.

Por tanto, es posible que el autor del software derivado distribuya éste como un software propietario, sin proporcionar el código fuente y, en suma, reservarse en exclusiva los derechos de modificación y distribución sobre el software derivado.

Como resumen, en la tabla siguiente comparamos el contenido esencial de una licencia propietaria emblemática: la licencia de usuario final (más conocida por su acrónimo inglés EULA, o CLUF en versión española) de Microsoft® relativa a Windows®, con la licencia libre GNU-GPL.

**Tabla 1.**

Tabla resumen sobre derechos y restricciones en las licencias de software propietario y las licencias de software libre		
	CLUF	GNU-GPL
Derechos otorgados	<ul style="list-style-type: none"> <li>• Puede usarse en un único ordenador con un máximo de 2 procesadores.</li> <li>• La licencia sólo puede transferirse una vez a otro usuario. El licenciatario debe destruir su copia.</li> </ul>	<ul style="list-style-type: none"> <li>• Permite el libre uso, copia, modificación y redistribución del software.</li> <li>• Pueden concederse sucesivas licencias y se puede cobrar por los servicios sobre el software.</li> </ul>

Tabla resumen sobre derechos y restricciones en las licencias de software propietario y las licencias de software libre		
	CLUF	GNU-GPL
Restricciones obligaciones	<ul style="list-style-type: none"> <li>Se prohíbe la copia, la modificación y la redistribución</li> <li>No puede ser usado como servidor web o de archivos (<i>fileserver</i>).</li> <li>Impone una limitación sobre la ingeniería inversa (la única modificación posible).</li> <li>Registro necesario a los 30 días.</li> </ul>	<ul style="list-style-type: none"> <li>Las modificaciones y obras derivadas deben distribuirse bajo la misma licencia (<i>copyleft</i>).</li> <li>La distribución tiene que incluir el código fuente.</li> </ul>
Derechos reservados	<ul style="list-style-type: none"> <li>Puede dejar de funcionar si se efectúan cambios en el hardware.</li> <li>Las actualizaciones del sistema pueden modificar la licencia, si el licenciente lo desea.</li> <li>Se reserva el derecho para, en cualquier momento, recoger la información del sistema y su uso, y entregar dicha información a terceros.</li> <li>La garantía es por los primeros 90 días, y limitada al precio del software.</li> <li>Las actualizaciones y los parches no tienen garantía.</li> </ul>	<ul style="list-style-type: none"> <li>En principio, la GPL no ofrece ninguna garantía, aunque permite al licenciente que decida ofrecerla, o se vea obligado a ello por la ley que le sea aplicable.</li> </ul>

## 6.4. Garantías y responsabilidades

Una cuestión muy importante en las relaciones proveedor-licenciente y usuario-licenciatario es **determinar las consecuencias legales** que se derivan ante una incidencia en el funcionamiento del software. Ello, máxime si tenemos en cuenta que el software es relativamente inestable (es susceptible de sufrir fallos de funcionamiento o defectos de configuración, etc.), así como las importantes inconveniencias y perjuicios que puede sufrir el usuario como consecuencia de una incidencia en el software (en especial, aquellas empresas y entidades cuya actividad depende del buen funcionamiento de sus sistemas informáticos).

### 6.4.1. Consideraciones generales

Las licencias de software acostumbran a regular los derechos del usuario –y consiguientes obligaciones del proveedor– en caso de que se produzca alguna incidencia en el software: fallos de funcionamiento, defectos varios, o que no se corresponda con las características que expone el proveedor y que motivaron la adquisición de la licencia por el usuario.

Cuando se da alguna de estas circunstancias, el usuario se ve impedido de utilizar el software, o de usarlo para las finalidades que lo llevaron a adquirir la licencia.



Si la licencia está vigente y el usuario no tiene culpa alguna de la incidencia, un principio de justicia nos diría que el proveedor-licenciante debe asistir al usuario y poner fin a la incidencia: hablamos entonces de que el proveedor debe prestar al usuario una garantía sobre la continuidad en el buen funcionamiento del software.

Es más, tal incidencia puede haber causado al usuario daños y perjuicios. Pensemos sobre todo en aquel software destinado a empresas o profesionales, del cual depende en la práctica la buena marcha de su actividad cotidiana: un defecto o fallo de funcionamiento puede paralizar su actividad, lo que sin duda le causaría daños y perjuicios.



Otra cuestión consistirá en determinar si el proveedor puede ser “culpable” y, por tanto, responsable de los daños y perjuicios que sufre el usuario por causa de una incidencia en el funcionamiento del software, de modo que deba indemnizarle por ello.

Hemos visto que los proveedores de software, en particular cuando el software es de consumo y comercializado en masa, no negocian el clausulado de las licencias con los usuarios, sino que imponen el contenido de las cláusulas.



A los proveedores-licenciantes les resulta de gran interés establecer en la licencia de uso limitaciones o, incluso, exclusiones de garantías y responsabilidades frente al usuario. Sin embargo, en muchos casos, una cláusula de exoneración o limitación no será legal.

El mismo principio de justicia al que nos referíamos nos dice que sería injusto y/o abusivo que la licencia permitiera al proveedor (en especial, cuando cobre un precio) desentenderse de las incidencias que

se produzcan en el software. En particular, será injusto cuando el usuario no haya tenido ninguna oportunidad de negociar el contenido de estas cláusulas, sino que le hayan venido impuestas por el proveedor-licenciante.



Cuestión distinta serán aquellas licencias en las que el usuario-licenciario sí que haya tenido la oportunidad de negociar el contenido de la licencia de uso, y una exoneración o limitación de garantías y/o responsabilidad a favor del proveedor-licenciante tenga causa en otra contrapartida a favor del usuario (por ejemplo, una rebaja en el precio; una mayor garantía a cambio de menos responsabilidad, etc.), lo que podrá suceder en aquellas licencias de uso de software especializado, de precio elevado y adaptado a las necesidades del usuario. En tal supuesto, dicha limitación o exoneración sí que podría entenderse como justa, en tanto que pactada libremente por dos partes en igual o similar posición negociadora.

Las leyes sobre propiedad intelectual (en España, la Ley de la Propiedad Intelectual) no se ocupan de regular las garantías y responsabilidades que han de contemplar las licencias, sino únicamente de regular los derechos exclusivos del autor sobre el software.

Sin embargo, en todos los países se aplican normas de derecho general de las obligaciones y contratos, normas sobre garantías en otros contratos (aplicables por analogía a la licencia de software, como apuntábamos en la unidad 4), normas protectoras de los consumidores, etc.; todo ello, para obligar al proveedor a que asuma ciertas garantías y responsabilidades frente al usuario, sin que pueda eludirlas mediante la licencia.

A continuación, veremos qué significan las garantías y las responsabilidades respecto al software licenciado, para analizar después las limitaciones y exclusiones de garantías y/o responsabilidad que acostumbra a incluirse en las licencias de uso, destacando la discusión sobre su validez legal. Acabaremos este apartado con una aproximación a las limitaciones y exclusiones en las licencias de software libre, y el consiguiente debate sobre su validez; para ello, destacaremos las cláusulas que estipula la GNU-GPL en esta materia.



### 6.4.2. Distinción entre garantías y responsabilidades

Ambas figuras –garantía y responsabilidad– se solapan parcialmente y resultan algo difíciles de distinguir. Mejor dicho, podemos afirmar que las “responsabilidades” son una de las consecuencias a las que puede dar lugar la infracción de una garantía.

#### Garantías



Denominamos **garantías** a los compromisos u obligaciones que el proveedor-licenciante asume a favor del usuario respecto a las condiciones (características, prestaciones, buen funcionamiento) que debe cumplir el software objeto de la licencia. De este modo, si el software no cumple o deja de cumplir en algún momento dichas condiciones, el proveedor-licenciante está obligado a emprender las actuaciones oportunas para que el software se ajuste a las mismas.

En particular, cabe destacar la **garantía de buen funcionamiento**, en virtud de la cual el proveedor debe asegurar al usuario-licenciario que el software funcionará correctamente durante el plazo de vigencia de la licencia –o, al menos, durante un tiempo determinado– de modo que, en caso de que el software presente algún fallo, prestará la asistencia oportuna al usuario para remediarlo.



A todo software se le atribuyen unas características y prestaciones determinadas en función del tipo de software de que se trate; así como de aquellas características especiales que el proveedor haya incluido en la licencia (o bien en la documentación accesoria, envoltorios, folleto, publicidad, etc.). Además, se entiende que el software estará siempre listo para funcionar correctamente, cuando el usuario decida ejecutarlo correctamente. De lo contrario, estaremos ante una incidencia debida a un fallo de funcionamiento o bien el software presentará un defecto.

Las licencias de uso regulan qué garantías debe prestar el proveedor, el modo de prestarlas y el plazo (durante cuánto tiempo, a partir del inicio de la licencia). Es decir, en el caso de que ocurriera alguna de las incidencias descritas, el proveedor-licenciante asistirá o no al usuario para poner fin a la incidencia y, en su caso, como le asistirá. El modo de poner fin a la incidencia será mediante:

- Reparación del fallo o defecto
- Sustitución de la copia por otra
- Resolución o cancelación de la licencia: el proveedor devuelve el precio al usuario y devolviendo éste la copia del software al proveedor. En principio, el usuario sólo podrá acudir a la resolución de la licencia como último recurso, cuando no sea posible la reparación del defecto o sustitución de la copia.

En cualquier caso, las cláusulas de las licencias que estipulen las garantías –así como sus posibles limitaciones o exoneraciones– deberán respetar una serie de normas legales imperativas que, en cada país, obligan a prestar unas garantías mínimas sobre el software.

Creemos conveniente explicar de manera sencilla las clases y origen legal de las garantías. En derecho continental, como en España, las clases y categorías jurídicas de las garantías son distintas respecto de las propias del derecho anglosajón. Sin embargo, muchas licencias de software, aun escritas en castellano y para regir en España, hacen referencia a las garantías típicas del derecho anglosajón. Esto hace que la redacción de dichas cláusulas nos parezca poco clara y confusa, incluso para los propios juristas.

El contenido y alcance de las garantías es similar en ambos casos, así como las acciones (*remedies*, su acepción inglesa) que se estipulan a favor del usuario para hacerlas efectivas: reparación, sustitución de la copia o devolución del precio con cancelación de la licencia.

Conforme al derecho español, las garantías sobre el software serían las siguientes:

- a) La **garantía de saneamiento frente a defectos ocultos**, por aplicación analógica de las normas que regulan la compraventa (Có-

digo civil y Código de Comercio): el licenciatario podrá reclamar –en el plazo de los 6 meses siguientes al inicio de la licencia– que se le devuelva el dinero y se cancele la licencia, o bien que el proveedor le rebaje el precio abonado. Esta garantía será aplicable, por analogía, a muchas de las licencias, en particular las del software comercializado en masa a consumidores.

Los Tribunales vienen aplicando a las licencias de uso, por analogía, la garantía de saneamiento frente a defectos ocultos. Aun en el caso que una licencia de uso, por sus circunstancias específicas sea asimilable, no a una compraventa, sino a un arrendamiento de cosa, de obra o prestación de servicio (por ejemplo, por concederse para un plazo determinado; por obligarse el proveedor a adaptar el software a necesidades particulares del usuario, a realizar tareas de instalación, etc.), se entienden que rigen garantías similares:

- Deber del proveedor de mantener al usuario en el uso normal y pacífico del objeto contractual (garantía propia en el arrendamiento de cosas).
- Deber de mantener el resultado final en las condiciones pactadas (garantía propia en el arriendo de obra).
- Deber de prestar el servicio con la diligencia propia de un profesional en la materia (garantía propia de la prestación de servicios).

El incumplimiento de estas garantías da lugar a que, en este caso el usuario, pueda pedir igualmente la reparación del defecto, sustitución de la copia o resolución de la licencia (devolviéndose las partes, respectivamente, la copia del software y el precio abonado).

- b) La **garantía de buen funcionamiento del software**, por aplicación de la normativa sobre garantías en la Ley General de Defensa de los Consumidores y Usuarios (art. 11).

En el plazo mínimo de 6 meses, el licenciatario podrá reclamar frente al proveedor, en caso de error, defecto o falta de las condiciones óptimas para cumplir el uso a que esté destinado el software. El proveedor estará obligado a reparar el error (debiendo de tener un servicio técnico adecuado para ello) y, si la reparación no es posible o satisfactoria, deberá sustituir la copia del software por otra o devolver el precio pagado al licenciatario.

Este régimen de garantía se aplicaría en principio sólo a aquellas licencias en las que el licenciatarlo sea un consumidor. Sin embargo, en la práctica esta garantía podrá extenderse a cualquier licencia, sobre la base del **principio de buena fe** en los contratos. En cualquier caso, si el licenciatarlo es consumidor, este régimen de garantía es de aplicación imperativa, sin que el proveedor pueda restringirlo por medio de la licencia.

En términos similares a lo señalado anteriormente, cuando la licencia se asemeje más a un arrendamiento o a un contrato de obra o servicio, la garantía de buen funcionamiento será inherente a la licencia de uso, por aplicación analógica de los deberes del proveedor de mantener al usuario en el uso normal y pacífico del objeto contractual, o de mantener el resultado final en las condiciones pactadas.



Existe la reciente (en las fechas que se escriben estas líneas) Ley 22/2003, de Garantías en la Venta de Bienes de Consumo, en trasposición de una directiva de la Unión Europea. Establece una garantía de dos años a favor del consumidor, durante la cual el vendedor o el fabricante responderán de la falta de conformidad del producto con las finalidades propias del mismo, o con las especialmente pactadas. Sin embargo, y pendientes todavía del reglamento que la ha de desarrollar, el software quedaría fuera de su ámbito de aplicación, por lo que el régimen legal aplicable en materia de garantías seguirá siendo el descrito anteriormente.

En derecho anglosajón, también hablamos de garantías que son originarias de las normas que regulan el contrato de compraventa (*implied warranties*), y que se aplican analógicamente en las licencias de software. Su denominación es distinta y, sin embargo, aparecen citadas en multitud de licencias al ser éstas traducciones o adaptaciones de una licencia original en inglés:

- a) **Garantía mercantil o de mercantibilidad** (*merchantability*): el software debe poder comercializarse legalmente (no es algo prohibido) y debe ser adecuado para cumplir los fines normales propios de sus características.

**Ejemplo**

Por ejemplo, incumpliría esta garantía un software consistente en una hoja de cálculo que no permitiera sumar o restar.

- b) **Garantía de adecuación a un fin particular** (*fitness for a particular purpose*): el software debe ser adecuado para cumplir un fin particular cuando el licenciatarlo adquirió la licencia motivado por la posibilidad de cumplir tal fin y el proveedor sabía, o podía saber, que el licenciatarlo quería adquirir la licencia precisamente por ello.

**Ejemplo**

Esto que parece tan complicado podemos explicarlo siguiendo el ejemplo de antes: se incumple esta garantía si, pongamos por caso, el licenciatarlo quiere una hoja de cálculo que sirva para realizar ciertas operaciones aritméticas muy complicadas, se lo indica expresamente al proveedor, pero resulta que al final la hoja de cálculo no es capaz de realizarlas.

- c) Junto a estas garantías, en derecho anglosajón también se habla de la **garantía de título y no infracción** (*title and non-infringement*): corresponde a la garantía de “titularidad”, a la cual ya nos hemos referido, cuyo contenido damos aquí por reproducido.

**Nota**

Podéis ver la garantía de “titularidad” en la unidad 5.

Explicaremos aquí también el significado de una expresión habitual en las licencias: la mención de que el software se entrega “tal cual” (*as is*) significa que el proveedor lo entrega sin prestar ninguna garantía. Si bien decimos que se trata de una mención habitual en las licencias de uso, en muchas ocasiones no será válida conforme a derecho. Por lo pronto, en España difícilmente será legal, en tanto que infrinja las garantías imperativas que hemos enumerado antes: garantía de saneamiento frente a vicios ocultos y garantía de buen funcionamiento.

## Responsabilidad



La **responsabilidad** consiste en el deber del proveedor de indemnizar al usuario por aquellos daños y perjuicios que éste haya sufrido como consecuencia de un

error en el software, defecto, fallo de funcionamiento o falta de adecuación con las características que del mismo cabe esperar.

Aquí nos ceñimos a los supuestos en que, al concurrir alguna incidencia en el software (lo que sin duda supondrá el quebrantamiento de una garantía), el usuario sufre por ello unos daños y perjuicios. Por tanto, al usuario no le bastará entonces con la prestación de la garantía (que el proveedor le repare el software, le proporcione una copia nueva o le devuelva el precio para que el usuario vuelva a contar con un software en perfecto funcionamiento), sino que además pretenderá obtener del proveedor una indemnización por los daños, en la medida en que éste sea responsable de los mismos.

Para considerar al proveedor como responsable de los daños, el fallo o defecto que los haya causado no debe tratarse de algo fortuito o producido exclusivamente por culpa del propio usuario, sino achacable de alguna forma al proveedor-licenciante:

- a) Bien por lo que jurídicamente se llama **dolo**: cuando el proveedor conocía la existencia del fallo o defecto en el software que produjo los daños al usuario.
- b) Bien por **culpa o negligencia**: no conocía la existencia del fallo o defecto en el software que produjo los daños, pero debería haberlo conocido, si hubiera cumplido su labor de programación o mantenimiento posterior del software con el grado de diligencia exigible. Si hubiera sido diligente, se supone que habría podido prever y evitar los daños.

#### Ejemplo

Pensemos en una empresa que sufre una paralización de su actividad por un fallo de funcionamiento en una aplicación informática. En este caso, si la empresa sufre perjuicios (pérdidas de archivos, negocios no realizados, salarios abonados a empleados que no pueden trabajar, etc.), puede plantearse exigir una indemnización al proveedor.

Aquí también creemos conveniente hacer una referencia a las distintas clases y categorías jurídicas en materia de daños y perjuicios entre los países de derecho continental y los de derecho anglosajón. Al igual que en las garantías, las licencias de software suelen hacer referencia a las clases de daños que se contemplan en el derecho anglosajón. También su contenido y alcance es similar. Conforme al derecho español, cabe hablar de:

- a) **Daño emergente:** valor de las pérdidas patrimoniales y morales diversas que el usuario puede sufrir, a consecuencia de la incidencia, así como los gastos en los que haya incurrido para remediarla: por ejemplo, si un fallo en el software da lugar a una pérdida de información, el valor de esta pérdida de información; daño a la imagen que el usuario empresario sufre frente a clientes.
- b) **Lucro cesante:** ganancia dejada de obtener, por culpa de la incidencia: por ejemplo, los ingresos que el usuario empresario deja de percibir durante el tiempo en que su actividad queda paralizada por culpa del fallo en el software.

En derecho anglosajón, por su parte, se habla de otro tipo de daños, que no equivalen exactamente a las categorías anteriores:

- a) **Daños directos o incidentales** (*direct or incidental damages*): los que son consecuencia directa de la incidencia (por ejemplo, de nuevo, la pérdida de información, gastos de reconstrucción de la información perdida, etc.).
- b) **Daños indirectos o consecuenciales** (*indirect or consequential damages*): daños que derivan de la incidencia, aunque son una consecuencia indirecta de la misma, si bien las partes conocían o debían conocer que podrían sufrirse en caso de ocurrir tal incidencia (por ejemplo, pérdida de imagen ante clientes).

Hay ciertos daños, como la pérdida de beneficios (*loss of profit*, en buena medida coincidente con el concepto continental de *lucro cesante*) que se incluirían en principio entre los daños indirectos. Sin embargo, en ocasiones el criterio de los Tribunales británicos o estadounidenses ha variado, para incluirlos dentro de los daños directos. Por ello, las licencias suelen citar la pérdida de beneficios, de forma independiente a los daños directos e indirectos, para que quede bien claro que también se refieren a ellos.

Conviene distinguir entre garantías y responsabilidades, tanto por su verdadera diferencia conceptual, como por su distinto régimen legal. Así pues, la validez de ciertas limitaciones de garantías debe valorarse de forma distinta a la validez o no de las limitaciones o exoneraciones de garantías. En especial, veremos a continuación que, en las licencias de software libre, la validez de la cláusula de “ausencia de garantía” debe valorarse de forma distinta a la cláusula de “ausencia de responsabilidad”.

#### **6.4.3. Limitaciones y exclusiones de garantías y responsabilidades en las licencias de software**

Hemos dicho antes, y el lector lo habrá leído en cualquier licencia, que los proveedores suelen incluir cláusulas de limitación o exoneración de garantías y responsabilidades en las licencias de software (*disclaimers*). Sin embargo, la validez legal de dichas cláusulas es cuestionable.

Con estas cláusulas, el proveedor pretende ahorrarse riesgos y costes. Veamos a continuación, la limitación y exoneración de garantías, así como la limitación y exoneración de responsabilidad.

#### **Limitación y exoneración de garantías**

Los proveedores-licenciantes de software pretenden sustraerse de ciertas garantías a prestar al usuario, a acortar el tiempo en el que deberá prestarlas, etc., para lo cual imponen limitaciones o exoneraciones de garantías en el texto de las licencias de uso. Sin embargo, al haber abonado el usuario un precio por la licencia, el proveedor está obligado a prestar ciertas garantías, conforme a diferentes normas legales que son de aplicación:

- 1) Si el licenciataria es un consumidor, debe observar el régimen de garantía imperativo que establece la Ley General de Defensa de Consumidores y Usuarios (art. 11). Durante 6 meses como mínimo, el usuario podrá reclamar en caso de error, defecto o falta de las condiciones óptimas para cumplir el uso a que el software esté destinado. El proveedor estará obligado a reparar el error (debiendo disponer de un servicio técnico adecuado para ello) y, si la reparación no es posible o satisfactoria, deberá sustituir la copia del software por otra o devolver el precio pagado al licenciataria.



- 2) Si el licenciatario no es un consumidor, sino empresario o profesional, el proveedor podrá restringir en la licencia su deber de garantía. Sin embargo, tampoco podrá desentenderse así como así de las incidencias que puedan ocurrirle al software, y el licenciatario podrá recurrir a la aplicación analógica de las normas (Código civil y Código de comercio) que estipulan para el contrato de compraventa la **garantía de saneamiento frente a vicios** (defectos) **ocultos**, el principio de la **buena fe** en los contratos, u otras garantías similares, para exigir al proveedor que el software siga en perfecto funcionamiento. Pero será caso por caso en el que se deberá valorar la validez o no de las restricciones en las garantías.

### Limitación y exoneración de responsabilidad

Asimismo, los proveedores-licenciantes de software pretenden sustraerse de responsabilidad en las licencias, imponiendo cláusulas que exoneran o limitan el deber de indemnizar al usuario cuando éste sufra daños a causa de una incidencia en el software.



Son típicas en las licencias de software las **cláusulas que exoneran de responsabilidad** al proveedor frente a daños y perjuicios que sufra el usuario debido a una incidencia en el software. O, en caso de tener que indemnizar al usuario por daños y perjuicios, **limitan** la posible indemnización del proveedor a una cantidad equivalente al precio que el usuario ha abonado por la licencia (junto, en su caso, a las cuotas que haya abonado por la prestación de un servicio de mantenimiento).

La validez de esta cláusula es más que dudosa. Por lo pronto, la exoneración o limitación de responsabilidad no es válida cuando:

- a) Sea **responsabilidad por dolo**: por *dolo* entendemos no sólo el caso en el que el proveedor haya causado a sabiendas los daños (lo que no será muy normal), sino también el supuesto en el que el proveedor conocía la existencia de la incidencia, que podía producir determinados daños al usuario, y no impidió que se produjeran.



En otros países distintos a España, tanto de derecho anglosajón como de derecho continental (por ejemplo, Alemania), además del dolo, tampoco es limitable la responsabilidad por negligencia grave: cuando la incidencia sea achacable al proveedor por una falta de la diligencia ya no normal, sino básica, que tendría cualquier proveedor.

#### Nota

Existen normas protectoras de los consumidores que prohíben o restringen las cláusulas de exoneración o limitación de responsabilidad en las licencias de software frente a un usuario consumidor.

#### Nota

En el apartado 5.1.2. podéis ver cuándo un usuario empresario o profesional puede instar la nulidad de una cláusula de la licencia que sea abusiva.

- b) Se trata de responsabilidad por daños que consistan en **muerte o daños corporales** a las personas: en principio, nos cuesta pensar en un software cuyos fallos o errores produzcan este tipo de daños, salvo en casos muy puntuales (como, por ejemplo, el software de un aparato médico, el empleado por controladores aéreos, etc.).
- c) Cuando el licenciatarlo que sufra los daños es un **consumidor**: en virtud de lo que dispone la Ley General de Defensa de Consumidores y Usuarios (artículos 11.3, 25 y 26), el usuario tiene derecho a ser indemnizado por los daños y perjuicios que demuestre haber sufrido por los fallos o falta de adecuación del software, salvo que se hayan producido por su propia y exclusiva culpa.

Por tanto, si el licenciatarlo es un consumidor, el proveedor no puede válidamente reducir el límite de responsabilidad a una cantidad máxima, pues tal cláusula será automáticamente nula por infringir la ley y por ser cláusula abusiva. El único límite cuantitativo válido lo establece la propia Ley General de Defensa de Consumidores y Usuarios, en 500.000.000 de las antiguas pesetas.

Sin embargo, cuando el usuario-licenciatarlo es un **empresario o profesional**, la limitación de responsabilidad a una cantidad máxima es en principio válida, porque la legislación permite a las partes pactar sobre la misma.



Ello es importante, porque los proveedores están interesados especialmente en limitar sus responsabilidades frente a licenciatarlos empresarios o profesionales, porque es a éstos a los que el mal funcionamiento de un software puede causarles perjuicios mucho mayores –al menos, económicamente– que a un consumidor.

No obstante lo anterior, habrá que estudiar cada caso concreto para determinar si puede resultar especialmente injusta y abusiva. En tal supuesto, la limitación podrá ser **declarada nula** si se considera que, aunque el licenciatarlo sea un profesional, la limitación de responsabilidad es tan desproporcionadamente abusiva que:

- en la práctica, supone dejar en manos del proveedor el cumplimiento de sus obligaciones a su libre albedrío, y
- vulnera el principio de buena fe en los contratos.

En **derecho anglosajón**, la jurisprudencia (decisiones de los tribunales) tiene un criterio más desarrollado, pero que en la práctica equivale a lo apuntado anteriormente: el proveedor de software no puede limitar su responsabilidad ante el licenciatarlo-consumidor, al ser de aplicación las leyes protectoras del consumidor. Sin embargo, cuando el licenciatarlo sea un empresario o profesional, podrá limitar su responsabilidad en la licencia, salvo cuando ello sea “irrazonable”. Para determinar si la exoneración o limitación de responsabilidad es razonable o no, los tribunales atienden a varias circunstancias, lo que se denomina el *test de razonabilidad* (*test of reasonableness*):

- Si ha habido un verdadero proceso de negociación de las cláusulas contractuales, en particular las relativas a garantías y responsabilidades; o si, por el contrario, el proveedor impuso al licenciatarlo el contenido de dichas cláusulas.
- Si el licenciatarlo conoció perfectamente la existencia y alcance de la cláusula de limitación, si estuvo o no estuvo asesorado por un abogado, que le informó de la misma antes de firmar la licencia.
- Si la cláusula de limitación o exoneración fue aceptada por el licenciatarlo a cambio de algo a su favor (por ejemplo, una rebaja del precio).

De este test se deducirá que la cláusula de limitación o exoneración es válida, por ser razonable si el usuario tuvo la oportunidad de negociarla con el proveedor, conocerla, aceptarla como parte del proceso de negociación antes de la firma de la licencia, etc. Por el contrario, será nula por “irrazonable”, cuando en la práctica haya venido impuesta por el proveedor a cambio de nada.

Asimismo, tampoco podrá ser válida una limitación de responsabilidad a una cantidad máxima que resulte irrisoria (por ejemplo, la cantidad mayor entre el precio de la licencia o 5 dólares, como se ha estipulado en alguna licencia de Microsoft®), en la medida en que conduzca en la práctica a una exoneración de responsabilidad no consentida especialmente por el licenciatario.

#### 6.4.4. Especial referencia a las exclusiones de garantías y responsabilidades en las licencias de software libre



Se dice, y en buena medida es cierto, que las licencias de software libre se conceden sin garantía alguna para el usuario y sin asumirse ningún tipo de responsabilidad.

No obstante, las cláusulas que así lo prevén son de una legalidad bastante discutible, según hemos analizado, en especial las que persiguen la exoneración de responsabilidad. Será la legislación aplicable la que, en definitiva, determinará si el proveedor del software libre debe prestar o no alguna garantía o incurre en algún tipo de responsabilidad frente al usuario-licenciatario.

Nos centraremos en el análisis de las cláusulas que contiene la GNU-GPL al respecto. En cualquier caso, su contenido es sustancialmente idéntico al de otras cláusulas que contemplan las licencias BSD o Mozilla, por citar unos ejemplos emblemáticos.

#### Exclusión de garantías y garantías voluntarias



La cláusula 11 de la GNU-GPL tiene como título “Ausencia de garantía” (“no warranty”) y establece que:

“Como el programa se licencia sin cargo, no se ofrece ninguna garantía sobre el programa, ello con el límite máximo permitido por la legislación aplicable. Excepto

cuando se pacte de otra forma por escrito, los titulares del *copyright* y/u otras partes proporcionan el programa “tal cual”, sin garantía de ninguna clase, expresa o implícita, incluidas pero no limitadas a las garantías implícitas de mercantibilidad o adecuación para un fin particular. Usted asume cualquier riesgo referente a la calidad y prestaciones del programa. Si el programa se prueba como defectuoso, usted asume el coste de cualquier servicio, de reparación o corrección.”

De este modo, el software libre que se licencia a través de la GNU-GPL, en principio se distribuye sin que el proveedor ofrezca garantía alguna. Sin embargo, es posible que el proveedor agregue una cláusula de garantía, a cambio de una contraprestación económica o no.



Esto es importante, porque el miedo que se tiene sobre la ausencia de garantías en el software libre ha de ser relativo: recordemos que el modelo de negocio, de haberlo, se basa en la prestación de servicios añadidos sobre el software, tales como actualización, mantenimiento o corrección de errores. Es muy probable que, sobre todo en aquel software libre que sea complejo, con un destino comercial o profesional, la garantía venga dada por la prestación de dichos servicios, a cambio de una cuota.

Por lo demás, la validez legal de esta ausencia de garantía podría cuestionarse, debiendo valorarse a la luz de lo que ya hemos comentado. No obstante, la validez de esta ausencia de garantías puede sostenerse cuando el software libre se haya distribuido de forma gratuita.

En efecto, en tal caso podrá decirse que la distribución del software es equiparable a una donación. Y las normas que rigen la donación (el Código civil, en España) no obligan al donante (en este caso sería el proveedor) a garantizar el objeto donado (en este caso, el derecho de uso sobre el software) frente a defectos ocultos o asegurar su buen funcionamiento. Si la cosa donada resulta defectuosa, el donante no está obligado, en principio, a repararla o sustituirla por otra. Distinto

#### Nota

Podéis ver la validez legal de la ausencia de garantía en los apartados 6.4.2 y 6.4.3.

será la responsabilidad de indemnizar por daños y perjuicios sufridos a partir de un fallo de funcionamiento o defecto en el software licenciado.

Así pues, tenemos un fundamento legal para defender la validez de esta ausencia de garantías que estipula la GNU-GPL, ausencia que podrá sostenerse frente a un usuario-licenciatarario profesional. Sin embargo, conforme al derecho español también, debemos entender que el proveedor deberá prestar la garantía a que se refiere el artículo 11 de la Ley General de Defensa de Consumidores y Usuarios (ya analizada), cuando el proveedor sea un empresario-profesional, y el licenciatario un consumidor: garantizar el buen funcionamiento del software en los seis primeros meses de vigencia de la licencia de uso.



El artículo 1.2. de la Ley General de Defensa de los Consumidores y Usuarios señala que:

“a los efectos de esta ley, son consumidores o usuarios las personas físicas o jurídicas que adquieren, utilizan o disfrutan como destinatarios finales, bienes muebles o inmuebles, productos, servicios, actividades o funciones, cualquiera que sea la naturaleza pública o privada, individual o colectiva, de quienes los producen, facilitan, suministran o expiden.” Por su parte, los que deben cumplir las normas de esta ley son el “fabricante, importador, vendedor o suministrador de los productos y servicios”.

De este modo, la Ley General de Defensa de Consumidores y Usuarios está pensando más bien en los productos y servicios que los consumidores y usuarios adquieren a cambio de un precio; pero no excluye de su protección aquellos casos en los que adquieren los productos y servicios de forma gratuita a un “fabricante y suministrador”. Es por ello por lo que debemos entender que el proveedor empresario o profesional que distribuya software libre a licenciatarios consumidores debe observar la garantía establecida en el artículo 11 de la LGDCU.

## Exclusión de responsabilidad



Por su parte, la cláusula 12 de la GNU-GPL establece una exoneración de responsabilidad:

“En ningún caso, salvo que lo exija la legislación aplicable o haya sido acordado por escrito, ningún titular del *copyright* ni ninguna otra parte que modifique y/o redistribuya el programa según se permite en esta licencia será responsable ante usted por daños, incluyendo cualquier daño general, especial, incidental o consecuencial producido por el uso o la imposibilidad de uso del programa (incluidas, pero no limitadas a la pérdida de datos o a la generación incorrecta de datos o a pérdidas sufridas por usted o por terceras partes, o a un fallo del programa al funcionar en combinación con cualquier otro programa), incluso si dicho titular u otra parte ha sido advertido de la posibilidad de dichos daños.”



En cuanto a esta exoneración de responsabilidad, no parece que exista circunstancia alguna –por el hecho de tratarse de software libre– que permita al proveedor librarse del régimen legal aplicable (en España y, de forma similar, en muchos otros países), que prohíbe exoneraciones absolutas de responsabilidad.

Así, como hemos visto, no podrá exonerarse de responsabilidad, ni siquiera limitarla, cuando la responsabilidad provenga de dolo o el licenciatario sea un consumidor. En tales casos, una cláusula de exoneración o limitación de responsabilidad no será válida.

Por otra parte, si el licenciatario es un empresario o profesional, esta exoneración de responsabilidad también presenta dudas sobre su validez. No se trata de una limitación de la cantidad máxima de una posible indemnización, sino de una auténtica negación de hacerse responsable de los daños en cualquier caso. Pues bien, en la medida en que los da-

### Nota

Podéis ver el apartado 6.4.3.

ños se produzcan por unos fallos en el software que no sean imprevisibles e inevitables para el proveedor, o bien por culpa exclusiva del usuario, los tribunales muy posiblemente lo considerarían responsable, por negligencia.

En este caso, el hecho de que se pueda distribuir gratuitamente el software libre, no tiene relevancia. Aunque le fueran de aplicación las normas de la donación, éstas no eximen al donante de indemnizar al donatario por los daños que le cause la cosa donada (siempre que exista dolo o negligencia por su parte).

## 6.5. Otras cláusulas

A continuación, haremos una referencia a ciertas cuestiones que también se regulan en las licencias de software. No podemos decir que todas las licencias de software las contemplen expresamente pero, de hacerlo, tienen una importancia destacada en la relación contractual entre proveedor y usuario.

### 6.5.1. Jurisdicción competente y legislación aplicable

En muchas licencias se estipula expresamente una cláusula sobre jurisdicción competente y legislación aplicable, la cual tendrá importancia en caso de divergencias entre las partes que den lugar a un litigio. Son de particular relevancia, claro está, cuando proveedor y licenciatario tienen residencia en países diferentes. En virtud de este pacto, en la licencia se determina:

- a) La **jurisdicción competente**: los tribunales de qué país (región, ciudad, etc.) serán los competentes para resolver los litigios que surjan entre las partes como consecuencia de la licencia. Asimismo, las partes pueden acordar que los litigios no se resuelvan por juzgados y tribunales ordinarios, sino por medio de arbitraje privado.

Por tanto, si una parte quiere reclamar algo a la otra, deberá demandarla ante los tribunales o árbitros que se hayan pactado como los competentes.



- b) El **derecho aplicable**: de qué país será el derecho (leyes, reglamentos, etc.) al que se deba recurrir para aplicar e interpretar las cláusulas de las licencias. En caso de litigio, el tribunal o árbitro designado como competente deberá resolverlo conforme al derecho que las partes hayan pactado como aplicable.

En caso de que las partes no hayan acordado expresamente cuál será la jurisdicción competente y/o el derecho aplicable a la licencia, habrá de estarse a lo que determinen las leyes sobre derecho internacional privado de cada país.

Entendemos que no es objeto de esta unidad ni del curso hacer un estudio pormenorizado de estas cláusulas. No son propias de las licencias de software, sino de cualquier contrato, de cierta importancia, en que las partes no residen en el mismo estado.



A los efectos de este apartado, basta con que conozcáis que, en principio, son válidos los pactos de las partes, por los que designen en la licencia la jurisdicción competente y el derecho aplicable al contrato.

La **excepción** se tiene, de nuevo, cuando el usuario-licenciatario sea un consumidor. En tal caso:

- a) El usuario podrá demandar al licenciante tanto en los tribunales correspondientes al domicilio del licenciante, como en los del domicilio del usuario (lo que sin duda será más cómodo y barato para él). Sin embargo, si el licenciante pretende demandar al usuario, sólo podrá hacerlo ante los tribunales del domicilio del demandado. Por lo que respecta a la posibilidad de someter los litigios a arbitraje, sólo serán válidos los “arbitrajes de consumo”, que siguen un procedimiento más sencillo y económico.
- b) Aunque se haya pactado un derecho aplicable distinto al del país de residencia del usuario, éste podrá reclamar igualmente la aplicación de aquellas leyes protectoras de los consumidores en su país de residencia.

**Ejemplo**

Pensemos en una licencia de software en la que el proveedor es estadounidense y el usuario es un consumidor español. Si, por ejemplo, en la licencia se estipula que los tribunales competentes son los americanos, y el derecho aplicable será el federal de los EE.UU. y el del estado de California:

- El usuario podrá igualmente demandar al proveedor en España, y los tribunales españoles se considerarán competentes.
- El usuario podrá alegar la aplicación de la Ley General de Defensa de Consumidores y Usuarios, así como otras normas protectoras de los consumidores en España.

Sin embargo, para que el licenciatario consumidor pueda beneficiarse de estas normas protectoras de sus intereses, se requiere que haya contratado la licencia con el proveedor y éste haya efectuado algún tipo de actividad comercial específicamente dirigida al país de residencia del usuario (publicidad, apertura de una tienda, etc.).

Esto es importante, si se tienen en cuenta la multitud de licencias que se contratan por Internet, en particular, desde las páginas web de los proveedores de software. En principio, en caso de litigio por una licencia de un software adquirido por Internet, el usuario consumidor podrá beneficiarse o no de las normas protectoras mencionadas, según si la página web se dirige de forma específica a su país de residencia (además o no de a otros países).

**Ejemplo**

Siguiendo con el último ejemplo, pongamos que la licencia corresponde a un software descargado en Internet. Habrá que comprobar si la página web del proveedor estadounidense de alguna manera se dirigía específicamente a España (por ejemplo, con signos expresivos tales como tener un apartado en idioma español, consignar el precio en euros, señalar un servicio

técnico o sucursal que el proveedor tenga en España, etc.). En tal caso será cuando el licenciatarlo consumidor pueda demandar en España al proveedor en el supuesto de que surja un litigio entre ambos, y requerir la aplicación de la normativa española protectora del consumidor.

En las licencias de software libre, vemos cómo algunas (la GNU-GPL, por ejemplo), no contienen cláusula alguna sobre jurisdicción competente o derecho aplicable. En cambio, otras licencias como la Mozilla sí que regulan este aspecto, al someter la licencia (cláusula 11, de la versión 1.1):

- Al derecho del estado americano de Connecticut.
- En caso de que una de las partes de la licencia sea ciudadano o entidad estadounidense, a la jurisdicción de los tribunales federales del estado de Connecticut.

La validez y eficacia de esta cláusula deberá valorarse conforme a lo que hemos explicado en este apartado, en especial cuando el licenciatarlo sea un consumidor.

### 6.5.2. Pactos de confidencialidad

Los pactos o cláusulas de confidencialidad son propios de las licencias de software, no tanto comercializado en masa, sino del software especializado con aplicaciones profesionales.

Así, en el software “de consumo”, el licenciatarlo no tiene acceso a ninguna información o datos confidenciales del proveedor: sólo recibe una copia en formato ejecutable del software.

Sin embargo, en el caso de software más complejo y especializado, con aplicaciones profesionales, que incluso se adapta a las necesidades específicas del licenciatarlo, es posible que el licenciatarlo acceda a información o datos técnicos del licenciante que éste mantiene como

#### Nota

Respecto al acceso limitado al código, podéis ver el apartado 6.2.3.

confidenciales, en tanto que secretos industriales o empresariales: acceso limitado al código fuente, documentos de diseño, especificaciones de las interfaces, diagramas de flujo, algoritmos, hojas de decodificación, ideas, etc.



Pues bien, si el usuario-licenciatario tiene acceso a información o datos técnicos sobre el software que el proveedor-licenciante mantiene como confidenciales, para revelarlos sólo de manera limitada a sus licenciarios, en la licencia se establecerá un **pacto de confidencialidad**.

En virtud de tal pacto, el licenciatario devendrá obligado a:

- a) No revelar a terceros los datos confidenciales.
- b) No utilizar los datos confidenciales para otros fines más que los expresamente permitidos por la licencia.
- c) Limitar el uso y acceso a los datos técnicos (cuando el licenciatario es una empresa o entidad) sólo a aquellos empleados o colaboradores que deban utilizar el software licenciado; y adoptar las medidas oportunas para asegurarse de que tales empleados o colaboradores respetan también la obligación de confidencialidad (por ejemplo, haciéndoles firmar un compromiso de confidencialidad equivalente).

Si el licenciatario vulnera su obligación de guardar confidencialidad, el proveedor licenciante podrá cancelar la licencia y, en su caso, reclamarle daños y perjuicios.

En el caso de licencias de software libre, también podrá contemplarse un pacto de confidencialidad, en la medida en que el licenciatario tenga acceso a datos e información que el licenciante pretenda conservar como confidenciales. No obstante, como no podrá ser confidencial el código fuente del programa, la mayoría de las licencias de software libre no incluyen pactos de confidencialidad.

### 6.5.3. Cláusulas relativas a patentes

Destacamos también que las licencias de software, y en particular las de software libre, hacen referencia a las patentes. El software libre se encuentra sujeto a sucesivos desarrollos y actualizaciones por la comunidad de usuarios y puede en algún momento entrar en conflicto con una patente; o bien el de un software derivado a partir de un software libre puede intentar patentarlo (sobre todo en aquellos países más receptivos con la concesión de patentes de software, a la vez de más interesantes desde el punto de vista de la explotación del software, tales como los EEUU o Japón). Por tanto, es lógico que las licencias de software libre prevean estas situaciones.

Así, tenemos los ejemplos siguientes:

- 1) La **GNU-GPL**, seguramente porque la Free Software Foundation se muestra radicalmente en contra de las patentes sobre software, no contempla la posibilidad de que el software licenciado, se trate del original o de una obra derivada, sea objeto de una patente. Sí que se ocupa de regular situaciones que puedan surgir cuando el software libre licenciado entra en conflicto con la patente de otro software:
  - Si una patente de software impide al licenciatario cumplir con las obligaciones de la GNU-GPL, ello no le exime de cumplir con la GNU-GPL. Así pues, si una patente no permite al licenciatario la redistribución libre del software (por ejemplo, sin cláusula *copyleft*, sin proporcionar el código fuente), el licenciatario se abstendrá de redistribuir el software.
  - Si una patente impide el uso y/o distribución del software en ciertos países, quien licencie el programa puede añadir una limitación explícita en la licencia, según la cual algunos países quedarán excluidos de la libre distribución del software.
- 2) La **Mozilla** prevé la posibilidad de que un desarrollador cree una obra derivada a partir del software licenciado, y solicite y obtenga una patente (en Estados Unidos) sobre tal obra, que será parte del software. Para impedir que modificaciones del

software libre constituyan violaciones de patentes, la Mozilla obliga a la persona “patentadora” a conceder a los otros licenciarios una licencia de patente respecto de la modificación patentada.

Además, en caso de que el software objeto de la licencia se encuentre bajo una demanda de violación de patentes por un tercero, el licenciante (*contributor*) deberá avisar de ello mediante un archivo con el código fuente que se titulará “LEGAL”, describiendo la reclamación y el tercero que la realiza.

## 6.6. Conclusiones

En esta unidad hemos estudiado el distinto esquema de derechos, restricciones y prohibiciones que se estipulan en las licencias propietarias y libres. Asimismo, hemos visto un punto muy importante que se regula en las licencias de uso, cuáles son sus consecuencias en caso de que se produzca una incidencia en el software (garantías y responsabilidades); y también hemos obtenido una referencia a cuestiones que se regulan con frecuencia en las licencias de uso, tales como cláusulas de jurisdicción competente y legislación aplicable, pactos de confidencialidad o cláusulas relativas a patentes.

De todo ello, podemos extraer las conclusiones siguientes:

- a) Las licencias de software propietario acostumbran a conceder únicamente al usuario un derecho limitado al uso del software, sujeto a múltiples restricciones.
- b) Las licencias de software propietario vedan al máximo al usuario el derecho a realizar copias del software, si bien la legislación permite a éste realizar una copia de seguridad.
- c) Salvo ciertas licencias “llave en mano” y las “semilibres”, las licencias propietarias prohíben al usuario modificar el software, medida que refuerza el licenciante manteniendo en secreto su código fuente. Asimismo, suelen prohibir la redistribución del software por parte del usuario.

- d) Las licencias de software libre conceden y aseguran a los usuarios las libertades de uso, modificación y redistribución del software licenciado, con o sin modificaciones.
- e) De este modo, las licencias libres confieren al usuario un derecho a utilizar el software sin restricciones.
- f) Las licencias libres permiten al usuario modificar el software, para lo cual el licenciente le proporciona –o, al menos, pone a su disposición– el código fuente. No obstante, el usuario debe respetar los anuncios de *copyright* del autor original, advertir qué ficheros ha modificado y, en suma, no debe menoscabar la reputación de dicho autor original.
- g) Las licencias libres también permiten al usuario redistribuir el software, con o sin modificaciones. Si incorporan una cláusula *copyleft* (como la GNU-GPL), el usuario debe distribuir el software derivado con el mismo grado de libertad –como mínimo– contemplado en la licencia del software originario. Si no incorporan esta cláusula, el usuario puede optar por redistribuir el software derivado como libre o como propietario.
- h) Los licenciantes acostumbran a incluir en las licencias exoneraciones o limitaciones de garantías y responsabilidad. En muchos casos, tales exoneraciones o limitaciones no son válidas, en especial si el usuario es un consumidor.
- i) Las licencias de software libre incorporan exoneraciones de garantías y responsabilidades. En principio, es válida la exoneración de garantías, salvo que el licenciente cobre por la licencia (o, claro está, por prestar la garantía) o el usuario sea un consumidor. La exoneración de responsabilidad presenta serias dudas sobre su validez, conforme al derecho español.
- j) Algunas licencias incorporan pactos sobre la jurisdicción competente y el derecho aplicable, los cuales son válidos salvo cuando el usuario sea un consumidor, y se le impida acudir a los tribunales de su país o a la aplicación de las normas protectoras de los consumidores en su país.
- k) Los pactos de confidencialidad son propios de aquellas licencias de uso en los que el usuario accede a información secreta del licenciente, como el código fuente del software; por ello, no son inhabituales en licencias de software libre.

- l) Sí que son habituales, en las licencias de software libres, la existencia de cláusulas que se refieran a las patentes: tanto para regular qué se ha de hacer en el supuesto de que el software entre en conflicto con la patente de otro software, como para regular los derechos de los usuarios en el caso de que alguno de ellos obtenga una patente sobre un software derivado, creado a partir del original licenciado.



## 7. Licencias de software libre

En esta unidad vamos a analizar con detenimiento el elemento que constituye la base jurídica del movimiento de software libre y que lo diferencia del software tradicional o propietario: la licencia de software libre.

Ya hemos visto algunos aspectos relevantes de las licencias de software en general durante el estudio de sus conceptos fundamentales y las cláusulas más importantes. Aquí enfocaremos las modalidades principales de estas licencias libres y sus efectos legales. Esto nos permitirá comentar en la unidad 8 algunos otros temas importantes y correlativos que surgen como consecuencia de su aplicación.

En el primer apartado comentaremos el concepto de *libertad* relativo al software, tal como se plasma en una licencia y sus diferentes interpretaciones en el mundo del software libre. Expondremos dos marcos de estudio y clasificación:

- 1) La definición de *open source software* (software de fuente abierta) de Open Source Initiative (OSI) y los criterios que se han de cumplir para la certificación OSI.
- 2) La definición de *free software* (software libre) de la Free Software Foundation (FSF) y la compatibilidad con la licencia libre por excelencia, la Licencia Pública General GNU (GPL).

Después de esta introducción al tema de la libertad y apertura del software, y antes de pasar al análisis detallado de las principales licencias de software libre, nos vamos a detener brevemente para desmitificar algunas difundidas creencias sobre diversos aspectos legales del software libre y las licencias asociadas.

A continuación, en la tercera parte, analizaremos algunas de las licencias de software libre más frecuentes para entender mejor su fun-

cionamiento y sus características. Presentaremos para cada una de ellas una breve historia y un análisis legal, y destacaremos lo que permiten y lo que prohíben, sus condiciones y sus restricciones. Reseñaremos sus características particulares y su compatibilidad con otras licencias, especialmente la GNU-GPL.

En la cuarta parte, comentaremos otras licencias relacionadas con el software libre, como las de documentación libre y algunas licencias que intentan acercarse a la categoría de 'libre', como las licencias Sun Community y Microsoft Shared Source; y reservaremos una breve nota final a las licencias de tipo *freeware* y *shareware*.

Con el aprendizaje de esta unidad, el lector alcanzará los objetivos siguientes:

1. Conocer las licencias de software libre de mayor uso, enfocando las licencias GPL, LGPL, BSD y MPL; y comprender sus particularidades.
2. Ser capaz de valorar las licencias de software libre a la luz de dos criterios de análisis: lo que significa *abierto* según la definición de la OSI y lo que significa *libre* de acuerdo con la Free Software Foundation.
3. Comprobar la validez legal de varios mitos que se han establecido alrededor del software libre.
4. Entender las diferencias principales entre las licencias libres, propietarias y otras licencias "casi" libres como las licencias Apple Public Source, Sun Community y Microsoft Shared Source.
5. Conocer algunas licencias para otros recursos libres como la documentación de software o las publicaciones en Internet.

### 7.1. Aspectos generales de las licencias libres

En las unidades anteriores hemos presentado las licencias de software en general y su ajuste al marco legal de la propiedad intelectual e industrial. En esta unidad, presentamos las licencias de software libre.



Es importante recordar que una **licencia de software** es un instrumento legal que autoriza a los usuarios del software a realizar ciertos actos que la ley normalmente reserva de manera exclusiva al titular de los derechos de autor o de patente.

Asimismo, la licencia permite al licenciante reservar los derechos que no se ceden e imponer y otorgar al licenciataria otras obligaciones y derechos no necesariamente vinculados con el derecho de autor (confidencialidad, etc.). La licencia establece, por lo tanto, lo que el usuario puede y no puede hacer con el software.

Tal y como hemos visto en las unidades anteriores, la diferencia entre el software libre y el software propietario reside en los derechos y obligaciones especificados en la licencia. Aquellos otorgados por las licencias de software libre ("licencias libres") suelen ser directamente opuestos a los otorgados y reservados por una licencia de software propietaria ("licencia propietaria"), sobre todo en cuanto al uso, la distribución y la modificación del software. En esta unidad analizaremos en detalle estos derechos y obligaciones de las licencias libres.

#### Nota

Mientras que en el apartado 7.3 nos concentraremos en las distintas licencias libres y resaltaremos las diferencias entre ellas y sus efectos, en esta primera parte comentaremos las diferentes interpretaciones de *libertad* y cómo se plasman en una licencia libre.

### 7.1.1. La libertad en el software

Como ya se ha comentado en este curso, la palabra *free* en inglés tiene dos sentidos: 'libre' y 'gratuito'. El uso de *free* en relación con el software no implica que el titular o proveedor del software libre otorgue o distribuya el software de manera gratis (aunque lo puede hacer), sino que lo distribuye bajo una licencia que permite a los usuarios aprovecharlo libremente en cuanto a uso, reproducción, modificación y distribución.

Ya hemos visto la interpretación de la Free Software Foundation de su concepto de libertad:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar el funcionamiento del programa y adaptarlo a sus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias (libertad 2).
- La libertad de mejorar el programa y publicar cualquier mejora, de modo que toda la comunidad se beneficie (libertad 3). El acceso al código fuente es un requisito previo para esto.

Esta definición de libertad es aceptada también por la OSI, como veremos a continuación.

Recalquemos que estas libertades de uso corresponden a los derechos exclusivos de explotación reservados a los titulares de derechos de autor por las leyes de propiedad intelectual que se han estudiado en la unidad 2:

- Libertad 0: el derecho de uso (no es un derecho exclusivo, sin embargo la licencia libre permite un uso sin restricciones).
- Libertad 1: el derecho de modificación.
- Libertad 2: los derechos de copia y distribución.
- Libertad 3: los derechos de modificación y distribución/publicación de las obras derivadas.

Así, cualquier licencia de software libre debe ceder a los usuarios estos derechos.

No obstante, no todas las licencias libres son iguales. Parafraseando (sin demasiado abuso) una célebre expresión de George Orwell, aunque todas sean libres, algunas son más libres que otras. Todas deben garantizar, al menos, las cuatro libertades, pero las disposiciones contenidas

en estas licencias pueden variar de manera significativa. El abanico de posibilidades va desde unas obligaciones mínimas (en las licencias de tipo BSD), que obligan únicamente a mantener el aviso de autoría y la negación de garantías y de responsabilidad (*disclaimer*), hasta el “máximo” (en cierto sentido) de la cláusula *copyleft* de la GPL, que obliga a distribuir cualquier modificación y obra derivada bajo la misma licencia GPL. Además, la gama va más allá de lo realmente libre y hay algunas licencias que intentan ajustarse al modelo de desarrollo libre, como la Sun Community Licence, que no lo son del todo.

Consideramos que las distintas licencias garantizan diferentes tipos de libertad.

- a) La BSD, por ejemplo, otorga más libertad a los desarrolladores, porque éstos pueden incorporar y distribuir implementaciones de “código BSD” bajo licencias tanto libres como propietarias.
- b) La GPL transmite más libertad a los usuarios finales, porque éstos siempre recibirán aplicaciones con código fuente abierta y una licencia libre.



La nomenclatura relativa a las licencias de software libre en sentido amplio, que vamos a utilizar en esta unidad es la misma que hemos presentado en la unidad 1:

- **Software libre y licencia libre:** seguiremos la práctica de la FSF, que usa el término de software libre para cualquier licencia que respete las cuatro libertades antes mencionadas.
- **Software abierto y licencia abierta:** software que cumple las Directrices de la Definición de Software de Código Abierto (OSD).
- **Software *copyleft* y licencia con *copyleft*:** aplicaciones y licencias que se distribuyen con una cláusula de *copyleft* robusta como la GPL.
- **Software y licencia propietaria:** aplicaciones distribuidas bajo licencias que no son libres.

**Nota**

Podéis ver el apartado 1.1.

**Lectura complementaria**

Hay varias referencias a textos de la OSI en el apartado OSI en bibliografía.

A continuación estudiaremos las dos categorías principales de software libre: el software libre propiamente dicho y el software de código fuente abierto.

### 7.1.2. Criterios de clasificación: apertura y libertad

Como ya señalamos, hubo una cierta escisión conceptual en el movimiento de software libre en 1998, que en realidad fue la concretización de una división que venía desarrollándose durante los años 1990. Esta división se formalizó con la creación de la Iniciativa de Código Fuente Abierto (Open Source Initiative u OSI) por Eric Raymond y Bruce Perens entre otros. Como resultado de esta iniciativa, se estableció la **definición de software de código fuente abierto** (Open Source Definition, OSD). La definición OSD nos provee de una primera herramienta de análisis y clasificación de las licencias de software libre, y llamaremos **licencias abiertas** a las que cumplen con sus directrices.

No obstante, ésta no es la única manera de clasificar las licencias. También es importante resaltar las características de las licencias que son libres según la perspectiva de la FSF (con *copyleft*) y explicar sus diferencias con las licencias abiertas.

**Reflexión**

En este curso exponemos en primer lugar los criterios de las licencias abiertas porque establecen directrices más amplias que los criterios de la FSF. Es decir, muchas licencias pueden ser abiertas, pero no todas tienen *copyleft* robusto y, por lo tanto, no pueden considerarse libres en la visión de la FSF. De modo que, aunque se puede argumentar que históricamente el movimiento libre de la FSF ocurrió primero, desde el punto de vista de la clasificación empezamos con el concepto más amplio.

### La definición de código fuente abierto (Open Source Definition)

La OSD fue diseñada para establecer una declaración abierta y comprensiva de los principios del movimiento de software abierto y una

manera de clasificar y “certificar” la multitud de licencias libres que existen. Se argumenta que, al establecer estándares de esta manera, la definición permite a desarrolladores, a usuarios, a organizaciones comerciales y a la Administración pública entender mejor el movimiento de software libre y respetar más sus principios.



Es importante entender que la OSD no es una licencia, ni un modelo de licencia, sino que establece directrices para la clasificación de licencias relativas a aplicaciones y productos de software en sus diversas formas (componentes, programas, distribuciones completas).

Es más, la OSI ha elaborado una marca de certificación, la marca OSI Certified, que es una manera ostensible de indicar que una licencia cumple con la OSD. La marca sirve también para diferenciar el término general *open source*, que no tiene un uso suficientemente definido para garantizar esta conformidad.

La OSD surge de las Directrices Debian de Software Libre (*Debian Free Software Guidelines*), adaptadas en 1998 básicamente por la eliminación de las referencias a Debian. En efecto, la definición de software abierto en las DFSG era suficientemente amplia como para incluir las licencias de tipo BSD, la GPL y su hermana LGPL, así como la del MIT/X y la Apache. Por lo tanto, sus requerimientos fueron adoptados por la OSI como las pautas generales que toda licencia abierta debe cumplir.

La definición OSI v 1.0 enfatiza los **cuatro elementos fundamentales** del movimiento de software libre, recogidos en las cuatro libertades enumeradas por la FSF:

- (0) El libre uso.
- (1) La libre reproducción y distribución.
- (2) La disponibilidad y el acceso al código fuente.
- (3) El derecho de crear obras derivadas.

La OSD ha sido modificada varias veces desde su primera versión 1.0. La versión actual (versión 1.9) tiene diez criterios. La última modificación

es la inclusión de la décima directriz efectiva relativa a la neutralidad de la tecnología (antes, en el décimo lugar, había una lista de ejemplos).



Hallaréis la OSD en el sitio web de la OSI: [www.opensource.org](http://www.opensource.org) o, en castellano, en:

<http://gd.tuwien.ac.at/orgs/www.OpenSource.org/docs/osd-spanish.php>

(no incluye la nueva directriz 10). Esa traducción no es oficial y en el texto que reproducimos a continuación hemos modificado algunos términos, sobre todo los relacionados con la traducción de *must not* ('no debe', que muchas veces se ha traducido con el término confuso 'no tiene que'). También os aconsejamos leer algunos textos que comentan estas directrices, entre los cuales se encuentran "The Open Source Definition" de Bruce Perens.

A continuación reproducimos la lista de directrices de la OSD en su versión no oficial en castellano, junto con algunos comentarios que indican la razón y los efectos de cada una.

Tabla 2.

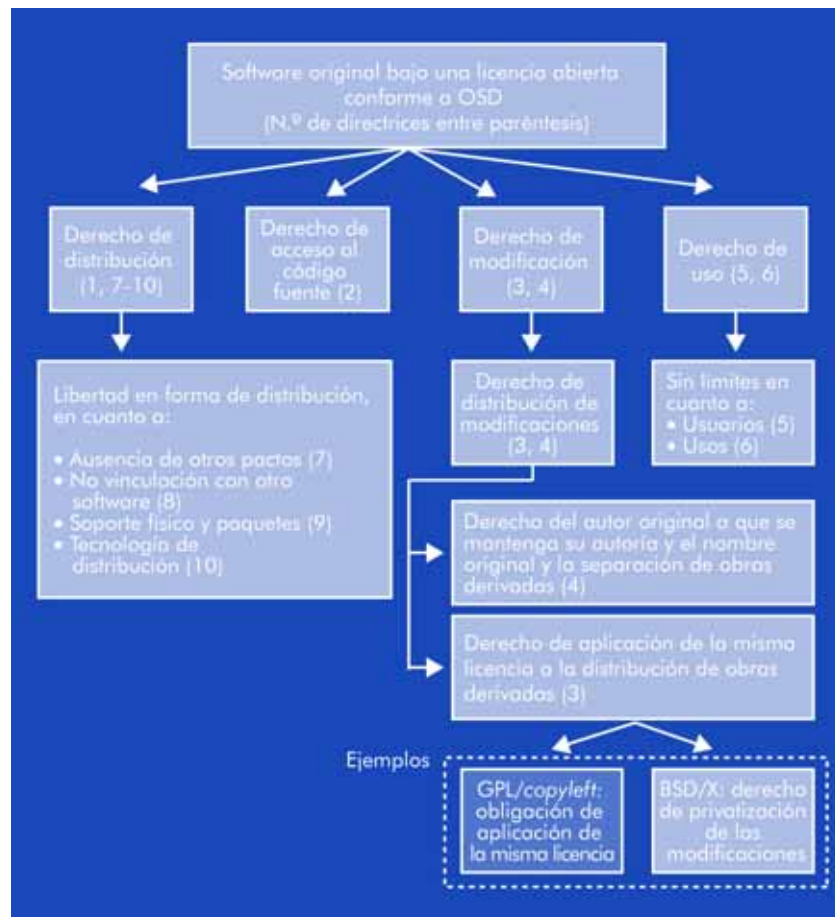
Directriz y comentario	
1	<p><b>Redistribución libre.</b> La licencia no deberá impedir la venta o el ofrecimiento del software como un componente de una distribución de software agregado conteniendo programas de varias fuentes distintas a ninguna parte. La licencia no deberá requerir el pago de los derechos de autor u otra tasa por dicha venta.</p> <p>Garantiza el derecho de distribución. Implica que un usuario puede copiar y vender o distribuir gratuitamente el software abierto. (iSin embargo, no indica que el software <b>tiene que</b> ser distribuido gratuitamente!)</p>
2	<p><b>Código fuente.</b> El programa tiene que incluir el código fuente y tiene que permitir la distribución tanto en código fuente como en forma compilada. Si alguna forma de un producto no es distribuida con el código fuente, tiene que haber un medio bien-publicado de obtener el código fuente por no más que un costo razonable de reproducción preferentemente, una descarga a través de Internet sin cargo. El código fuente tiene que ser la forma preferida en la cual un programador modificaría el programa. El código fuente deliberadamente escondido no está permitido. Las formas intermedias, tales como la salida de un preprocesador o traductor no están permitidas.</p> <p>El software debe incluir el código fuente y la licencia debe permitir que se realice distribuciones de código binario, siempre y cuando la forma de obtener el código fuente esté indicada claramente. El código fuente es necesario para que los destinatarios tengan la libertad de modificar el programa.</p>
3	<p><b>Obras derivadas.</b> La licencia tiene que permitir modificaciones y obras derivadas, así como su distribución bajo los mismos términos de la licencia del software original.</p> <p>Garantiza el derecho de modificación (para los arreglos, adaptaciones, extensiones, etc.) y de distribución de las modificaciones. La licencia debe permitir realizar las modificaciones y adaptaciones del software para crear obras derivadas. Además, se debe <b>permitir</b> que estas modificaciones sean distribuidas bajo los mismos términos que la licencia original del software. Esto no implica que la obra derivada <b>deba</b> distribuirse bajo estos términos. La BSD, por ejemplo, permite modificar el software original y comercializar la modificación en formato binario solamente, a diferencia de la licencia GPL (que es compatible con esta directriz) que obliga a mantener las obras derivadas bajo GPL.</p>



Directriz y comentario	
4	<p><b>Integridad del código fuente del autor.</b> La licencia puede impedir que el código fuente sea distribuido en forma modificada <i>solamente</i> si la licencia permite que la distribución de “archivos parches” con el código fuente con el objetivo de modificar el programa en el tiempo de construcción. La licencia tiene que permitir explícitamente la distribución del software construido a partir del código fuente modificado. La licencia puede requerir que las obras derivadas tengan un nombre distinto o un número de versión distinto al del software original.</p> <p>Garantiza el derecho de distribución de las obras derivadas y permite mantener la autoría de cada componente de una aplicación. Una manera permitida de mantener esta separación es la de obligar a distribuir una obra modificada como (1) la aplicación original más (2) un “parche” que integra la modificación al software original en el momento de la instalación (o construcción –<i>build time</i>). Otra manera de proteger la autoría es con un control de la nomenclatura de las versiones. Se permiten estas restricciones particulares sobre la distribución de los programas derivados para que el autor original pueda proteger su reputación, ante posibles problemas en el funcionamiento del software causados por una modificación o ante una diferencia de “calidad” en el desarrollo de la modificación.</p>
5	<p><b>La no-discriminación con respecto a las personas o grupos.</b> La licencia no debe discriminar a ninguna persona o grupo de personas.</p> <p>Garantiza un uso más amplio del software abierto en cuanto a las personas (usuarios). No se puede restringir el uso por motivos políticos, religiosos, etc. Asimismo, notad que si el marco legal puede imponer restricciones de uso (como por ejemplo, la criptografía en EE.UU.), la licencia misma no puede incorporarlas.</p>
6	<p><b>La no-discriminación con respecto a los sectores de actividad.</b> La licencia no debe restringir a nadie que haga uso del programa en un sector de actividad específico. Por ejemplo, no puede impedir que el programa sea usado en un negocio o que sea usado para una investigación genética.</p> <p>Garantiza un uso más amplio del software abierto en cuanto a las áreas de uso: no se pueden restringir los usos privados, comerciales, educativos o militares. Notad que se amplían al máximo los usos del software.</p>
7	<p><b>Distribución de la licencia.</b> Los derechos adjuntos al programa tienen que aplicarse a todos aquellos que reciben el programa sin la necesidad de ejecutar una licencia adicional para estas partes.</p> <p>La licencia abierta no debe obligar a los usuarios a firmar un “consentimiento”, ni por la licencia ni por cualquier cláusula o pacto adicional (una carta de confidencialidad, por ejemplo). Asimismo, la falta de procedimientos adicionales es necesaria para permitir a licenciarios segundos y terceros aprovechar los derechos especificados en la licencia y quedarse vinculados por las obligaciones correspondientes. Ya hemos comentado en las unidades anteriores que este mecanismo puede entrar en conflicto con el marco legal de los derechos de autor y de contrato en relación con la necesidad de prestar consentimiento por parte del licenciario.</p>
8	<p><b>La licencia no tiene que ser específica de un producto.</b> Los derechos adjuntos al programa no tienen que depender de que el programa forme parte de una distribución particular de software. Si el programa es extraído de esa distribución y es usado o distribuido de acuerdo a los términos de la licencia del programa, todas las partes a las que el programa sea redistribuido deben tener los mismos derechos que son garantizados en conjunto con la distribución original del software.</p> <p>Garantiza un uso más amplio del software abierto en cuanto a la forma de distribución utilizada. Los derechos que otorga la licencia no deben ser diferentes para un software incluido en una distribución original y para el mismo software redistribuido de manera diferente o separada. Es decir, no se puede restringir una versión de Linux a su uso con un paquete determinado de distribución. La versión de Linux queda abierta, aun separada del paquete de distribución original.</p>
9	<p><b>La licencia no debe limitar a otro software.</b> La licencia no debe imponer restricciones sobre otro software que se distribuya junto con el software licenciado. Por ejemplo, la licencia no tiene que insistir en que todos los otros programas distribuidos en el mismo medio tengan que ser software de código fuente abierto.</p> <p>Garantiza una forma más libre de distribución: la licencia no debe poner límites sobre el software que se distribuya con el mismo. Por ejemplo, la licencia no debe obligar a que todos los programas distribuidos conjuntamente con el software en cuestión sean libres o abiertos. Como consecuencia, se puede distribuir software GPL, BSD y propietario sobre un mismo CD. Es importante notar la diferencia entre:</p> <ul style="list-style-type: none"> <li>• “agregar” aplicaciones lógicamente separadas sobre un mismo soporte –la flexibilidad de distribución garantizada por esta directriz; y</li> <li>• “agregar” aplicaciones no lógicamente separadas, es decir, reunidas para crear una obra derivada. Esta directriz no trata de la derivación.</li> </ul>
10	<p><b>La licencia debe ser neutra respecto de la tecnología.</b> Ninguna disposición de la licencia debe predicar una tecnología o un tipo de interfaz particular.</p> <p>La aceptación de la licencia no debe depender del uso de una tecnología o interfaz. Esta directriz se refiere a licencias que pueden obligar a usar determinados sistemas tecnológicos para prestar el consentimiento a la licencia (por ejemplo, el uso de <i>click-wrap</i>). Hay otras maneras de distribuir el código y otras interfaces posibles (FTP, interfaces no gráficas) que pueden ser independientes o incompatibles con la especificación de una tecnología en particular.</p>

Para ilustrar estas directrices de manera simplificada, establecemos un diagrama esquemático que resume los derechos que se debe garantizar al usuario-licenciatario de software bajo una licencia abierta.

**Figura 1.** Diagrama esquemático de los derechos mínimos otorgados por una licencia abierta OSD



Obsérvese que bajo los criterios de la OSD, no hay ninguna **obligación** de redistribución de obras modificadas bajo la misma licencia (la obligación *copyleft* que se incluye en la licencia GPL, indicada en gris en el diagrama), sino un **derecho**.

La OSD trata de reconciliar las libertades del software libre (en general) con las necesidades comerciales de las empresas implicadas en la creación, la distribución y aun el uso de software libre. Sostiene los derechos fundamentales para cualquier aplicación libre (copia, modificación, distribución, acceso a código fuente), mientras que la distribución posterior de obras modificadas permanece flexible.

**Experiencia**

Un ejemplo interesante de la aplicación de la OSD surge en el caso de KDE, Qt y la empresa Troll Tech. KDE es un intento de crear una interfaz gráfica de escritorio para Linux (similar a Windows). KDE depende de unas bibliotecas gráficas llamadas Qt, que pertenecen a Troll Tech. Sin embargo, la licencia de Qt no cumplía con la OSD, dado que se requería una licencia especial de desarrollador para incorporar dichas bibliotecas en aplicaciones que no fueran X Windows System (Qt obtenía ingresos por la cesión de licencias a Microsoft y Macintosh). Por lo tanto, una aplicación libre como KDE tenía incorporados elementos que no se consideraban libres. Bajo la presión de la comunidad libre y, en particular, de la OSI Troll Tech acordó en un primer paso crear una licencia especial para liberar el código Qt en el caso de la fusión o la quiebra de la empresa. Luego, con el inicio del desarrollo de GNOME, un producto abierto directamente competitivo con KDE y la creación de bibliotecas libres similares a Qt (Harmony), Troll Tech modificó su licencia para que cumpliera con la OSD.

**El software con copyleft robusto**

Ahora que hemos visto los requisitos generales para que una licencia se considere abierta –y para algunos, eso es sinónimo de libre– es primordial considerar el concepto de *libertad* de la Free Software Foundation y su implementación a través de la licencia GPL. Ya hemos comentado que la GPL cumple con la OSD, por lo que se considera una licencia abierta. Sin embargo, contiene otros requerimientos, permitidos bajo la OSD, que la hacen muy particular. Ello no solamente porque es la licencia más usada en el mundo del software libre con un 75% del código libre publicado, ni porque es el precursor de muchas licencias libres actuales (aunque no todas –la BSD precede), sino porque la filosofía de libertad de la FSF ha sido la base y uno de los elementos más destacados del movimiento libre.

El objetivo, muchas veces repetido, de R. Stallman y la FSF es garantizar la libertad de uso del software y mantener esta libertad en relación con obras derivadas de software originalmente libre.

**Nota**

La FSF trata de mantener la libertad del software en distribuciones posteriores de software.

En el apartado 7.1.1 hemos comentado lo que significa “libre” para ellos: gozar de las cuatro libertades fundamentales. Las directrices de la OSI, aunque la FSF pueda no estar del todo de acuerdo con ellas, son una primera aproximación a esta libertad. Sin embargo, para la FSF no son suficientes. Lo importante desde la perspectiva de la FSF es que una aplicación es realmente libre solamente si las distribuciones posteriores –del mismo software o de obras derivadas– permanecen libres. Es decir, la licencia que se aplica al software incluye términos de redistribución que no permiten a los redistribuidores añadir a su licencia cualquier restricción adicional (a las de la licencia original GPL), ni al software original, ni a obras derivadas.



Esta condición particular, que se llama *copyleft*, establece la imposibilidad legal de “capturar” el software libre, modificarlo y privatizarlo. Por lo tanto, el pool, o la cantidad de software con *copyleft* disponible, no puede más que aumentar a medida que los desarrolladores crean nuevas aplicaciones a partir del software con *copyleft*.

En el apartado siguiente analizaremos en detalle el mecanismo legal con el cual la licencia GNU GPL (que llamaremos simplemente “la GPL”) implementa esta filosofía y por qué ha sido llamado “virus” por algunos y “herencia” por otros.



Es importante darse cuenta de que el *copyleft* no afecta a los derechos del licenciatario original (por ejemplo, un simple usuario), sino que restringe las libertades relativas a la distribución posterior de cualquier código GPL (original, modificado, o íntimamente incorporado en otras aplicaciones).

Comprender esto permite entender por qué la cláusula *copyleft* de la GPL no impide el uso comercial de aplicaciones bajo GPL o con *copyleft* en las organizaciones privadas o públicas: porque dichas organizaciones son usuarios finales.

A diferencia de la OSI, que usa la OSD para determinar si una licencia es abierta o no, la FSF clasifica una licencia según los criterios siguientes:

- a) Si califica como licencia de software libre (es decir, que cumple con las libertades 0, 1, 2 y 3).
- b) Si es una licencia con *copyleft* (o *copyleft robusto*, como se ha dicho a veces).
- c) Si es compatible con la GPL.
- d) Si causa algún problema práctico en particular.

De esta calificación surgen tres tipos de licencias:

- Licencias de software libre compatibles con la GPL.
- Licencias de software libre incompatibles con la GPL.
- Licencias de software no libre.

Dada la importancia práctica de la compatibilidad con la licencia GPL, usaremos esta clasificación para nuestro análisis en el apartado 7.3.

Aquí no comentaremos los efectos comerciales, tecnológicos o económicos de la cláusula de *copyleft*, que han contribuido a lo que se ha llamado una *creación destructiva* o una *metodología disruptiva* de desarrollo de software. Sin embargo, es importante observar que los aspectos legales de la cláusula de *copyleft* han provocado mucha preocupación en el mundo del software en general. Sobre todo, se ha temido que la vinculación o incorporación de código bajo la GPL afecte al uso o la distribución comercial de la aplicación resultante o que el uso de software bajo GPL (por ejemplo, GNU/Linux) impida el uso de otras aplicaciones propietarias. Estas dudas, que son más bien mitos, se considerarán en el apartado siguiente 7.2.

### Las similitudes y diferencias entre licencias abiertas y con *copyleft*

Aunque no se puede decir que la OSI y la FSF sean dos movimientos opuestos, es cierto que persiguen objetivos diferentes que se plas-

#### Nota

Podéis ver también la compatibilidad con la GPL en el apartado 7.3 y la importancia de la compatibilidad entre licencias en general en el apartado 8.1.3.

**Lectura complementaria**

Para una visión particular: Richard Stallman. *Why “free software” is better than “open source”* (en bibliografía).

man en las licencias libres. Por un lado, la FSF defiende la libertad de uso y distribución del software “a cualquier precio”... aun al coste de perder el apoyo y las contribuciones de desarrolladores que quieren mantener un control económico sobre sus obras derivadas. Por el otro, la OSI ha apaciguado el conflicto aparente entre el desarrollo comercial y el mundo del desarrollo libre, ganando adeptos y defensores en el mundo empresarial.

**a) Las similitudes**

Hay similitudes evidentes entre una licencia abierta y una licencia con *copyleft*:

- Ambas garantizan las cuatro libertades esenciales de software libre mencionadas en el primer apartado.
- Tanto la OSD como la GPL incluyen la obligación de proveer acceso al código fuente (*open source*, que podría llamarse más correctamente *available source*, es decir, código disponible).
- Una licencia con *copyleft* cumple con la OSD, y por tanto es software abierto (ved la tabla a continuación).

A continuación, mostramos cómo la licencia GPL cumple con las directrices de la OSD: no agrega ninguna restricción que éstas no permitan. Otros elementos añadidos en la licencia se comentarán a continuación en el apartado 7.3.

**Tabla 3.** Conformidad de la GPL con las directrices OSD

#	Criterio OSD	Cláusula GPL
1	Redistribución libre.	1: Permite copiar y distribuir.
2	Código fuente.	3: Obliga a acompañar el original (y obras derivadas) con su código fuente (u ofrecerlo).
3	Obras derivadas.	2: Permite modificar y distribuir obras derivadas.
4	Integridad del código fuente del autor.	No hay obligaciones de integridad. Pero requiere avisos de autoría.

#	Criterio OSD	Cláusula GPL
5	No-discriminación con respecto a las personas o grupos.	0: Permite ejecutar el programa sin restricciones. 8: Se reserva el derecho de discriminar por razones geográficas en el futuro.
6	No-discriminación con respecto a los sectores de actividad.	0: Permite ejecutar el programa sin restricciones.
7	Distribución de la licencia.	5 y 6: Vincula al licenciataria por la modificación o distribución y transmite la licencia a terceros.
8	No-específica de un producto.	No restringe.
9	No-restringir a otro software.	No restringe.
10	Neutralidad respecto de la tecnología.	No restringe.

### b) Las “diferencias”

Aunque no se puede hablar realmente de diferencias, dado que la GPL es una licencia abierta conforme a la OSD, destaquemos los elementos legales que particularizan la GPL y otras licencias *copyleft*:

- 1) Primero, la GPL y similares son licencias, mientras que la OSD es una lista de directrices para crear una licencia abierta. Por lo tanto, en términos de comparación, habría que considerar las licencias abiertas, como Apache, BSD, MPL, etc., y no solamente los criterios de la OSD.
- 2) La particularidad más importante es la obligación de *copyleft* añadida en la GPL, que exige la persistencia de la libertad de uso y explotación sobre los programas derivados (la herencia de la libertad). Las licencias “abiertas sin *copyleft*” como la BSD, son más “permisivas”. El criterio 3 de la OSD permite privatizar el software abierto, es decir, variar las condiciones de la licencia abierta relativas a la aplicación original para imponer nuevas condiciones “cerradas” sobre la distribución de un programa derivado. La directriz 4 permite mantener la integridad del código original y, a diferencia de la GPL, autoriza al licenciante a exigir a los usuarios que no distribuyan el software modificado directamente, sino que lo distribuyan en dos partes, acompañando los archivos fuentes originales con “archivos parche” separados.

En la realidad, salvo esta cláusula de *copyleft* que instrumenta las diferencias conceptuales entre software *copyleft* y software abierto, las discrepancias no son legales sino de postura.

#### Nota

Podéis ver en la unidad 1 un comentario sobre la historia de la FSF y la OSI.

Dada la importancia del software con *copyleft* sobre todo el sistema operativo GNU/Linux, lo esencial desde un punto de vista práctico-legal es ver en qué medida una licencia abierta es o no compatible con la licencia GPL, es decir, determinar si las aplicaciones correspondientes pueden, desde el punto de vista legal, vincularse o integrarse entre sí. En un paradigma de arquitecturas de software distribuidas y “por componentes”, este tema de la interrelación entre módulos de software es fundamental: si dos programas no pueden integrarse o comunicarse lógicamente, no se puede hablar de plataforma integrada ni interoperable.

**Nota**

Este tema se comentará en el apartado 7.3, después de aclarar algunas dudas o “mitos” sobre las licencias libres en general y realizar un análisis más detallado de las principales licencias libres.

## 7.2. Unos mitos legales para desmitificar

Después de este análisis introductorio a las licencias libres, podemos comentar y aclarar varios mitos o conceptos equivocados relativos a diversos aspectos legales de las mismas, antes de profundizar el estudio de algunas entre ellas.

**Nota**

Hay otros “mitos” relativos a los aspectos tecnológicos o comerciales del software libre, que no tratamos aquí: la falta de soporte y mantenimiento, la falta de seguridad, el riesgo de la bifurcación (*forking*), la posibilidad de introducir elementos dañinos en software libre, la carencia de modelo de negocio viable en base al software libre, etc.

**Lectura complementaria**

Para un comentario sobre estos mitos, se recomienda leer: *Dispelling Myths about the GPL and Free Software*, de John Viega and Bob Fleco (en bibliografía).

### 7.2.1. “El *copyleft* está en contra o no respeta el derecho de autor”

Este mito consiste en creer que el software libre (y las licencias asociadas) crea un nuevo marco de derecho de la propiedad intelectual,



el *copyleft*, “en vez de” el *copyright*. Al contrario, tal como lo explican todos los defensores del software libre, las licencias libres se fundamentan directamente en el derecho de propiedad intelectual vigente, ya sea el derecho de autor de estilo continental, sea el *copyright* anglosajón. La FSF y otros autores del movimiento de software libre usan sus derechos de autor otorgados por el marco legal justamente para defender las libertades ofrecidas por sus licencias.

#### Ejemplo

Por ejemplo, aunque no haya ninguna decisión judicial al respecto, Eben Moglen, abogado de la FSF, ha comentado que varias veces ha tenido que amenazar con acciones legales (fundamentadas en el derecho de la propiedad intelectual) a empresas que habían “privatizado” código bajo licencia GPL.

Consideremos, por ejemplo, las dos características principales del software libre, las libertades de uso y el efecto *copyleft*.

- 1) En relación con las **libertades de uso**, el marco legal permite a los titulares definir los derechos de explotación de la obra protegida. En lugar de restringir los usos del licenciante como lo hace la mayoría de las licencias propietarias, la licencia libre los amplía al máximo permitido.
- 2) Respecto al *copyleft*, el autor de una obra derivada puede crearla únicamente porque el titular de la obra original en que se basa se lo permite, bajo ciertas condiciones. Si estas condiciones –por ejemplo, la de distribuir la obra derivada bajo la misma licencia– no se cumplen, la licencia original se resuelve y la obra derivada es una violación de los derechos originales. Por ello, se ha dicho que el *copyleft* es legal como cláusula resolutoria.

Por lo tanto, no hay contradicción ni oposición entre los derechos de autor legislados y los derechos bajo una licencia libre. Es más, se puede argumentar que en tanto que una licencia libre respete las excepciones y los usos permitidos del usuario bajo este marco legal, se ajusta más al derecho que muchas licencias propietarias.



“La licencia GPL no agrega nada al derecho de autor [restricciones de uso, etc.]. [...] El derecho de autor otorga a los titulares los poderes de prohibir el ejercicio de derechos de copia, modificación y distribución, derechos de los que nosotros consideramos que los usuarios deben beneficiarse. Por lo tanto, la GPL elimina las restricciones permitidas por el sistema de propiedad intelectual.”

Eben Moglen. *Enforcing the GNU GPL*.

<http://linuxtoday.com/developer/2001091801320OPLL>

### 7.2.2. “El software libre no tiene titulares o propietarios u obliga a ceder sus derechos de autor”

No hay nada más equivocado desde el punto de vista legal. El marco jurídico de la propiedad intelectual confiere derechos de autor automáticamente a los creadores del software (o, en algunas circunstancias, a las empresas contratantes de los creadores). Casi la única obligación compartida por todas las licencias libres es la de mantener los avisos de titularidad de los creadores iniciales del software (el famoso *Copyright Notice*). Y, como lo indica E. Moglen de la FSF, estos titulares de los derechos actuarán con fuerza para defenderlos.

#### Experiencia

En el caso MySQL AB contra Progress Software, MySQL AB defiende su titularidad en la aplicación de base de datos MySQL y los derechos asociados. Inició un juicio contra Progress Software por violación de derechos de autor y de la licencia GPL sobre el programa MySQL.

Estos derechos se pueden ceder, pero únicamente con el consentimiento explícito del autor (con la excepción de los derechos morales sobre el software, en países donde estén reconocidos, los cuales no se pueden ceder). Por lo tanto, las licencias no pueden quitar la titu-

#### Nota

Ver el caso MySQL AB contra Progress Software en la unidad 8.

laridad del software a sus creadores. Algunas licencias libres obligan a distribuir o publicar el código fuente de obras nuevas, que se consideran modificaciones u obras derivadas de software original, pero no a ceder el código o los derechos del titular sobre éste.

### **7.2.3. “No se puede hacer un uso comercial del software libre”**

Otra creencia equivocada: como hemos visto, no hay límites sobre el uso del software libre, solamente algunas condiciones sobre su modificación y distribución posterior. Las licencias libres no afectan a los usuarios finales. No obstante, es cierto que algunas licencias cuya clasificación es “lingüísticamente” similar a las de software libre (*freeware* y *shareware*) prohíben el uso comercial de las versiones gratuitas. Estas licencias no se consideran ni abiertas ni libres (por ejemplo, a menudo el software de este tipo no se distribuye con el código fuente).

### **7.2.4. “El software libre y el software propietario son incompatibles”**

Otro mito es que una licencia libre (y por lo tanto, el software libre) es incompatible con una licencia y el software propietario, ejecutados en un mismo sistema o plataforma informática. Si esto fuera cierto, ninguna aplicación propietario, como las bases de datos de Oracle o las aplicaciones de IBM, podría ejecutarse sobre Linux, OpenBSD o los servidores web Apache. Y viceversa, aplicaciones libres como MySQL no podrían ejecutarse sobre sistemas operativos propietarios como UNIX, Solaris de Oracle o AIX de IBM. Justamente, por ejemplo, Samba existe para relacionar aplicaciones libres y el SO Windows en un mismo sistema o red. Lo que sí que puede suscitar incompatibilidades es la integración o mezcla de software con *copyleft* y software propietario, lo que comentaremos a continuación.

### **7.2.5. “No se puede integrar o mezclar código libre y código propietario”**

Esta afirmación sostiene que el código libre (en general) no puede mezclarse o integrarse con código propietario en una misma aplicación, sin afectar a dicho código propietario y, por lo tanto, sin violar

las condiciones de uso de éste. Una manera más fuerte para expresar esto es afirmar que el software libre, y el código bajo GPL en particular, es “vírico” e “infecta” a otras aplicaciones: cualquier aplicación que integre código GPL vuelve a ser código bajo GPL. Esta afirmación es parcialmente falsa.

Hay que tomar en cuenta los aspectos siguientes:

- a) **Integración por el usuario final:** desde el punto de vista del programa libre, las licencias no restringen sus usos con otras aplicaciones propietarias. La posibilidad de modificación es una condición de ser libre y no hay restricciones sobre su uso. Y es por ello que se ha de distribuir el código fuente con el código objeto o ponerlo a disposición del destinatario. Sin embargo, cualquier integración de código libre (A) con un software propietario (B) podrá ser considerada como una modificación de los dos software en cuestión (y es realizable únicamente si uno tiene el código fuente de B!). Esto es permitido por la licencia libre del software (A). Sin embargo, dependiendo de las restricciones contenidas en la licencia propietaria (B), dicha modificación puede constituir una infracción de la misma. Esto no es un problema del software libre, sino de la licencia de software propietario.
- b) **Integración por un intermediario:** donde sí que puede haber restricciones relativas a la integración de software de distintos tipos, ya sea libre, abierto o propietario, es respecto de su distribución posterior. Sobre todo, la condición de *copyleft* robusto de las licencias de tipo GPL prohíbe la “integración” de código bajo la GPL con código propietario en una distribución posterior bajo licencia propietaria, una práctica que se ha llamado la “privatización” del software libre. Esta restricción se aplica a cualquier distribución propietaria del código objeto (sin adjuntar el código fuente) o con condiciones incompatibles con la licencia GPL. Hay ciertas licencias libres que contienen cláusulas que tratan de permitir esta integración, como la LGPL o la MPL en algunas circunstancias, que veremos en el apartado siguiente.

En este sentido, la cuestión de si uno puede integrar un software libre con cualquier otro para su distribución posterior, depende de la forma en que esto se realice. El tratamiento del resultado de esta integración, lo que se considera una “obra derivada”, es complejo y lo

estudiaremos a continuación, en el análisis de la condición de *copyleft* robusto en la licencia GPL y los derechos otorgados en la LGPL.

Asimismo, en cuanto al código bajo licencia propietaria, hay que tener en cuenta los derechos del usuario que descompila el programa con el fin de su interoperabilidad con otro (que puede ser libre) y lo modifica por necesidades legítimas del usuario (para cuyo uso el software se adquirió) y, eventualmente, para la corrección de errores (por ejemplo, si se produce un error cuando interactúa con un software libre). Estos derechos están sujetos a las condiciones que se han comentado en la unidad 2.

### Reflexión

La relación e integración entre software libre y propietario depende en gran medida de las formas de interacción tecnológica entre sus aplicaciones o sus componentes. Según los diseños y lenguajes de programación, se habrá de considerar la compilación, la interpretación, la simple ejecución, las llamadas a funciones, rutinas o bibliotecas, los vínculos estáticos y dinámicos y las interfaces API, entre otros. En entornos distribuidos, estas relaciones se complican, por ejemplo, con componentes CGI, ASP y otras formas de interactuar con programas a distancia.

En algunos casos no se considerará “modificado” el programa inicial y, por tanto, en caso de código GPL no se aplicaría el *copyleft* sobre la distribución posterior. En otros, esta interacción o integración implicará una modificación, permitida por el autor original del software libre bajo las condiciones de la licencia: se aplicará el *copyleft* en caso de la GPL pero no hay restricciones en el caso de la BSD.

El entendimiento de estas relaciones es fundamental para analizar una propuesta técnica y decidir sobre las licencias posibles o los componentes libres con los cuales un programa puede interactuar o integrarse.

### 7.2.6. “Todo el software libre es igual, bajo los términos de la GPL”

Ya hemos comentado que hay variaciones sustanciales entre las licencias libres y veremos a continuación el detalle de estas diferencias. Habría que ser mucho más cuidadoso en el uso del término *software libre*, así como distinguir a menudo entre licencias libres propiamente dichas (en la interpretación de FSF), licencias abiertas y licencias que no son ni libres ni abiertas. Es importante manejar con claridad los términos *código abierto*, *persistencia* y *copyleft*, que son característicos de estas licencias libres.

### 7.2.7. “Las licencias libres obligan a publicar sus modificaciones particulares”

Esta es una de las ideas falsas más propagadas sobre el funcionamiento de las licencias libres. Consideremos otra vez la posición de los usuarios finales e intermedios (desarrolladores de programas para terceros):

- a) **Usuarios finales:** la mayoría de las licencias libres no obligan a los usuarios ni a distribuir sus modificaciones o adaptaciones de software libre (obras derivadas, en lenguaje legal), ni a publicarlas o contribuir con ellas al desarrollo de la aplicación modificada. Algunas licencias requieren esto último sólo en relación con correcciones o modificaciones del código central o núcleo del programa. Como veremos, estas obligaciones no se aplican a elementos adicionales agregados al núcleo o a cualquier extensión de la aplicación. Por lo tanto, los usuarios finales no tienen que publicar sus obras basadas en software libre.
- b) **Los profesionales y las empresas desarrolladores de programas:** las personas que desarrollan para clientes tampoco tienen que distribuir al público o a los autores originales cualquier modificación de un software libre, pero sí que tienen que respetar la licencia libre original, muchas de las cuales obligan a proveer el código fuente a los usuarios/clientes destinatarios o, si sólo se distribuye el código objeto, ofrecer el código fuente a cualquier tercero (la GPL) o al destinatario (la MPL). Éste es uno de los requisitos para utilizar software libre y abierto.

**Ejemplo**

La licencia Apple Public License 1.x obligaba a remitir a Apple cualquier modificación del programa original y era una de las razones por las cuales no se consideraba una licencia libre.

**7.2.8. “Nadie es responsable por el software libre, ni tiene garantía”**

Hay que admitir que esto es cierto, bajo las licencias actuales de software libre, aunque haya dudas legales sobre la efectividad de las cláusulas de negación de garantía y de responsabilidad. Esto ya se ha comentado con respeto al marco jurídico en la Unión Europea. El mito, en realidad, consiste en pensar que los licenciantes propietarios aceptan mayor nivel de responsabilidad. Hemos visto que la licencia típica de software propietario también intenta limitar la responsabilidad del licenciante (autor o distribuidor), muchas veces al precio pagado por la aplicación o una suma similar.

Con los sistemas de distribución virtual en Internet, se podría argumentar también que es difícil identificar a los licenciantes y con ello recurrir a alguna indemnización. Muchos sitios de distribución de software libre, como Sourceforge, no son los titulares licenciantes, ni siquiera distribuidores “oficiales”. No obstante, en algunos casos, como el de la FSF o en casos de negocios basados en la distribución de paquetes de software libre como Red Hat o Suse, hay una entidad legal identificable contra quien se podría intentar una acción por daños y perjuicios, si fuera necesario. Además, la obligación de mantener el aviso de autoría (*copyright notice*) permite identificar a los autores de cualquier componente deficiente, aunque no son necesariamente los que hayan distribuido el programa al perjudicado.

El mismo argumento se aplica a las garantías. Las licencias libres en sí mismas no ofrecen garantías, pero tampoco son de mucha utilidad las de las licencias propietarias. Como se ha visto, muchas veces su garantía contractual está limitada, por ejemplo, a la devolución del precio de compra en caso de una avería, dentro de un límite de 90 días, sin garantizar el funcionamiento adecuado de las aplicaciones. Por un lado, hay que considerar las garantías obligatorias por ley, que se aplican a software libre y propietario. Por otro lado, las licen-

**Nota**

Podéis ver las cláusulas de negación de garantía y de responsabilidad en las unidades 4 a 6.

cias libres permiten a los distribuidores de software libre agregar cláusulas de garantía (con contraprestación económica o no), lo que se hace con muchos paquetes de distribución comercial.

**Nota**

Luego de esta introducción a las licencias libres y una presentación de las diferentes categorías de licencia, pasemos a considerar las principales licencias libres para estudiar sus particularidades y efectos.

### 7.3. Estudio particular de las licencias de software libre

A los fines de estudio, hemos clasificado las licencias abiertas y libres en cuatro categorías que examinamos en este apartado. Estas categorías se distinguen de varias maneras. No obstante, la mayor diferencia reside en cómo tratan las obras derivadas y modificaciones y en la posibilidad de privatizar el código original (el *copyleft*). Las cuatro categorías son:

- 1) La GPL y las licencias libres con cláusulas de *copyleft* robusto.
- 2) Las licencias de tipo BSD, que incluyen las licencias MIT y X, compatibles con la GPL.
- 3) Las licencias más complejas, como la Mozilla Public License (MPL) o Apple 2.0, que no son compatibles con la GPL.
- 4) Otras “licencias libres” que no son libres, aunque se intenten aprovechar del modelo de desarrollo libre (por ejemplo, la Microsoft Shared Source Initiative).

Y como cualquier programa es casi inútil sin su documentación técnica correspondiente, a continuación consideraremos unas licencias de documentación libre.

Después de una introducción de cada licencia, presentaremos los derechos otorgados, las obligaciones y restricciones impuestas y cualquier otro elemento importante. Luego comentaremos la licencia para destacar algunos aspectos prácticos útiles para su comprensión y uso. Para los elementos más importantes indicamos las cláusulas relevantes, a fin de que el lector pueda encontrar la disposición en la licencia.

**Nota**

Recomendamos descargar de Internet las licencias y tenerlas a mano durante el estudio.



**Nota**

Durante nuestro análisis vamos a poner en práctica las nociones de derecho que hemos aprendido en las unidades anteriores. Por ello, creemos oportuno recordar algunos de los conceptos legales fundamentales que se han estudiado:

- a) **Origen de derechos:** los derechos de autor se confieren automáticamente a los creadores de software y a veces a las empresas contratantes.
- b) **Cesión de derechos:** solamente los autores y otros “proveedores-licenciantes”, titulares de los derechos, pueden especificar el alcance de la licencia y los derechos otorgados. Los licenciatarios originales no pueden conceder a sus cesionarios (secundarios) más derechos que los que ellos mismos hayan recibido.
- c) **Obras derivadas:** cualquier modificación de una obra original constituye una obra derivada pero nueva, propiedad del “modificador”. Sin embargo, los derechos del modificador están restringidos por los derechos del autor original y las obligaciones impuestas por él en la licencia original.
- d) **Condiciones de licencia:** la naturaleza de la licencia, como contrato o simple autorización no pactada (bajo el derecho anglosajón), determina el alcance y la validez de las obligaciones impuestas. Una simple autorización no puede imponer obligaciones mayores que las condiciones de control exclusivo otorgadas por la ley y estrictamente relacionadas con el uso del software.
- e) **Patentes:** el derecho de las patentes no controla la copia, distribución y modificación de una obra, sino el uso y explotación de la invención o proceso patentado, ya sea copiado o recreado. Los derechos de patente también pueden cederse por licencia. Una patente sobre un proceso informático restringirá sustancialmente su difusión y utilidad para la comunidad libre.

### 7.3.1. Las licencias de software libre con copyleft

En este apartado explicaremos en detalle las dos licencias con *copyleft* que constituyen las bases fundamentales del movimiento de software libre y son modelo o inspiración de muchas otras licencias libres: la GPL y la LGPL. Al final del apartado comentamos algunas licencias que tienen cláusulas de *copyleft*, pero no son necesariamente compatibles con la GPL porque permiten acciones o imponen obligaciones incompatibles con la GPL.

#### 1) La Licencia Pública General GNU (la GPL)

Creada en 1986, se ha descrito a la GPL como “una parte manifiesto político y otra parte licencia”. En su preámbulo contiene una enunciación de la filosofía del software libre y un resumen sencillo de la licencia. En la parte principal especifica los derechos otorgados a los usuarios y las condiciones y limitaciones aplicadas a la licencia. La filosofía ya se ha comentado varias veces y no vamos a añadir nada más, salvo que, en caso de duda sobre cualquier elemento de la licencia, el preámbulo será tenido en cuenta para la interpretación de los términos.

Es importante resaltar que pese a su tono familiar y sencillo, la GPL fue diseñada por Richard Stallman junto con varios abogados americanos; por lo tanto, no contiene disposiciones cualesquiera, sino un mecanismo de transmisión y resolución de derechos deliberado y muy sutil. A continuación, examinamos en detalle la licencia, dada su importancia histórica, legal y práctica (se aplica a casi el 75% del software libre disponible). Primero presentamos el contenido de la licencia, luego haremos algunos comentarios sobre la misma.

#### a) Definiciones útiles

La licencia GPL contiene varias definiciones que son de gran utilidad en el momento de interpretar sus efectos:

- **Programa:** el término *programa* se define como cualquier programa u obra a la cual se ha adjuntado la licencia. Técnicamente, podría aplicarse a un fichero de texto, de imagen u otro (cl. 0).

- **Obra basada en el programa:** esta expresión significa el programa original o cualquier obra derivada de él según la definición del derecho de *copyright*. Esto quiere decir una obra que contenga el programa o una parte del mismo, ya sea una copia fiel o literal, o con modificaciones y/o traducida. La licencia, como el derecho de *copyright*, considera una traducción como una modificación de la obra original (cl. 0 y 2).
- **Código fuente:** para un programa ejecutable, el código fuente significa la forma preferida del trabajo cuando se le hacen modificaciones. Para un ejecutable completo, significa el código fuente de todos los módulos que contiene más los archivos con la configuración de la interfaz y los guiones (*scripts*) utilizados para controlar la compilación e instalación. No incluye el código fuente de los módulos equivalentes del sistema operativo donde el programa se ejecuta, salvo que dichos módulos acompañen al ejecutable (cl. 3).

#### b) Los elementos esenciales de la licencia

Los **derechos otorgados** por la licencia garantizan las cuatro libertades fundamentales del software libre. Son:

- El derecho de copia y de distribución del código fuente original (cl. 1).
- El derecho de modificación (cl. 2).
- El derecho de distribución de las modificaciones eventuales, siempre que se distribuyan con la misma licencia GPL y sin cobrar por ella (cl. 2b –la cláusula *copyleft*). El efecto de esta cláusula sobre obras derivadas se comenta más adelante.
- El derecho de copia y de distribución del programa en forma de código objeto o ejecutable, siempre que se ponga a disposición de terceros el código fuente sin cobrar más que el coste de la entrega de dicho código fuente (cl. 3).

Por lo tanto, la GPL permite modificar un programa para su uso personal (privado o comercial) y distribuir copias de estas modificaciones sin necesidad de avisar los cambios a los autores originales (pero

sí a los destinatarios). Asimismo, permite cobrar por la distribución de una copia y ofrecer garantías a cambio de un pago (cl. 1).

El **acceso al código fuente** es un segundo aspecto esencial de la licencia. Un programa se puede distribuir bajo la GPL en formato binario (código objeto), pero se debe acompañar del código fuente o el ofrecimiento de proveerlo a cualquier tercero en un plazo de tres años (cl. 3). Dar acceso al código fuente desde Internet es suficiente para cumplir con esta cláusula.

Otro elemento esencial se refiere a **restricciones, condiciones y actos prohibidos**. La GPL mantiene cierto control sobre el programa, no con respecto a su uso, sino a su transmisión a terceros:

- Cualquier distribución del programa o de una obra derivada del mismo debe estar acompañado de los avisos de autoría, una indicación de cualquier modificación que se hubiera hecho (y la fecha de la misma), el aviso de que no hay ninguna garantía y una copia de la licencia (cl. 1 y 2). Se ha aceptado que indicar el URL de la licencia en [www.fsf.org](http://www.fsf.org) es suficiente.
- No permite copiar, modificar, sublicenciar o distribuir el programa de una manera distinta de la expresamente permitida por la licencia, con menos libertades o mayores restricciones (cl. 2b, cl. 6).
- Si se intenta realizar cualquier acto que consista en una violación de la licencia, el licenciataria pierde sus derechos originales (cl. 4).
- Si se desean incorporar partes del programa bajo GPL a otros programas libres que tengan condiciones de distribución diferentes, se debe obtener el permiso del autor del primero (cl. 10).

#### c) Otros aspectos importantes de la licencia

##### Ámbito de aplicación

- La licencia GPL se aplica a cualquier programa u otra obra, que contenga un aviso del titular del derecho de autor indicando que puede distribuirse bajo los términos de la misma.

- La licencia controla solamente los actos de copia, modificación y distribución del programa. Es decir, no regula ningún otro acto, sobre todo, el uso o la ejecución del programa. La licencia considera que éste no es un acto regulado por el derecho de autor y los usuarios son, por lo tanto, libres de ejecutar el programa de la manera que quieran.

**Autoría:** se respeta el reconocimiento del autor del software. Por ello:

- La licencia obliga a mantener de forma adecuada y bien visible el aviso de autoría sobre cualquier copia del programa (cl. 1). En particular, en el apéndice de la GPL, se recomienda que el autor añada al principio de cada fichero fuente una línea de *copyright*, de la forma clásica “© año, autor”.
- Cualquier modificación debe llevar avisos visibles, indicando la autoría (cl. 2) y que se ha cambiado el programa original y la fecha del cambio (cl. 2a). Si el programa modificado es interactivo, debe procurar que el programa muestre el anuncio de *copyright* al iniciarse su ejecución en uso interactivo (cl. 2c). Así, queda protegido el autor original en caso de degradación de calidad o no funcionamiento del programa modificado.

**Aceptación y revocación:** no se necesita aceptar la GPL para usar el programa. Sin embargo, para modificarlo y distribuirlo, esta aceptación es necesaria. Ya hemos visto que en ausencia de una licencia otorgada por el autor –y, por lo tanto, en la ausencia de esta aceptación de la licencia por parte del usuario– las acciones de modificación y distribución son prohibidas por las leyes de derechos de autor.

La GPL establece ciertas normas para su aceptación. La cláusula 5:

- advierte que el usuario no está obligado a aceptar la licencia para el simple uso del programa;
- por el contrario, obliga al usuario a aceptar la licencia si quiere modificar o distribuir el software;

- entiende que, por el hecho de modificar o distribuir el software, el usuario está indicando que acepta la GPL y todos sus términos y condiciones.

Asimismo, en la cláusula 4, como término resolutorio, la licencia avisa al licenciataria de la revocación de los derechos del usuario en caso de violación de los términos de la licencia, como por ejemplo, la imposición de términos más restrictivos sobre cualquier distribución del programa o de obras derivadas.

**Versiones:** la versión actual de la licencia es la 2.0. La licencia permite a los autores-licenciantes referirse a nuevas versiones de la misma (se habla de una 3.0, comentada más adelante) agregando “cualquier versión posterior” (cl. 9). Esta flexibilidad permite que sus programas puedan mantener la compatibilidad con futuros programas bajo una versión posterior de la GPL. En este caso, los licenciarios pueden elegir la versión aplicable. L Torvald, por ejemplo, ha excluido las versiones posteriores para el *kernel* de Linux: se queda con la versión 2.0.

**Garantías:** las cl. 11 y 12 aclaran que no se ofrece ninguna garantía sobre el funcionamiento correcto del software cubierto por la licencia y niegan cualquier responsabilidad por daños y perjuicios. Sin embargo, hemos visto en la unidad 6 que la validez de estas cláusulas es dudosa (en jurisdicciones distintas a la de EE.UU. y aun en EE.UU. en algunas circunstancias) bajo el derecho de protección del consumidor y la prohibición de las cláusulas abusivas.

**Derecho aplicable:** la licencia GPL no incluye ninguna cláusula indicando el derecho aplicable o los tribunales competentes para entender un conflicto referente a ella. Por lo tanto, se aplicará el derecho relevante en los tribunales correspondientes bajo los principios del derecho internacional privado. Un consumidor, por ejemplo, puede elegir el derecho y los tribunales de su domicilio.

**Patentes:** el último párrafo del preámbulo resalta los peligros que presentan las patentes para el software libre. Sin embargo, no hay ninguna cláusula que restrinja las patentes eventuales sobre código GPL u obligue a licenciarlas a favor de los demás usuarios. La licencia obliga a distribuir el programa y cualquier obra derivada bajo

términos iguales a los de la GPL (cl. 2b) o con condiciones menos restrictivas (cl. 6 –siempre que se mantenga el *copyleft*). Como consecuencia lógica, cualquier licenciataria que obtenga una patente sobre una obra de software bajo GPL deberá ofrecer una licencia que permita el uso libre conforme a la GPL por parte de todos los destinatarios posteriores.

**Otros pactos:** otros conceptos cubiertos por la licencia de interés son los siguientes:

- Si, como consecuencia de una resolución judicial o de alegaciones de derechos de terceros, se imponen condiciones al licenciataria que contradicen las de la licencia (por ejemplo, las obligaciones de *copyleft*), dichas condiciones no lo excusan de las de la GPL. Por lo tanto, si no se puede distribuir el programa u obra derivada de una manera que satisfaga ambas obligaciones, no se debe distribuir (cl. 7). Esto, por ejemplo, restringirá la distribución del software en un país donde un tercero tenga una patente válida sobre un proceso incorporado en el código.
- Asimismo, si se declara inválido cualquier término de la licencia, el resto de la misma ha de cumplirse (manteniendo el espíritu de la GPL) (cl. 7).
- El autor original del programa puede imponer límites geográficos adicionales a la libre distribución si fuera necesario, para cumplir con los derechos de propiedad intelectual e industrial de terceros en otros países (cl. 8).

### Obras derivadas y el efecto *copyleft*

Un tema importante por aclarar es la cuestión de las obras derivadas y la aplicación de la licencia a las mismas por el *copyleft*. Este tema ha suscitado mucha polémica en el mundo del software libre y del software en general. Es un concepto clave para entender la licencia GPL porque define el alcance de la cláusula de *copyleft*, que es lo que más lo distingue de otras licencias libres. Ya hemos dicho varias veces que no se puede “privatizar” un software bajo la GPL, ni las obras derivadas de éste. Por tanto, algunos desarrolladores dudan

de incorporar o vincular demasiado íntimamente sus trabajos con un programa *copyleft*, porque temen verlos caer bajo la licencia GPL, en circunstancias en que no pueden o no quieren permitirlo (por ejemplo, un desarrollo propietario o bajo una licencia libre diferente).



Las obras siguientes comentan este tema (en bibliografía):

D. Ravicher. *On Open Source Legal Issues*.

M. Assay. *A Funny Thing Happened...*

L. Rosen. *The Unreasonable Fear of Infection*.



¿En qué consiste una obra derivada u obra basada en el programa, según los autores de la licencia? La definición citada se refiere a la definición bajo el derecho de *copyright*: es una obra que contenga el programa o una porción del mismo.

Pero la palabra *contener* en el ámbito de la programación deja lugar a dudas. La licencia y los comentarios (FAQ) sobre la misma nos aportan cierta ayuda.



En la última parte de la cláusula 2, la GPL incluye un párrafo que indica que el *copyleft* se aplica a una obra derivada como “un todo”. Si partes identificables de una obra de modificación pueden ser razonablemente consideradas obras independientes y separadas por sí mismos, la licencia no se aplica a dichas partes. Esto se refiere a las particularidades de la arquitectura y el diseño de los programas.

Los componentes de software pueden interrelacionarse de distintas maneras, por ejemplo:

- en la programación, por llamadas, vínculos o enlaces de distintos tipos;



- al final del desarrollo, en la compilación del programa para crear el ejecutable; o
- cuando se interpreta en el momento de la ejecución.

Cada una de estas interacciones puede tener efectos legales diferentes.

La cuestión debatida, es decir: si una arquitectura o diseño supone o no la creación de una obra derivada, es complicada por la evolución de los métodos de programación (estructurado o por objetos) y los lenguajes informáticos (C, C++, Visual Basic, Java, PHP, etc.), muchos de los cuales no existían en el momento de redacción de la licencia. En este momento, podemos decir lo siguiente:

- Si, cuando se compila un desarrollo nuevo basado en un software bajo GPL, el ejecutable final incluye elementos del programa original (será el caso de componentes con vínculos estáticos entre ellos) entonces, no hay ninguna duda de que las modificaciones no se pueden considerar separables y, en consecuencia, la obra completa tendrá que distribuirse bajo la GPL.
- Si el programa original GPL y el desarrollo particular pueden co-existir de manera separada (incluso cuando se encuentren sobre un mismo soporte) y el desarrollo particular llama al programa GPL en tiempo de ejecución (este es el caso de un vínculo dinámico), entonces, desafortunadamente, la situación no es tan clara. Hay dos opiniones:
  - R. Stallman, autor de la licencia, ha insistido muchas veces en que considera que los programas con vínculos dinámicos a código GPL sí que quedan afectados por la GPL. Precisamente, sostiene que por ello se inventó la licencia LGPL, para permitir este tipos de vínculos sin aplicación del *copyleft* de la GPL.
  - En la práctica, y con el tiempo, se ha reconocido mayoritariamente que una obra que tenga vínculos dinámicos con código bajo GPL no se ve afectada por la GPL.

**Nota**

Las FAQ están en [www.gnu.org/licenses/gpl-faq.html](http://www.gnu.org/licenses/gpl-faq.html).

Obsérvese que la mera reunión o agregación de una obra (separable y no basada en un programa bajo GPL) en un mismo soporte o medio de almacenamiento con código bajo GPL, por ejemplo, para su distribución, no implica que esta otra obra deba ser distribuida bajo la GPL.

Entre las **preguntas más frecuentes** sobre la licencia GPL, hay una serie de cuestiones que exponen casos de modificaciones, vínculos y llamadas a código bajo GPL que la FSF “resuelve” en caso de dudas, pues ofrece su interpretación de la licencia y del derecho. Por ejemplo, aclara que un programa nuevo compilado por un *compiler* bajo GPL no tendría que distribuirse bajo esta licencia, excepto si el ejecutable resultante después de la compilación incorporara elementos del compilador libre u otro programa bajo GPL.

Pero el tema no está resuelto del todo, sobre todo porque no todos los programas utilizan conceptos de llamadas, vínculos estáticos o dinámicos o la compilación. Podemos citar, por ejemplo, las clases de Java, que se enlazan o se “interpretan” en tiempo de ejecución.



Al final, la FSF deja a juicio de los creadores de modificaciones y obras derivadas considerar si las mismas caen bajo la GPL.

La prudencia dicta que se debe evaluar los riesgos legales respecto a un desarrollo y arquitectura en particular, considerando el diseño y las consecuencias potenciales de caer bajo la GPL.

**Nota**

Dada la extrema importancia de este tema, será objeto de un caso práctico de estudio en la unidad 9.



Linus Torvald ha incluido expresamente en la licencia GPL que cubre el núcleo Linux del SO GNU/Linux una cláusula extra, al efecto de que él, como autor licenciante, no considera que los programas que tienen vínculos dinámicos al núcleo estén afectados por *copyleft*.

Las aplicaciones de usuarios y otros elementos no centrales de un sistema operativo como los controladores de dispositivos (*drivers*) interactúan de manera dinámica con los componentes y módulos del núcleo del sistema. Por ello, las aplicaciones y los *drivers* son específicos para una plataforma u otra. Hay una posibilidad de que esta interacción con un sistema operativo bajo la GPL afecte a estos programas y *drivers*. Sin esta aclaración, casi cualquier programa que se ejecute sobre GNU/Linux y llame a sus bibliotecas centrales podría considerarse, bajo la interpretación más estricta de la licencia, afectada por la GPL. Esto reduciría el uso y la difusión de GNU/Linux como sistema operativo a un entorno de programas compatibles con la GPL.

### Otros comentarios

Ya hemos aclarado algunos mitos y dudas legales en relación con las licencias de software libre en general. Éstos se refieren también a la licencia GPL: el respeto del marco legal de derechos de propiedad intelectual, la posibilidad de uso comercial, las garantías que no se ofrecen, etc. Aquí queremos comentar algunos problemas potenciales que pueden surgir en la aplicación de la licencia GPL en particular y examinar otros temas como la persistencia de la licencia, traducciones, su validez y la “compatibilidad” con ella.

**Persistencia:** la GPL es una licencia persistente, en la medida que obliga a realizar cualquier distribución posterior del programa (y cualquier obra basada en él) bajo la misma licencia. La persistencia de los términos de la licencia, es decir la continuidad de la aplicación de la licencia al programa, tiene dos aspectos:

- Relativo al **programa inicial**: cualquier destinatario de una distribución del programa inicial debe recibirlo bajo los mismos términos que el licenciatario original (cl. 6) y ello aun si éste está en violación de la licencia. Por ejemplo, si el licenciatario original redistribuye un “programa GPL” bajo términos más restrictivos que la GPL, el licenciatario pierde sus derechos, pero el destinatario

de la distribución recibe el programa bajo los términos originales de la GPL.

- Relativo a **obras derivadas** (u “obras basadas en el programa”): los mismos términos se deberán aplicar a las obras derivadas del programa inicial distribuidas a terceros (cl. 2.b). El receptor de una obra derivada de un programa GPL siempre deberá recibir software bajo GPL.

En la práctica, esto significa que, si un desarrollador incluye código bajo GPL en su desarrollo propietario y lo distribuye a terceros bajo una licencia no-GPL, primero, este desarrollador infringe la licencia y pierde sus derechos de modificación y distribución y, segundo, el cliente del desarrollador (u otro destinatario) recibirá por lo menos el código original bajo GPL (y probablemente los demás códigos también). Cualquier destinatario tendrá el derecho de exigir el código fuente y de copiar, modificar y distribuir su programa (también bajo los términos de la GPL).

**Traducciones:** la GPL no tiene traducciones oficiales. Es decir, la versión original en inglés será la que determine los términos de distribución cuando se aplica la GPL original a una obra. Pero hay traducciones oficiosas indicadas en las páginas de la FSF. Ésta no aprueba las traducciones como oficialmente (jurídicamente) válidas. Si hubiera un error de traducción, los resultados podrían ser no sólo desagradables, sino espantosos para la comunidad del software libre. Habría versiones y modificaciones de software “casi-GPL” (en su matiz extranjero) mezcladas con programas realmente GPL (en su versión en inglés). Observad que si un autor aplica una licencia GPL traducida a su programa, entonces será la traducción de la licencia la que prevalecerá, no la GPL original en inglés (salvo indicación contraria).

**Validez de la licencia como contrato:** según se ha comentado, en muchos países como España, las condiciones de una licencia son vinculantes si el licenciataria ha tenido la oportunidad de conocer las mismas antes de aceptarlas y ha expresado su consentimiento. El mecanismo de la GPL intenta asegurarse de ello:

- **Términos:** la obligación expuesta en las cláusulas 1 y 2 obliga a acompañar cualquier distribución del programa con una copia de

#### Nota

Podéis ver las condiciones de una licencia en la unidad 5.

la licencia. Por lo tanto, cualquier usuario tendrá ocasión para ver sus condiciones. Sin embargo, habría argumentos, que hemos comentado en la unidad 5, para sostener que un usuario de código bajo la GPL puede no haber visto las mismas antes de empezar a trabajar con él y, por lo tanto, no tendría que respetar las condiciones de la licencia.

- **Aceptación:** ya lo hemos comentado. Cualquier copia, modificación o distribución significaría el consentimiento implícito del usuario.

Se quiere resaltar aquí que este tema no será de gran importancia en la práctica:

- La GPL no intenta imponer obligaciones relativas al uso, para vincular a los simples usuarios. Se concentra en otorgarles libertades, lo que no necesitaría el consentimiento de los mismos.
- La licencia tampoco intenta otorgar al autor mayores derechos que los ya otorgados por las leyes de derechos de autor. Estos derechos suplementarios necesitarían el consentimiento expreso e informado del vinculado.
- Las obligaciones impuestas relativas a la modificación y distribución necesitarían el consentimiento del licenciatario. Se argumenta que cualquier persona que realice estos actos habrá visto las condiciones y el acto mismo constituiría la expresión del consentimiento.

En esto, el **derecho anglosajón** puede ser más permisivo, diferenciando una licencia de *copyright* (una autorización unilateral) de un contrato (un acuerdo mutuo). Aun si el licenciatario de código bajo GPL no acepta expresamente los términos como “contrato”, seguirá vinculado por las condiciones de uso como “licencia”. Éstas son válidas siempre que se queden dentro del monopolio reservado a los autores por el derecho de *copyright*. Los ámbitos donde la GPL impone obligaciones, relativas a la modificación y la distribución del código, están bien dentro de los derechos reservados por el *copyright*. En consecuencia, se argumenta que ellas son válidas y vinculantes sin la aceptación explícita por el licenciatario.



El movimiento de software libre considera que:

“[la ejecución de un software libre] es un derecho del cual todos los usuarios deben beneficiarse. Casi todos los que usan a diario software bajo la licencia GPL no necesitan licencia alguna, ni aceptan ninguna. La GPL impone obligaciones únicamente si uno quiere distribuir software derivado de código bajo GPL y solamente requiere el consentimiento cuando ocurra esta redistribución. Y, como no se puede redistribuir sin licencia, podemos suponer con seguridad que cualquier redistribuidor tiene la intención de aceptar la licencia. Después de todo, la GPL requiere que cada copia de software bajo GPL incluya la licencia, por lo tanto todos están informados.”

E. Moglen

**Compatibilidad con la licencia GPL:** se trata de la compatibilidad legal.



Un programa legalmente compatible con código bajo GPL es un programa que se distribuye bajo términos que son compatibles con los términos de esta licencia.

No pueden ser más restrictivos, como cualquier licencia propietaria, pero sí que pueden ser más permisivos, como en la licencia BSD moderna o en la LGPL, que estudiaremos a continuación. La compatibilidad con la licencia tiene la doble ventaja de facilitar la integración de componentes libres en distribuciones y plataformas más complejas e integradas y de asegurar que su código puede integrarse sin miedo con el 75% de los programas de software libre disponibles en Internet, que están bajo la GPL.

#### Ejemplo

Algunos ejemplos de incompatibilidad con la GPL son:

- La cláusula de la licencia BSD original y la licencia Apache, que obligaba a incluir una mención de los

autores originales en cualquier publicidad o material promocional del programa.

- Las cláusulas de reservas de derecho de la Netscape Public Licence, que permiten a Netscape beneficiarse de las modificaciones de terceros al Navigator e incorporarlas en nuevos productos de Netscape.
- La obligación de obtener una licencia de desarrollador para integrar elementos de Qt en aplicaciones que no son Windows X System (en la licencia Qt).

#### f) Conclusiones

Después de esta presentación de la GPL nos queda, para finalizar este análisis, resumir cuáles son algunas de las **consecuencias prácticas más importantes** de la licencia. Se sintetizan a continuación:

- Toda persona puede ejecutar un programa bajo GPL, para fines personales o comerciales.
- Toda persona puede copiar, modificar y distribuir a cualquier persona copias y modificaciones del programa, siempre que ofrezca acceso al código fuente.
- No se pueden cambiar las condiciones de la licencia GPL sobre el programa, ni sobre las obras derivadas: siempre será la GPL.
- Hay que distribuir el código fuente sin canon o regalía para el destinatario. Por ejemplo, Red Hat no cobra por la licencia de GNU/Linux, sino por los servicios y garantías ofrecidos y los costes del soporte físico (el CD-ROM).
- Con respecto a la GPL, se puede incorporar un programa bajo la misma con un programa bajo licencia propietaria, siempre que no se distribuya el programa resultante. Las obligaciones de *copyleft* surgen en el momento de la distribución. Sin embargo, es más que probable que esta mezcla de códigos sea una viola-

**Lectura complementaria**

Para más comentarios y explicaciones sobre la GPL, consultad el apartado "GPL" en bibliografía.

ción de la licencia propietaria, que normalmente prohibiría su modificación.

- No se puede aplicar o pedir una patente sobre software GPL u obras derivadas de él sin otorgar una licencia de libre uso para todos, sino sus condiciones de distribución serían más restrictivas que la licencia GPL original.
- El autor del programa bajo GPL puede infringir sus propios términos y mezclarlo con otro código no compatible con la GPL.
- Dicho autor puede licenciar su código de varias maneras (el licenciamiento doble... o más), en "violación" de la licencia.



En conclusión, en lo relativo a la aplicación de la **cláusula de copyleft**, se puede decir que los que quieren utilizar código bajo GPL, tienen una elección explícita: o aceptan sus términos y condiciones de modificación y distribución o deben encontrar otra solución para evitar usar dicho código.

Asimismo, en la práctica, un subterfugio que evite las obligaciones legales de la licencia por una razón u otra (como por ejemplo, la no-aplicación de la licencia bajo el derecho de un país) quizás no tendría consecuencias legales, pero sí que sería sancionado por la comunidad de software libre y sería probablemente excluido de cualquier desarrollo comunitario, por no mencionar la publicidad negativa que pueda surgir alrededor del programa y su distribuidor.

## 2) La Licencia Pública General Menor o de bibliotecas GNU (la LGPL)

En este apartado consideraremos la segunda licencia de la Free Software Foundation: la Licencia Pública General Menor (o de bibliotecas) GNU. Inicialmente, esta licencia se llamó Library GPL, puesto que fue diseñada expresamente para aplicarse a bibliotecas informáticas. Luego, la FSF cambió su nombre a Lesser GPL (GPL menor), porque consideraba que garantizaba menos libertad



que su hermana mayor, la GPL. La versión actual es la 2.1, de febrero de 1999.

Ya hemos comentado que cuando un programa se enlaza con un componente de software, ya sea estáticamente o mediante un componente o API compartida de manera dinámica, la combinación se considera una obra basada en el software original o derivada de éste. Si el software está bajo la GPL, este enlace obliga a distribuir todo el programa final bajo la GPL. La LGPL se creó para permitir que se enlacen algunos componentes de software específicos –las bibliotecas– con programas no libres, sin afectar al programa resultante. Por tanto, una biblioteca bajo LGPL ofrece cierta comodidad o certeza para los desarrolladores de aplicaciones propietarias que quieren vincular sus programas con componentes bajo licencias libres, pero temen el efecto *copyleft* de la GPL.

Actualmente, la FSF no recomienda el uso de la LGPL, excepto por razones estratégicas: el uso de la LGPL permite la distribución y el uso más amplio de su código y, por lo tanto, favorece establecer un componente –una biblioteca, un módulo de programa, etc.– como un estándar en el sector. Sin embargo, la LGPL no favorece el desarrollo de aplicaciones libres, un objetivo fundamental para la FSF y, por ello, no recibe su plena aprobación.

La LGPL deriva de la GPL y la mayoría de sus términos son similares a los de aquella.

#### a) Definiciones útiles

Como la GPL, la LGPL define programa y código fuente e incluye tres definiciones nuevas:

- **Biblioteca:** se trata de un conjunto de componentes de software destinados a enlazarse con programas (que usan las funciones incorporadas en las bibliotecas) para crear un ejecutable.
- **Obra basada en una biblioteca:** recoge la definición de programa en la GPL: significa la biblioteca original o cualquier obra derivada de ella según la definición del derecho de *copyright*, es decir, una obra que la contenga o contenga parte de ella.

- **Obra que usa una biblioteca:** es una obra separada que no contiene ninguna parte u obra derivada de la biblioteca, sino que se destina a ejecutarse con la biblioteca por medio de la compilación o enlaces.

Hay que resaltar, como lo hace también la licencia, que el resultado completo de la compilación de la “Obra que usa una biblioteca” más la “biblioteca bajo LGPL”, se considera una obra derivada de la biblioteca y, por lo tanto, sujeta a la licencia LGPL. Es para estas compilaciones que la LGPL permite una distribución particular.

#### a) Componentes esenciales

**Derechos otorgados y actos permitidos:** las libertades fundamentales del software libre se mantienen.

La LGPL otorga los derechos de:

- Copiar y distribuir copias de la biblioteca en las mismas condiciones que la GPL (cl. 1).
- Modificar la biblioteca o una porción de ella, formando una obra basada en la misma y distribuirla, siempre que:
  - la obra modificada sea una biblioteca de software;
  - los ficheros modificados indiquen en qué fecha se modificaron;
  - la obra modificada se licencie bajo la LGPL; y
  - si una funcionalidad en la biblioteca modificada hace referencia a una función o tabla de datos que es provista por un programa que usa esta facilidad, la misma debe mantenerse operativa aunque el programa no lo provea alguna vez (cl. 2).
- Convertir la licencia LGPL aplicada a su biblioteca a la licencia GPL, en cualquier momento (no se puede volver atrás) (cl. 3).

- Distribuir sin restricción un ejecutable que consista en la compilación de (a) Obras que usan la biblioteca y (b) la misma biblioteca (cl. 6). Esta es la excepción a la cláusula de *copyleft* habitual de la GPL. Se debe permitir al destinatario modificar el programa para su uso particular y para realizar operaciones de ingeniería inversa para la corrección de errores.

En cuanto a restricciones, condiciones y actos prohibidos, las obligaciones de la GPL también se mantienen:

- Cualquier distribución de la biblioteca o de una obra derivada de ella debe acompañarse de los avisos de autoría, cualquier cambio y las fechas de cambio, la notificación de la falta de provisión de garantía y una copia de la licencia (cl. 1).
- Se debe acompañar cualquier distribución de la biblioteca (independiente o compilada) con su código fuente, o proveer u ofrecer acceso al mismo a terceros, o utilizar un sistema de interfaz abierto de enlace con la biblioteca (si ésta ya existe en el equipo del usuario, por ejemplo en el sistema operativo y por tanto no hace falta distribuirla) (cl. 4, cl. 6).
- Se debe distribuir cualquier modificación directa de la biblioteca bajo la misma licencia LGPL (el *copyleft* para modificaciones –cl. 2).

#### **b) Otros aspectos relevantes de la LGPL**

Las demás disposiciones de la licencia son iguales o similares a la GPL, en cuanto a la autoría, la aceptación y la revocación de la licencia, su persistencia, los derechos de aplicar restricciones geográficas o por razones de propiedad intelectual vigente (patentes o derechos de autor en el país destino), la ausencia de garantías y la negación de cualquier responsabilidad.

Hay que observar también que en las cláusulas 5 y 6 se da un tratamiento bastante minucioso de diferentes formas de enlazar con las bibliotecas, para el cual referimos al lector a la licencia misma.

Por su vocabulario, la LGPL está destinada al uso para bibliotecas. Pero no está restringida a las mismas puesto que hay otros progra-

mas que se han distribuido bajo esta licencia. Los autores de software son libres de elegir la licencia que quieran, sea el que fuere su programa.

### c) Comentarios

Como comentario práctico, se puede decir que cualquier código bajo la licencia LGPL reúne las mismas características legales que la GPL comentada antes y, además, permite enlaces dinámicos y potencialmente estáticos (según la arquitectura del programa) entre un software propietario y las bibliotecas libres en cuestión. Por lo tanto, dentro de los límites de las cuestiones técnicas del tipo de enlace, se puede combinar, integrar y distribuir estas bibliotecas en software propietario.

#### Ejemplo

Un ejemplo de este tipo de software es la biblioteca de C (*libgcc*) que se distribuye con Linux, que se puede usar para desarrollar programas propietarios que se ejecutan sobre Linux.

El tema del uso estratégico de la licencia es interesante. Aunque la LGPL provea mayor certeza y flexibilidad para los desarrolladores en general y para los de software propietario en particular, ofrece menos ventajas para los desarrolladores de software libre, porque permite la privatización parcial de las bibliotecas en programas propietarios.

Sin embargo si se quiere convertir una biblioteca en un estándar, la LGPL permite el uso extensivo de la misma y favorece su difusión en desarrollos libres y propietarios y, al mismo tiempo, mantiene su propia libertad.

Se ha sugerido que la LGPL se usa generalmente cuando una biblioteca libre hace la misma tarea que otras no libres: en este caso, no se ganaría mucho si la biblioteca estuviera cubierta por la GPL, porque los desarrolladores propietarios podrían usar la alternativa.



“El uso de la LGPL para la biblioteca C o para cualquier otra biblioteca, es un tema de estrategia. La biblioteca C hace un trabajo genérico; todo sistema propietario o compilador viene con una biblioteca C. Por lo tanto, el hacer que nuestra biblioteca estuviera sólo disponible para el software libre, no le hubiera dado al software libre ninguna ventaja: sólo hubiera desalentado el uso de nuestra biblioteca. No hay ninguna razón ética para permitir aplicaciones propietarias en un sistema GNU, pero estratégicamente, parece que si no se permite, ello contribuirá más a desalentar el uso del sistema GNU que a alentar el desarrollo de aplicaciones libres.”

R. Stallman

Observad también que la licencia LGPL es compatible con la GPL.

### 3) Otras licencias con *copyleft*

En este último apartado sobre las licencias con *copyleft*, presentamos una tabla que detalla varias licencias de software libre que incluyen obligaciones de tipo *copyleft*. Algunas son compatibles con la GPL, por lo tanto, se puede mezclar su código con código bajo GPL y distribuir el resultado sin problema. Otros, por las obligaciones adicionales que imponen, no son compatibles con ella. Las resumimos en las tablas 4 y 5.

**Tabla 4.** Licencias compatibles con la GPL

eCos license versión 2.0	Es una licencia de la FSF, sobre el sistema de <i>Embedded Configurable Operating System</i> . Básicamente, consiste en la GPL más una excepción que permite enlazar el programa con otros programas que no están bajo la GPL y con efectos muy similares a la LGPL. Aunque se integre por compilación o enlace con un programa propietario distribuido en binario, se debe proveer o poner a disposición del usuario el código fuente de eCos.
Sleepycat Software Product License (Berkeley Database)	Es una licencia que se aplica, sobre todo, a un motor de base de datos de la empresa Sleepycat (antiguamente Berkeley Database). Sigue el modelo simple de la licencia BSD comentada a continuación y agrega una obligación de distribuir o poner a disposición el código fuente del software y de cualquier otro programa que utilice el software (una obra derivada). Asimismo, dicho programa debe ser libremente redistribuible bajo términos razonables (el <i>copyleft</i> ).

Tabla 5. Licencias incompatibles con la GPL

Licencia Aferro	<p>Aferro es un software para gestionar y extender las comunidades virtuales con funcionalidades de <i>rating</i> y de comercio electrónico. La licencia es una variación de la GPL, redactada con la ayuda de la FSF. La licencia cubre el caso de la arquitectura de programas distribuidos en redes o servicios enlazados vía servicios web. En este caso, el usuario licenciatario no recibe el programa como distribución de software, sino como un servicio a través de la web y puede ofrecer el mismo servicio a terceros, evitando las obligaciones de <i>copyleft</i> de la cl. 2b. La licencia Aferro agrega a la GPL una cláusula "2.d" que, si en el caso de un servicio ofrecido por red el programa original tiene una función para proveer el código fuente también vía web, el licenciatario no puede eliminar esa función y debe ofrecer acceso por web al código fuente de la obra derivada.</p> <p>Es incompatible con la GPL, porque esta adición crea una obligación más restrictiva que la GPL. Se ha criticado esta licencia por restringir las comunicaciones de red al protocolo HTTP, cuando en el futuro puede haber otras formas de comunicación. La OSI también critica este aspecto, porque el acceso al código no debe estar vinculado a una tecnología en particular (directriz 10).</p>
La licencia OpenSSL/SSLeay	<p>Se aplica a programas de seguridad SSL. Es una combinación de las licencias Open SSL y SSLeay. Está modelada sobre la licencia BSD comentada a continuación y agrega al final de la licencia SSLeay una cláusula de <i>copyleft</i>, en la cual obliga a cualquier obra derivada a distribuirse bajo los mismos términos. Se prohíbe expresamente mezclar este código con código bajo la GPL. También es incompatible con la GPL porque tiene una cláusula con respecto a la publicidad y la atribución a los autores (que proviene de la versión anterior de la BSD y Apache).</p>
IBM Public License v. 1.0 Common Public License v. 1.0	<p>Estas licencias son instrumentos nuevos desarrollados por IBM, con un formato diferente de la GPL y la BSD, los dos modelos predominantes.</p> <ul style="list-style-type: none"> <li>Definen "contribuciones" como modificaciones y componentes agregados al programa, que son obras derivadas de él.</li> <li>La redistribución del código fuente se debe hacer bajo una licencia compatible con la original e indicar cualquier diferencia (cl. 3).</li> <li>Se puede redistribuir el programa (y obras derivadas) en binario, solamente si el re-distribuidor provee un mecanismo para que el destinatario pueda recibir el código fuente.</li> <li>Los redistribuidores comerciales deben indemnizar a cualquier otro autor contra demandas de terceros que surjan de la distribución comercial, excepto en lo que se refiere a la propiedad intelectual.</li> <li>Cada contribuidor al código otorga una licencia de patente "libre" a todos los destinatarios y usuarios del código (incluso los otros autores) (cl. 2).</li> </ul> <p>La licencia es incompatible con la GPL, por la obligación de licenciar cualquier patente y de compensar a coautores contra las demandas de usuarios comerciales.</p>



Hay varios análisis de licencias de software libre accesible in Internet. Podéis consultar:

R. Brooks. *Open Source Licenses Overview*. Electronic Freedom Foundation: Guide to Licenses.

Stig Hackv  n. *A Quick Survey of Open Source Licences* (en bibliograf  a).

### 7.3.2. Las licencias sin *copyleft* robusto, compatibles con la GPL

En este apartado presentamos otras licencias libres muy utilizadas en la comunidad de desarrollo libre, especialmente la licencia BSD

que ha sido un modelo para muchas otras licencias. La mayoría se encuentran “al otro extremo” de la GPL en el espectro de licencias libres, por no contener obligaciones de *copyleft* y permitir la privatización de obras derivadas.

### 1) La licencia *Berkeley Software Distribution* (BSD) y similares

La licencia *Berkeley Software Distribution* (BSD) es quizá el modelo más simple de todas las licencias libres. Surge de las distribuciones de versiones de UNIX de la Universidad de California, Berkeley, en los años 1970 y 1980, en las raíces del movimiento de software libre. El principio detrás de la licencia es que el código es fruto de las investigaciones y los trabajos universitarios financiados por el gobierno de los EE.UU. (y los impuestos del pueblo americano), por lo tanto, debe ser de acceso libre, protegiendo lo que llamaríamos aquí los “derechos morales” de los autores por la simple obligación de mantener los avisos de autoría (*copyright notice*).

Como “abuela” de las licencias libres, se ha utilizado la BSD o variantes de ella para muchos programas libres actuales, que comentaremos en seguida.

#### a) Elementos esenciales

La BSD transfiere al usuario (licenciataria) todos los derechos transferibles bajo el derecho de propiedad intelectual:

- **Derechos otorgados:** permite el uso, modificación, copia y redistribución sin restricción del software bajo BSD, en formato de código objeto (binario) o fuente.
- **Obligaciones impuestas:** la distribución en forma de código fuente ha de acompañarse del aviso de *copyright*, la lista de condiciones y la negación de cualquier garantía y responsabilidad. Las redistribuciones en binario deben reproducir lo mismo en la documentación. No se puede usar el nombre del autor o de los contribuyentes para fines de promoción de obras derivadas sin su permiso.
- **Otros términos:** no se otorga ninguna garantía sobre el correcto funcionamiento del programa y se niega cualquier responsabilidad.

Por lo tanto, se puede realizar casi cualquier acto con código bajo BSD, siempre que se respete la mención de autoría del programa inicial. Si se modifica el software, se pueden eliminar estas indicaciones de autoría e imponer cualquier licencia nueva. Además, no hace falta proveer al usuario final el código fuente. Por lo tanto, no es una licencia persistente.

#### b) Otros aspectos relevantes de la BSD y comentarios

Las versiones anteriores de la licencia tenían una obligación de atribuir cada componente a sus autores originales en cualquier publicación o material de promoción del programa u obra derivada. Esta obligación causaba ciertas molestias, porque había que incluir atribuciones de autoría extensivas en toda la documentación y en el código fuente, relativo a cada autor que agregaba su nombre en una licencia. En un programa con cientos de contribuyentes, era muy difícil cumplir con esta obligación. Además, hacía que código bajo BSD fuera incompatible con código bajo la GPL. En julio de 1999, se eliminó esta obligación de la licencia BSD. De todos modos, actualmente, hay que fijarse bien en la versión de la licencia aplicada a código bajo BSD, a no ser que haya una versión anterior, y entonces asegurarse el correcto cumplimiento de los términos.

La BSD es una licencia muy corta, fácil de entender. Además, es la más permisiva de las licencias libres, en el sentido que permite realizar casi cualquier acto con el programa en cuestión. No contiene las obligaciones de *copyleft robusto* y permite incorporar y combinar el software bajo BSD en cualquier tipo de obra, libre o propietaria. Por ejemplo, se dice que hay componentes de software BSD en los sistemas operativos Windows NT y OS-X de Macintosh. La FSF discute la afirmación de que la BSD es la licencia más libre porque, aunque otorgue mayor libertad a los desarrolladores, no garantiza tanta libertad a los usuarios finales, ya que dichos desarrolladores pueden privatizar cualquier obra derivada.

#### Nota

Para un comentario interesante, podéis consultar:  
E. Leibovitch. *License to FUD (comparing GPL and BSD)* (en bibliografía).



Por la libertad otorgada a los desarrolladores para mezclar código bajo la BSD con código propietario, la licencia BSD es más favorable para el mundo de los ne-



gocios y los desarrollos comerciales y propietarios. Asimismo, igual que la LGPL, permite una gran difusión del código y su uso como referencia o estándar (para protocolos, servicios, bibliotecas y hasta sistemas operativos completos como el UNIX BSD).

Sin embargo, también permite lo que se llama *bifurcación de código*, el *forking*, porque cualquiera puede adaptar, modificar y extender el núcleo del programa y crear una versión “similar pero suficientemente diferente”. Esto se nota, por ejemplo, en la multiplicación de sistemas operativos de tipo BSD, como el OpenBSD, la FreeBSD y la NetBSD.

Cualquier código bajo la licencia BSD es compatible con código GPL, pero no viceversa: se puede incorporar código BSD en un programa bajo la GPL (con el resultado de una obra bajo GPL) pero no se puede incorporar código GPL en un software BSD, porque afectará al resto del programa bajo la licencia BSD.

### c) Otras licencias similares a la BSD

La BSD ha sido modelo de muchas licencias parecidas, entre las cuales citamos:

- Las licencias MIT.
- Las licencias de la familia X: X, XFree86, XOpen, X11.
- Licencias de la familia BSD: OpenBSD, Free BSD, NetBSD.
- La licencia Apache (que se comenta a continuación).
- Cryptix, Python, W3C Software notice, Zope Public License (ZPL), LDAP public license, Phorum, etc.
- Las licencias OpenSSL y Sleepycat siguen el modelo simplificado de la licencia BSD, pero incluyen cláusulas de *copyleft* (ya comentadas en el apartado anterior).

Las licencias X y MIT son similares a la BSD. Literalmente, desarrollan más los usos permitidos: el uso, la copia, la modificación, la fusión, la publicación, la distribución y/o la venta, y no distinguen entre distribuciones de código fuente y objeto. Al igual que la BSD, permiten cualquier uso privado o comercial sin restricciones del programa, siempre que en el software y la documentación de soporte se mantenga el aviso de *copyright* y las condiciones de la licencia. Incluyen la negación de garantía y de responsabilidad y la obligación de obtener el permiso para el uso del nombre del autor con fines promocionales.

Algunas de las licencias mencionadas llevan cláusulas adicionales que imponen mayores restricciones, y que comentaremos en la tabla siguiente.

#### Ejemplo

Por ejemplo, la licencia Apache incorpora obligaciones adicionales de publicidad que la hacen incompatible con la GPL (por ello la comentamos más adelante), mientras que la Licencia Pública Zope incluye una restricción (innecesaria en nuestra opinión) sobre los usos de la marca.

## 2) Otras licencias abiertas y libres compatibles con la GPL

Es suficiente consultar la página de licencias de la OSI o la FSF para ver que hay una multitud de licencias libres, varias de ellas compatibles con la GPL por no tener restricciones mayores que las permitidas. A continuación, seguiremos nuestra tabla analítica anterior con un comentario de algunas licencias más comunes, compatibles con la GPL.

**Tabla 6.** Otras licencias abiertas y libres compatibles con la GPL

Licencia	Comentario
Zope Public License (versión 2.0)	Sigue el modelo BSD e incluye una cláusula que prohíbe expresamente el uso de las marcas registradas o no ( <i>servicemarks</i> ) de Zope Corporation, excepto bajo un acuerdo paralelo específico. También se debe avisar de los cambios de los ficheros con la fecha de modificación.
Open LDAP License (versión 2.7)	Sigue el modelo BSD e incluye una cláusula que permite al autor original revisar la licencia, similar a la disposición de versiones de la GPL.

Licencia	Comentario
Artistic Licence 2.0	Es una licencia modelada sobre la GPL pero sin <i>copyleft</i> . Es algo larga y complicada de entender, a pesar de dividir los usos permitidos en párrafos separados (uso con y sin modificación, distribución con y sin modificación, lo mismo con y sin código fuente, etc.). La versión 2.0 está todavía en "versión beta". La primera versión de esta licencia tiene "agujeros legales" que permiten evitar los requerimientos, por ejemplo, vender el software en paquetes (cuando la venta está prohibida), enlazar o integrarse con programas propietarios (cuando no se debe mezclar con material propietario) o poner materiales en el dominio público. Tiene obligaciones relativas a los nombres, que distinguen entre modificaciones directas (llevan el mismo nombre) y obras derivadas (tienen que tener otro nombre).
Perl	Es una licencia muy particular, mezcla de la GPL y la antigua licencia Artistic. Se puede elegir una u otra. En tanto se elige la GPL, su código es compatible con la GPL, por supuesto. La FSF recomienda usar las versiones 4 o 5.

### 7.3.3. Las licencias sin *copyleft* incompatibles con la GPL



Este tipo de licencias son incompatibles con la GPL (en el sentido de que no se puede integrar material de estos programas en un programa o su obra derivada bajo la GPL), porque las licencias sobre estos materiales incluyen obligaciones que son más restrictivas que la GPL.

En muchos casos, estas licencias derivan de la obligación de publicidad que estaba incorporada en la primera versión de la BSD, pero también pueden surgir de obligaciones sobre patentes, nombramiento, indemnizaciones u otros temas.

#### 1) Apache

El servidor web Apache proviene de los laboratorios de la National Center for Supercomputing Applications de la Universidad de Illinois, EE.UU., y por lo tanto mantiene una licencia de tipo BSD. La versión actual es la 1.1.

#### Nota

La licencia se esconde dentro del código fuente (pero hay una versión en el sitio web de la OSI).

**Nota**

Apache Software Foundation:  
<http://www.apache.org/>

La licencia es una variante de la BSD, pero agrega algunas obligaciones extras:

- Hay una obligación de mantener la publicidad acerca de los autores originales en la documentación o en el software de redistribuciones: "This product includes software developed by the Apache Software Foundation".
- Las obras derivadas no deben usar el nombre Apache sin la autorización de la fundación Apache (para mantener la reputación de los autores originales).

Como cualquier programa bajo la BSD, se puede combinar el código de Apache con otro software, no se puede usar el nombre para fines publicitarios, no incluye ninguna garantía ni los licenciantes se hacen responsables por daños y perjuicios.

Hay que notar que la primera versión de la licencia (1.0) tenía la misma obligación de publicidad, pero en materiales publicitarios que mencionan el producto. Ahora se restringe a la documentación o al código. Sin embargo, por estas obligaciones, no es compatible con la GPL.

## 2) La Netscape Public Licence y la Mozilla Public Licence

Se desarrolló la Licencia Pública Netscape (Netscape Public License, NPL) en 1998, cuando Netscape "abrió" (como software abierto) el código de su navegador de Internet, el Netscape Navigator. El desarrollo de la licencia fue un proceso colaborativo entre varios de los "gurús" del movimiento abierto, como Linus Torvalds, Bruce Perens y Eric Raymond, que intentaron en un principio persuadir a Netscape para que empleara la GPL, pero ante la negativa de Netscape y la necesidad de respetar la propiedad intelectual de terceros, acabaron distribuyendo el código bajo la NPL, una licencia que cumple (justo) con la definición de la OSI.



Otro punto interesante es que, antes de abrir su código fuente al público, Netscape distribuyó un borrador de la licencia propuesta en un foro de noticias (newsgroup)

especialmente creado para opinar sobre la misma (netscape.public.mozilla.license). El proceso de desarrollo abierto para el software fue trasladado al mundo legal, despertó un gran entusiasmo y... ¡críticas! Hubo varias propuestas para modificar algunos términos de la NPL, sobre todo el término que permitía a Netscape utilizar el mismo código en otros productos que no estaban bajo la NPL. El problema no era que el código no se pudiera ceder bajo licencia doble (una práctica ahora común); era que versiones futuras privatizadas por Netscape podían incluir código contribuido por terceros del movimiento de software libre.

Al final, buscando un equilibrio entre los objetivos comerciales y de desarrollo libre de Netscape y la comunidad libre, se resolvió emitir dos licencias, la NPL y la MPL. La primera se aplica al código inicial de Navigator y a las "modificaciones" hechas en el mismo. La segunda se aplicaría a cualquier "agregado" al código y a cualquier programa totalmente nuevo que quisiera usar esta licencia. Ahora se usa la licencia MPL para varios programas, entre los cuales se encuentran el navegador Mozilla y otros programas de Mozilla.org.

Las dos licencias son idénticas, excepto por unos derechos reservados por Netscape en la NPL para su código inicial. Por lo tanto, las presentamos juntas (como N/MPL) y añadimos un apartado especial que describe los derechos particulares de Netscape en la NPL. La versión actual es la 1.1.

#### a) Componentes esenciales de la N/MPL

##### Definiciones

La N/MPL tiene una estructura de licencia clásica y empieza con definiciones importantes que permiten, entre otras cosas, diferenciar entre lo que es código original y código agregado:

- **Desarrollador inicial:** Netscape en el caso de la NPL. En código bajo MPL, el autor inicial indicado en el Anexo a la licencia y cualquier autor de contribuciones.

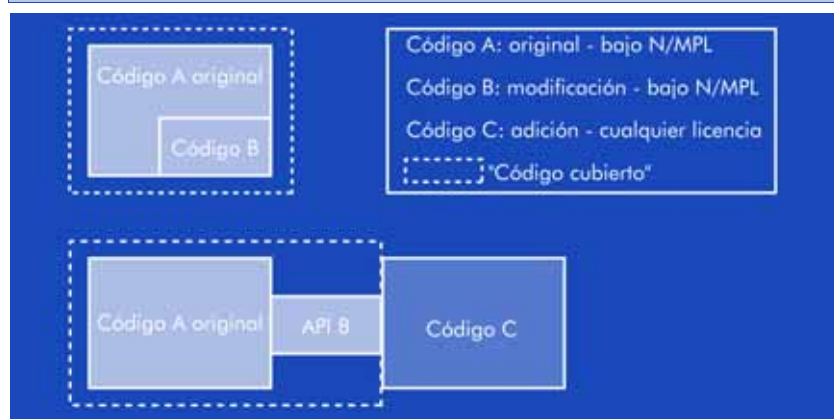
#### Lectura complementaria

La historia de Mozilla está en C. DiBona et al. (Ed.). (2001). "Open Sources: Voices".

- **Código inicial:** NPL, el código liberado por Netscape; MPL, código distribuido por los desarrolladores iniciales.
- **Modificación:** cualquier modificación al código cubierto que NO incluya una simple adición de un fichero nuevo separado o código nuevo que interactúe con el código original sin modificarlo (por ejemplo, a través de una API –aunque la API misma podría ser una modificación, si está integrado en el código cubierto). “Modificación” no se refiere tampoco a toda la obra modificada (que también bajo el derecho es una “obra derivada”), sino únicamente a la parte modificada.
- **Código cubierto** (por la licencia): código inicial más las modificaciones.
- **Contribuidor:** cualquier tercero que modifique código cubierto.
- **Obra mayor:** una obra separada del código cubierto pero que lo puede incorporar o enlazarse con él, sin modificarlo (cl. 3.7).

La definición completa de los diferentes tipos de código permite distinguir entre lo que es software libre (*código cubierto*, sujeto a la NPL o la MPL) y los elementos que se pueden mantener separados y eventualmente privatizar. El sentido de “Modificación”, resumido aquí, explicita muchas cosas que la GPL deja inciertas, sobre todo la cuestión de agregados nuevos que no modifican ninguna parte del código inicial en el momento del desarrollo. Por lo tanto, permite a un desarrollador agregar ficheros y programas separados (propietarios o libres) y distribuirlos separados del código cubierto, pero parte de un programa más grande (potencialmente propietario).

**Figura 2.** Persistencia en la licencia N/MPL



En la figura 2 se ilustran los elementos definidos por la licencia. Una Obra Mayor será constituida por el bloque A + B + C.

### Los derechos otorgados

- El desarrollador inicial otorga una licencia para usar, copiar, modificar y distribuir libremente el código (cl. 2.1).
- El desarrollador inicial otorga una licencia de patente suficiente para permitir el uso del programa y las modificaciones (cl. 2.1). (Estas licencias cruzadas de patentes se comentan en seguida.)
- Cada contribuidor otorga licencias similares relativas a su contribución (modificación) (cl. 2.2).
- Se puede distribuir el código en binario bajo una licencia compatible con la N/MPL, siempre que se respeten las obligaciones contenidas en la licencia, por ejemplo el acceso al código fuente (cl. 3.6).
- Se puede incorporar el código cubierto en una “obra mayor” (que lo incluya, pero no lo modifique) bajo cualquier licencia, siempre que se respeten las obligaciones relativas a la parte de código cubierto (cl. 3.7) (por ejemplo, el acceso al código fuente).

### Obligaciones

- El código fuente del código inicial y de cualquier modificación (Código Cubierto) debe distribuirse bajo la N/MPL, sin cláusulas más restrictivas (*copyleft* para el código cubierto, cl. 3.1).
- Si se distribuye el código cubierto en binario, se debe ofrecer acceso a su código fuente al destinatario de la distribución durante por lo menos 12 meses (cl. 3.2).
- Hay que acompañar cualquier modificación con una copia de la licencia y una indicación de las modificaciones y sus autores, y una indicación de cualquier reclamo conocido sobre el código (LEGAL.txt) (cl. 3.3-3.5).

### Los derechos de Netscape en la NPL

Netscape se reserva algunos derechos especiales:

- Puede recolectar el código cubierto, incluyendo las modificaciones y contribuciones de terceros y hacerlas suyas (privatizarlas), sin incorporarlas a versiones futuras del navegador (cl. IV).
- Puede licenciar a terceros el código cubierto bajo otros términos, en otros productos de Netscape, durante un plazo de dos años (cl. V3).

Estas disposiciones se incluyen para respetar diversas licencias propietarias relativas a código del Navigator otorgadas anteriormente a terceros, que no permitían a Netscape compartir con terceros este código de forma abierta u obligaban a un licenciamiento particular.

### Otros elementos relevantes

La licencia también incluye cláusulas relativas a:

- El cumplimiento en caso de inaplicabilidad de parte de la licencia (4).
- La aplicación de la licencia a código inicial y código cubierto (5).
- Las versiones y el renombramiento de variantes de la licencia (6).
- La ausencia de garantías (7).
- La negación de responsabilidad (9).
- La resolución de la licencia en caso de incumplimiento o litigio con otro contribuyente relativo a patentes (8.2).
- El derecho aplicable y la resolución de conflictos (11).
- Una declaración de responsabilidad mutua de los coautores (12).

Todavía más importante, la nueva versión 1.1 de la licencia MPL incluye una cláusula adicional 13 que permite al autor inicial licenciar el código cubierto, o partes designadas del mismo, bajo **licencia múltiple** (al principio, estaban previstas la GPL y la LGPL). Esto permite el uso comercial, pero más interesante es la posibilidad de distribuir el código bajo la GPL. De esta manera, la nueva versión de la licencia es compatible con la GPL en relación con el código así designado.



## Comentarios

La N/MPL es una licencia mucho más compleja y completa que la GPL y, obviamente, que la BSD. Fue redactada con y por abogados en el contexto de una empresa comercial, por lo que incluye definiciones específicas y contempla cuestiones tradicionales relacionadas con las licencias, como la jurisdicción competente y el derecho aplicable, etc. Aunque su efecto puede parecer más cercano a la BSD que a la GPL, hay varios puntos importantes que hemos de considerar, que comentamos en este apartado:

- 1) El primer comentario, casi evidente, es que se debe usar la NPL (en contraste con la MPL) únicamente para código que provenga de Netscape.



Los derechos reservados de Netscape incluidos en la NPL son únicamente para esta empresa.

Pese a ello, son muy interesantes porque ilustran las dificultades que surgen a la hora de cambiar los términos de una licencia relativos a código ya existente (de Netscape), contribuido (por terceros), adquirido bajo licencia ajena (de terceros) o ya licenciado (a terceros) bajo términos restrictivos para el licenciante. Hubo que considerar en qué grado la empresa podía “liberar” de manera útil el programa sin violar sus propias licencias y contratos comerciales anteriores o los de terceros. La hemos comentado con algo de detalle porque puede servir como metodología (pero no necesariamente modelo) para crear una licencia nueva o liberar un programa complejo ya existente.

- 2) **Persistencia y copyleft:** estas licencias tienen un efecto de persistencia parcial, como la LGPL:

- El código cubierto (incluida cualquier modificación) debe mantenerse bajo la licencia N/MPL.
- Cualquier extensión (una obra mayor) puede ser propietaria. Además, es muy fácil crear un archivo adicional propietario que llame al código original bajo N/MPL y distribuirlo todo bajo una licencia propietaria. Esto sigue la filosofía de la licencia BSD. Sin embargo, en todos los casos, el código fuente de la parte libre original debe distribuirse u ofrecerse al destinatario.

**Nota**

Cualquier código bajo la licencia MPL 1.0 es incompatible con la GPL.

3) En cuanto a la **compatibilidad con la GPL**, cualquier código bajo la licencia MPL 1.0 es incompatible con la GPL, fundamentalmente, y entre otras cosas, porque tiene demasiadas restricciones adicionales relativas a las patentes y la posibilidad de enlazarse con programas propietarios. La posibilidad de licencias múltiples ofrecida por la versión 1.1 facilita la compatibilidad si se elige la GPL.

4) En cuarto lugar, comentamos las **nuevas cláusulas de patentes** incluidas en la MPL. La cláusula resolutoria (cl. 8), combinada con la licencia de patentes (cl. 2.1), es parte de una nueva generación de cláusulas en las licencias libres, para crear un entorno de trabajo libre de patentes y libre del riesgo de patentes. Constituye lo que se llaman “las licencias cruzadas de patentes”. Los desarrolladores no pueden impedir que una persona solicite y obtenga una patente (en EE.UU.) sobre un proceso que puede ser parte de una modificación del programa inicial. El riesgo es que el uso o una modificación posterior pueden violar dicha patente si el usuario futuro no tiene una licencia de patente adecuada. Por lo tanto, estas cláusulas intentan realizar dos cosas:

- Por un lado, la persona “patentadora” debe otorgar (prospectivamente) a todos los otros licenciarios (usuarios y desarrolladores) una licencia de patente respecto al proceso o código patentado.
- Por otro lado, se rescindirán las licencias de derechos de autor (y de patente, si las hay) otorgadas a esta persona “patentadora” en el caso de cualquier litigio o intento de impedir la explotación libre de la modificación.

Por lo tanto, por ejemplo, Netscape renuncia a cualquier beneficio de una patente eventual sobre el código fuente del Navigator.

5) El equilibrio comercial



Los conceptos de *modificación* y *obra mayor* han sido cuidadosamente elaborados para encontrar un equilibrio entre la libertad de la BSD, que permite un uso ilimitado del código, y la libertad de la GPL, que obliga a mantener el código libre. Es decir, entre la promoción del desarrollo de software libre por empresas comerciales y la protección del trabajo de desarrolladores “libres”.

Este justo medio ha sido definido en la diferencia entre una modificación y una adición. Recordemos que la GPL, en contraste, afecta las adiciones que se vinculan demasiado íntimamente con código GPL.

- 6) **LEGAL.txt**: otro aspecto muy interesante de la MPL es la inclusión de este archivo, donde los autores deben consignar cualquier aviso de reclamo, litigio o restricción sobre una parte del código. Demuestra un conocimiento evidente del proceso de desarrollo libre, donde el riesgo de denuncias relativas a la propiedad intelectual e industrial es alto y la información transparente es primordial. Un desarrollador posterior debería utilizar este fichero para estudiar las limitaciones legales de un código provisto por terceros, quizás en relación con un litigio de patente, quizás por las limitaciones de ciertas partes de código que pueden estar bajo una licencia compatible pero diferente de la MPL.



Las labores de Netscape, y ahora mozilla.org, nos pueden enseñar varias cosas en relación con la creación de licencias libres.

Demuestran una tendencia de los participantes comerciales en el movimiento libre a redactar sus propias licencias. IBM tiene la IBM Public Licence, Sun, Apple y Microsoft intentan formular unas variantes más comerciales (comentadas más adelante). A veces se trata de “variaciones” para mejorar aspectos inciertos, como el tema de obra derivada en la GPL o las patentes, otras veces regulan temas en particular (aquí, los derechos y obligaciones de Netscape).

El proceso de creación de la MPL y el rechazo de los derechos reservados por Netscape demuestran que hay realmente un “efecto de comunidad” entre el movimiento libre. Como consecuencia, una licencia inadecuada será rápidamente rechazada. La historia de las licencias “Shared Source” de Microsoft y Sun confirma esto.

La MPL es casi un nuevo modelo de licencia abierta (pero no necesariamente FSF-libre) por excelencia, por sus orígenes mixtos (empresa comercial, movimiento libre, expertos legales) y por sus objetivos de desarrollo y explotación. Se adecua a muchos contratos y licencias comerciales, por lo tanto las empresas se sienten más cómodas con ella.

Otra observación relativa a la licencia MPL que hemos de hacer, es que contiene unas cláusulas más completas y, diríamos, más útiles para resolver problemas de violaciones de derechos de terceros, licencias de patentes y un mecanismo para el seguimiento de los autores. Esto será muy útil en el momento de buscar o excluir (en la medida de lo posible) una cierta responsabilidad legal por violación de derechos.

Finalmente, hay que notar que varias personas argumentan que esta licencia es demasiado complicada y, para garantizar una mayor transparencia en el mundo del software libre, conviene utilizar únicamente las licencias de las dos grandes familias, la GPL y la BSD.

### 3) Otras licencias no compatibles con la GPL

Como hicimos para las licencias anteriores, añadimos a continuación una tabla de resumen de otras licencias libres sin *copyleft* que no son compatibles con la GPL.

**Tabla 7.** Otras licencias no compatibles con la GPL

Licencia	Comentario
Python 2.0.1, 2.1.1 y versiones siguientes	Una licencia similar a la BSD, pero que obliga a aplicar el derecho del estado de Virginia al programa (y potencialmente sus obras derivadas). Como la GPL no tiene una cláusula de derecho aplicable, esto constituye una restricción adicional incompatible con ella.
PHP (versión 3.0)	Una licencia de la familia de BSD, pero incluye la obligación de incorporar una cláusula de publicidad de PHP.
Apple Public Source License (versión 2)	Es una variación de la MPL, con nuevos elementos como el derecho aplicable (California y para cubrir la posibilidad de ofrecer servicios por Internet (" <i>Externally deployable</i> "), similar a la Affero. Aunque las licencias iniciales de Apple Public Source no eran libres, se ha modificado la versión 2.0 para que lo sea: han eliminado la obligación de devolver a Apple cualquier modificación y la posibilidad de revocar en cualquier momento la licencia inicial (ved más adelante para un comentario breve sobre ello). Permite enlazar código bajo APSL 2.0 con programas no libres, por lo tanto, no es <i>copyleft</i> y por la obligación de licenciar las patentes, queda incompatible con la GPL.
The Q Public License (QPL), Versión 1.0.	Una licencia que obliga a distribuir cualquier modificación como un parche sobre el programa inicial. Además, hay que remitir al proveedor inicial (Trolltech) cualquier modificación que no esté a disposición del público.

## 7.4. Otras licencias de tipo "libre"

Hemos estudiado en bastante profundidad las principales licencias libres y hemos comentado sus rasgos particulares, compatibilidades

y consecuencias. En este nuevo apartado queremos completar nuestro análisis de las licencias llamadas “libres”, por el orden siguiente:

- 1) Las licencias que llamaremos “seudolibres”, que tratan de emular las libres pero contienen alguna restricción que no cumple con las libertades de la FSF o las directrices de la OSD.
- 2) Las licencias de documentación libre.
- 3) Las licencias de *freeware* y *shareware*, que no son en nada “libres”.

#### 7.4.1. Las licencias de software “seudolibres”

Aunque en esta unidad enfocamos las licencias de software libre, un breve análisis de las licencias creadas por empresas comerciales que intentan beneficiarse del modelo de desarrollo libre –sin pagar todos sus “costes”– es interesante. Primero, indica el abanico de posibilidades entre lo libre y lo propietario. Asimismo, permite elucidar la posición de dichas empresas al respecto e indicar algunas estrategias a evitar desde el punto de vista de las licencias libres.

##### 1) La licencia Sun Community Source

La licencia SCSL es una tentativa de ofrecer acceso al código y entornos de programación de Sun, por ejemplo, Java o Jini, y establecerlo como un estándar. En esto, ha tenido mucho éxito, sobre todo en relación con Java. Los “componentes” incluidos en la Sun Community son el *J2EE*, el Java Developers Kit (JDK), Personal Java y Embedded Java, entre otros.

La SCSL es, sobre todo, una licencia para desarrolladores, lo que es normal, dado que se aplica a lenguas y entornos de programación. Se “abre” principalmente a los fines de la investigación y desarrollo, sin embargo, la licencia permite a Sun mantener un control muy fuerte sobre la evolución del programa y los entornos de programación.



Ver en la bibliografía las obras siguientes:

Richard Gabriel; William Joy (1999). *Sun community source license principles*.

Stig Hackv  n (1998). *Not quite open source, but closer*.



Conceptualmente, esta licencia está a medio camino entre la MPL y una licencia propietaria: permite correcciones, modificaciones y extensiones, pero cualquiera de las mismas debe ser devuelta a Sun.

Los licenciarios bajo la SCSL tienen derechos en tres categorías:

- a) **Para la investigación:** se puede copiar, modificar y distribuir el código sin pagar regalías o cánones, bajo las condiciones usuales de código abierto, pero hay que enviar a Sun cualquier corrección de error y las API para extensiones, con una licencia para usarlos e incorporarlos a versiones futuras de su programa (ilibre o no!).
- b) **Para el uso interno:** cualquier distribución de código debe ser compatible con los criterios técnicos de Java. Hay que usar el logotipo de Sun y comprar una licencia para él. Asimismo, hay que pasar los exámenes técnicos de Sun, por ejemplo, el Jini Technology Compatibility Test.
- c) **Para el uso comercial** (para desarrollar software para clientes): además de lo anterior, se debe concluir un contrato de soporte con Sun. Sin embargo, se puede comercializar el código objeto bajo cualquier licencia (no tiene efecto *copyleft*). La distribución de código fuente de la plataforma se hace bajo la licencia original.

La licencia incluye un acceso a las especificaciones técnicas de las tecnologías Sun, pero no se permite hacer una reingeniería inversa (bajo amenaza de patente).

Como vemos, debido a las obligaciones adicionales (sobre todo, la de devolver a Sun cualquier modificación y cumplir con las especificaciones técnicas de Sun), esta licencia no cumple con las directrices de la OSI y menos con la GPL. Es más, ha suscitado una serie de críticas por ser “falsa”, tratar de venderse como una plataforma libre y permitir que Sun se aproveche de las modificaciones, arreglos y correcciones de errores (*bug fixes*) de los desarrolladores terceros. Por otro lado, la obligación de cumplir con los criterios de Sun le permite mantener un cierto control de calidad sobre el uso de su tecnología.

## 2) Microsoft Shared Source Initiative

Microsoft también creó una serie de licencias semilibres para una parte de sus programas. Se aplican al sistema operativo CE para dispositivos portátiles y CLI (*Common Language Infrastructure*) y las especificaciones de C#. También incluye elementos de Windows 2000 y XP. Este “gesto” permite sobre todo el estudio académico de las tecnologías en cuestión y, para las empresas comerciales que crean productos que se ejecutan sobre estas plataformas, integrar mejor sus programas con las de Microsoft.

Hay varias licencias dentro de la Iniciativa Shared Source. El modelo básico, por ejemplo, la licencia Shared Source de CE, abre el código a investigadores y estudiantes:

- Se puede descargar y estudiar el código fuente.
- Se puede usar el código para cualquier uso no comercial.
- Se puede modificar y distribuir las modificaciones para usos no comerciales, siempre que se mantenga la misma licencia.
- No se permite el uso, copia, modificación o distribución para fines comerciales.



Para más información sobre Shared Source, podéis consultar:

<http://www.microsoft.com/latam/licenciamiento/sharedsource/default.asp>.

Por restringir algunos usos del software, estas obligaciones hacen que no se pueda considerar las licencias como libres bajo las directrices de la OSI.

Luego, con el Windows CE Shared Source Premium Licensing Program, los fabricantes de aparatos OEM tienen acceso al código fuente de Windows CE y tienen el derecho de modificarlo y distribuir las modificaciones de manera comercial. Sin embargo, tienen que licenciar cualquier modificación a Microsoft de manera gratuita, lo que

permite a Microsoft incorporarlas en versiones posteriores del software después de un período de 6 meses.

Otras licencias de MSSl tienen variaciones sobre estos derechos otorgados y reservados. La licencia para ASP.net, por ejemplo, permite cualquier uso comercial y no-comercial, pero prohíbe combinar o distribuir el programa ASP.net con programas cuya licencia obligue al usuario (1) a distribuir el programa junto con su código fuente o (2) a permitir la modificación del software o la creación de obras derivadas, o (3) a distribuirlo sin cargo (por ejemplo, *freeware*). Los dos primeros términos parecen ser, básicamente, una referencia a software libre bajo la GPL y similares. Ello impide mezclar este código ASP.net con cualquier programa libre.

Observad que las leyes de derechos de autor no permiten controlar los usos de materiales protegidos por ellas. Por lo tanto, las limitaciones relativas a los “usos comerciales” pueden no ser vinculantes en el caso de una simple licencia (bajo derecho anglosajón). Por el contrario, serán efectivamente vinculantes si se puede probar la existencia de un contrato válido (lo que Microsoft intenta garantizar por varios métodos de *click-wrap* y *browse-wrap*).



Para la perspectiva de la FSF sobre Shared Source, podéis consultar en la bibliografía:

Bruce Perens (2002). *MS'Software Choice' scheme a clever fraud.*

### 3) Apple 1.x License

Las licencias Apple Public Source 1.0, 1.1 y 1.2 siguen el modelo de la MPL, con algunos elementos particulares:

- La APSL 1.0 no permite modificar el software para su uso personal sin publicar los cambios.
- Cualquier uso comercial o distribución de código modificado debe notificarse a Apple, entregándole el código fuente con una licencia de uso gratuita.



- Apple puede revocar la licencia en cualquier momento en el caso de alegación de violación de patente o de derechos de autor.

Luego, Apple creó dos versiones nuevas, la APSL 1.1 y la 1.2:

- La 1.1 modifica los derechos de “resolución” de Apple por una “suspensión” en relación con litigios de propiedad intelectual o industrial.
- En 2001, la 1.2 eliminó las restricciones de modificación y estos derechos de suspensión, pero mantuvo la obligación de entregar el código modificado a Apple.

La APSL 1.2 fue reconocida abierta por la OSI (icon muchas críticas de la FSF!). Ahora se ha publicado la versión 2.0, comentada más arriba, que es una licencia plenamente abierta.

#### 4) Aladdin Free Public License

La licencia Aladdin Free Public License (AFPL) relativa a Ghostscript merece una mención especial, porque tiene un carácter particular. No cumple con la OSD, aunque se inspira directamente en la GPL. Lo interesante es que mientras que la última versión disponible de Ghostscript se distribuye bajo la AFPL y obliga a obtener una licencia propietaria para usos comerciales, la penúltima versión del software se libera bajo la GPL. Por lo tanto, se comercializa la “mejor” versión del programa y los desarrolladores libres pueden aprovechar el código un poco más antiguo.

#### 7.4.2. Las licencias de documentación libre

Las licencias libres se aplican en su gran mayoría, pero no solamente, al software. Se ha creado una serie de licencias libres para documentación, sobre todo, porque el software va acompañado de una documentación técnica a menudo necesaria para su uso. No tendría sentido distribuir el software libre sin distribuir la documentación correspondiente bajo términos similares. Por ello, la FSF creó la General Free Document License para acompañar sus programas.

Obsérvese que en ciertos regímenes legales, por ejemplo, el de España, el software debe ir acompañado de la documentación correspondiente y la licencia del primero cubre la segunda. Este no es el caso de los países anglosajones, donde la obligación está limitada a los materiales de diseño (y no alcanza a los manuales de uso, etc.).

Además, siguiendo la tendencia a la apertura del conocimiento, se han creado otras licencias sobre documentación y materiales, sobre todo, académicos. Presentaremos un ejemplo: la iniciativa Creative Commons.

### 1) La licencia de documentación libre de GNU (la GFDL)

La GFDL se destina a usarse para documentación técnica, manuales de usuario y otros textos relevantes para el software libre. Se modela sobre la GPL, cambiando los pactos necesarios para adecuarse a un texto escrito en lugar de al software. La licencia busca el equilibrio entre permitir las modificaciones (sobre todo, aquellas necesarias para documentar una modificación del software), mantener la autoría de la obra inicial y respetar las ideas y opiniones de los autores originales.

#### a) Componentes esenciales de la GFDL

La licencia define varios elementos de un documento, para establecer los derechos y obligaciones correspondientes a cada elemento:

- Secciones identificadas como secundarias: avisos legales, dedicciones, reconocimientos, etc.
- Secciones invariables: son secciones secundarias determinadas expresamente como tales.
- Copia transparente: una copia legible y modificable por un tercero (por programas no propietarios o genéricos) como ASCII, XML con DTD público, HTML, etc., similar al código fuente de un programa. En contraposición a una copia opaca (por ejemplo, un fichero PDF).

La GFDL otorga varios derechos relativos a la copia, distribución, modificación, agregación y combinación, colección y traducción del documento original, que están, en su mayoría, permitidos bajo condiciones de respeto de la autoría original, manteniendo las partes invariables y previendo acceso a una versión transparente.

La modificación implica una serie de obligaciones: cualquier obra derivada debe cambiar de título en la portada, detallar los autores originales y las modificaciones, indicar dónde se puede encontrar la versión original y mantener los avisos de *copyright* y la licencia. Asimismo, se debe mantener las secciones invariables y el tono y contenido general de las secciones secundarias. Hay que eliminar de las obras derivadas cualquier indicación de patrocinio (*endorsements*).

#### **b) Otros aspectos relevantes y comentarios**

Como la GPL, la GFDL mantiene el *copyleft* de los documentos: hay que distribuir cualquier modificación bajo la misma licencia y no se puede combinar con texto que provenga de una obra bajo cualquier licencia más restrictiva.

Esta licencia ha provocado mucha discusión, sobre todo en lo relativo a las secciones invariantes. Hasta la misma FSF publica artículos explicando por qué no se debe usar la licencia: es muy restrictiva en algunos aspectos y demasiado flexible en otros. Por ejemplo, permite a segundos o terceros autores identificar elementos invariables, pero existe la posibilidad de que uno sea obligado a mantener invariables ciertas partes equivocadas u obsoletas. Tampoco se puede usar texto bajo GFDL como texto de ayuda en un programa de software interactivo (por tanto, es incompatible con la GPL).

La licencia no solamente puede aplicarse a documentación técnica para software: se puede usar sobre cualquier texto, específicamente cualquier obra “literaria” que se desarrolle a la manera del software libre: por obras en colaboración.

La GFDL no es la única licencia de documentación libre: en parte, debido a la polémica sobre la misma, muchos proyectos de software libre crearon su propia licencia: la FreeBSD Documentation License,

**Nota**

Creative Commons:  
<http://creativecommons.org/>

la Apple Common Documentation License, o la Open Publication License, la OR Magazine License (de O'Reilly).

## 2) La iniciativa Creative Commons

La iniciativa de *Creative Commons* ('espacio público creativo' o 'comunidad creativa' serían posibles traducciones en castellano, abreviada CC) es un proyecto afincado en la Universidad de Stanford, California, creado por una serie de expertos en derechos de autor. Intenta ayudar a los autores y creadores a distribuir libremente sus obras para uso del público, ampliando por tanto el número de obras creativas accesibles a todos. Se dirige, sobre todo, a las creaciones literarias y artísticas –no al software– y recomienda expresamente la GFDL para cualquier documentación informática. Además, la CC propone un sistema privado, bajo derecho americano, para limitar la duración de la protección de *copyright* a 14 años, en vez del plazo acordado por ley (generalmente, la vida del autor más 70 años) sobre la base de una declaración pública. Finalmente, permite dedicar obras al dominio público, también bajo las condiciones del derecho de autor de EE.UU.

**Nota**

La iniciativa Creative Commons actúa bajo un lema que es un juego de palabras sobre la reserva habitual de derechos de autor "All rights reserved". El lema es "Some rights reserved" ('algunos derechos reservados'), similar a la de la FSF: "All rights reversed" ('todos los derechos invertidos'). La licencia más libre de CC permitiría hasta usar la expresión "No rights reserved" ('ningún derecho reservado').

La CC ha creado una serie de licencias modulares que permiten a los autores establecer sus derechos de autor y a los licenciarios realizar una serie de actos de explotación, de manera muy similar a las licencias libres de software. Tiene dos aspectos interesantes. Primero, la licencia viene en tres formatos:

- Una versión legal para abogados: la versión completa de la licencia (*legal code*).

- Una versión fácil de leer: un resumen muy fácil de entender (*commons deed* o *human code*).
- Una versión legible por ordenadores: una expresión en *meta-data* RDF y XML para que un proceso informático automatizado pueda entender la licencia en el contexto de la web semántica (*digital code*).

En segundo lugar, los usuarios pueden elegir los derechos que se reservan y se otorgan en la licencia en función de cuatro criterios: atribución, usos comerciales, obras derivadas y herencia (*copyleft*). El sitio de CC contiene una herramienta automatizada para crear la licencia a partir de las repuestas a preguntas sobre dichos criterios. La herramienta crea y publica los archivos de la licencia y provee el documento resumen (*commons deed*) con iconos para una comprensión visual de la misma.

Como consecuencia, la licencia puede tener las siguientes características acumulables:

- *Attribution* (**atribución**): se permite cualquier acto de explotación y la derivación, siempre que se de crédito al autor original.
- *Noncommercial* (**no comercial**): se permite cualquier acto de explotación y la derivación, siempre que sea para fines no comerciales.
- *No Derivative Works* (**ninguna obra derivada**): no se permite modificar para crear obras derivadas.
- *Share Alike* (**compartir de manera igual**): se permite la redistribución de la obra y de obras derivadas solamente bajo términos iguales a la licencia original (*copyleft*).

#### Ejemplo

Algunos ejemplos de las licencias potenciales incluyen:

- a) La licencia "Attribution-NonCommercial-ShareAlike" permite la modificación, obliga a mantener la misma licencia en obras derivadas y prohíbe los usos comerciales.

#### Ejemplo

La licencia MIT OpenCourseWare es de este tipo, en:

<http://ocw.mit.edu/OcwWeb/Global/terms-of-use.htm>

Otro sitio con esta licencia es "HOWTO: Installing Web Services with [free software]", en:

[http://daydream.stanford.edu/tomcat/install\\_web\\_services.htm](http://daydream.stanford.edu/tomcat/install_web_services.htm)

- b) La licencia "Attribution-Noncommercial" obliga a dar crédito y restringe los usos comerciales. La Electronic Freedom Foundation en [www.eff.org](http://www.eff.org) usa esta licencia.

La licencia más libre (sin ser de dominio público) es la "Attribution" que obliga a dar crédito y permite todo lo demás.

LA CC ha inventado interesantes iconos para indicar las características de las licencias:



Además, las licencias contienen un núcleo de términos comunes a todas las variantes:

- Se obliga a mantener los avisos de autoría y *copyright*.
- Se permite establecer vínculos de Internet a las obras publicadas en ese medio.
- No se permite modificar la licencia.
- No se permite utilizar medios tecnológicos para restringir los usos legítimos de la obra (es decir, no se permite el uso de tecnologías de DRM).
- Se aplica a todos los países del mundo.
- Es irrevocable y dura para el plazo de la protección de *copyright*.
- Ofrece una garantía de titularidad y de no violación de derechos de terceros (para aumentar la confianza en el reuso y redistribución de la obra).

- Se permite al autor o titular de derechos distribuir la obra bajo una licencia diferente.
- Contiene una excepción especial que permite compartir ficheros (P2P *file-sharing*), lo cual no es considerado como una actividad comercial, siempre que no tenga fines de lucro.

Por lo tanto, la CC es una innovación que aprovecha las últimas tecnologías para flexibilizar la creación y uso de licencias en un entorno digital y, en consecuencia, la distribución de obras en Internet.

#### 7.4.3. Licencias de tipo freeware y shareware

Únicamente, se quiere comentar aquí que las licencias de tipo *shareware* y *freeware* no son licencias libres. Aunque los programas correspondientes puedan distribuirse gratuitamente, no proveen acceso al código fuente del código y, en su gran mayoría, no respetan las condiciones mínimas para ser libres o abiertas: las cuatro libertades de la FSF o las directrices de la OSD. Por lo tanto, no las incluimos en este estudio.



Para un comentario breve sobre dichas licencias, podéis consultar:

Peter L. Deutsch (1997). *Tipos de licencias para software redistribuible libremente*.

### 7.5. Conclusiones

Para terminar esta unidad, queremos comentar la evolución más reciente en materia de licencias libres.

El software y las licencias no existen en un entorno estable e invariante, sino en un medio cambiante que evoluciona con rapidez, tanto técnica como legalmente. Por un lado, por ejemplo, los enlaces dinámicos y los métodos de Java no existían cuando se redactó la licencia GPL v.2.0. Por lo tanto, la GPL se interpreta con dificultad en relación con estas nuevas tecnologías. Por otro lado,

la protección de la información de identificación de derechos (Rights Markup Information, RMI, que incluiría los avisos de autoría tradicionales) no existía hasta los tratados de la OMPI de 1996 y su implementación a través de la DMCA, en EE.UU., y la nueva Directiva de protección de derechos de autor en la sociedad de la información, en Europa. Tampoco se tuvo en cuenta esta protección en las licencias más antiguas, que en ciertos casos permiten retirar avisos de autoría.

En consecuencia, para mantener la libertad del código, se requiere una adaptación continua de la tecnología al marco legal y viceversa. En relación con el software libre, el marco legal incluye las licencias y, por ello, las licencias permiten actualizarse. Pero ya hemos visto que tratar de actualizar una licencia puede ser difícil, sino imposible, a menos que se mantenga un registro actualizado de autores y derechos y que haya una persona que tenga interés en defender el estado del código. En un modelo de desarrollo difuso como el del software libre, esto resulta muy difícil y poco factible.

La FSF, por ejemplo, argumenta que la cláusula que permite actualizar la GPL (y dejar a los usuarios la libertad de aplicar la versión que quieran) es una solución parcial a este problema. La FSF ya habla de una licencia GPL 3.0, que tendrá en cuenta las nuevas formas de distribución de software y de ofrecer servicios informáticos sin distribuir el software (en sistemas distribuidos y a través de servicios web, como los ASP –*Application Service Providers*). La licencia Affero ya está en esta línea.

Sin embargo, se argumenta que las licencias no son suficientes para mantener la libertad del código en un mundo en rápida evolución. La FSF propone otro modelo interesante, que agrega una estructura y unos procesos globales para proteger el software libre. Predican la centralización de los derechos de autor en un solo titular (por ejemplo, la FSF) que pueda gestionar estas licencias y su evolución frente al derecho y la tecnología: un fiduciario a nivel internacional, que reaccione con los cambios legales y tecnológicos y que tome una postura activa en la defensa de las libertades. Es cierto que hasta hoy, la FSF ha sido un modelo muy eficaz en la protección de código bajo GPL contra el abuso y la privatización.



Se propone una “Licencia Fiduciaria” entre los autores de software libre y la Fundación, que:

- a) Transfiere los derechos exclusivos de los autores a la FSF (en la medida de lo posible, bajo el derecho aplicable),
- b) Garantiza el uso de estos derechos para mantener la libertad del código.
- c) Reserva para los autores unos derechos individuales de explotación ilimitada.

Finalmente, queremos resaltar la **importancia de las licencias en el uso**, distribución y, eventualmente, la comercialización de software libre. Hay una gran variedad de licencias, que contienen una gran variedad de derechos y condiciones de uso (restricciones):

- 1) Para los usuarios, en un entorno de programación cada vez más complejo, como los entornos distribuidos (*web-services*, *online ASP*, etc.) o la programación por componentes, el estudio de las licencias que se aplican a los programas deseados –y sus diferencias y compatibilidades– es fundamental.
- 2) Asimismo, para los desarrolladores de software, la gestión de la propiedad intelectual de los contribuidores al código y de los usuarios intermedios y finales, es también fundamental. Es importante decidir de antemano la licencia que se va a usar y conocer los aspectos relevantes –ventajas, inconvenientes, consecuencias– de cada licencia, sobre todo antes de empezar a crear una licencia nueva!

#### Nota

Para más información, podéis consultar:  
<http://fsfeurope.org/projects/fla/>.



## 8. Los efectos prácticos de las licencias de software libre

En las unidades anteriores hemos presentado el marco legal del software en general y el software libre en particular, junto con un análisis y discusión de las licencias de software propietarias y libres. Ahora, ya armados con el conocimiento correspondiente, queremos cerrar la parte teórica de este curso con una reflexión sobre las consecuencias prácticas de las licencias de software libre (y de los aspectos legales del software libre en general) para los diferentes actores que se relacionan con él.

Asimismo, consideramos importante situar las licencias de software en relación con otras ramas del derecho desde un punto de vista práctico.

Por lo tanto, vamos a empezar por comentar algunos temas y efectos prácticos relativos a las licencias libres tales como la gestión de las contribuciones en proyectos libres, las licencias duales, la compatibilidad entre licencias y el cambio de una licencia libre o propietaria a otra.

En segundo lugar, vamos a reflexionar sobre algunos aspectos legales del software libre que no hemos visto hasta ahora: la relación entre estas licencias y otras ramas del derecho, especialmente, el derecho de la competencia y las licencias de patentes y de marcas, y la interrelación entre licencias libres y el proceso de estandarización formal.

Cerraremos la unidad con el estudio de dos importantes áreas del derecho relacionadas con el software libre que no se han desarrollado hasta ahora: por un lado, la protección de datos personales y de la privacidad y, por el otro, el control de la exportación de materiales y productos de criptografía o cifrado.

Con el aprendizaje de esta unidad, el lector será capaz de:

1. Valorar las consecuencias prácticas y algunos aspectos relativos a las licencias libres como la elección de una licencia libre, la com-

patibilidad entre las licencias, licencias duales o múltiples (*dual / multiple licencing*) o la bifurcación o división de código (*forking*).

2. Comprobar la relación entre licencias de software libre y otras ramas de derecho como las marcas, la competencia leal o el proceso formal de estandarización.
3. Conocer las obligaciones y derechos relativos a la privacidad (bajo el régimen europeo) y su vínculo con el software libre, sobre todo, en el ámbito de la seguridad.
4. Conocer la regulación de productos de seguridad y su relación con la distribución de software libre.

### 8.1. Algunos temas legales relacionados con las licencias

En el hecho de crear, distribuir o usar software libre, no se trata sólo de decidir qué software usar y descargarlo desde Internet o, en el caso de su desarrollo, de contribuir en el equipo de gestión del programa. Además de los aspectos técnicos y económicos, en tal decisión intervienen una multitud de temas legales importantes. Para tener éxito en cualquier actividad que involucre el software libre, ya sea su creación y distribución, o su implementación en organizaciones públicas o privadas, es necesario considerar una serie de factores legales y establecer las estrategias pertinentes.

Llamemos a esto la comprensión de los “metaaspectos” legales del software libre, es decir, no solamente los aspectos legales puros y duros (la propiedad intelectual, las licencias, etc.), sino también conceptos más amplios como las consecuencias legales de las licencias o las interrelaciones entre diferentes conceptos clave. Esta comprensión ayudará a entender, no solamente un caso particular de aplicación del software libre, sino también los debates públicos, la evolución de las licencias y los contratos de distribución y hasta los litigios pasados y futuros sobre el tema.

#### Nota

Podéis ver el caso de SCO en el apartado 7 de la unidad 10.

Por poner un ejemplo, el conflicto SCO debería ser, por sí solo, motivo de reflexión para quien quisiera instalar el sistema operativo GNU/Linux v. 2.2, el software objeto del litigio.

**Nota**

En este apartado queremos comentar algunos de estos elementos que hemos llamado “efectos prácticos relacionados con las licencias libres”. Estos efectos prácticos tienen que ver sobre todo con la gestión de las licencias y de la propiedad intelectual e industrial en el software relacionado. En concreto vamos a estudiar:

- Cómo elegir una licencia libre.
- Cómo gestionar las contribuciones a proyectos de software libre.
- La compatibilidad entre licencias.
- El licenciamiento dual o múltiple (*dual/multiple licensing*).
- Cómo cambiar de una licencia a otra.
- El efecto de las licencias sobre la división (*forking*) del software libre.

Terminaremos el apartado con un repaso de algunos problemas legales que pueden surgir en el momento de usar una licencia libre.

### 8.1.1. Elegir una licencia libre



Las disposiciones incluidas en las licencias de software libre resultan normalmente del compromiso entre varios objetivos determinados por los autores o jefes de equipo (coordinadores) de los proyectos de desarrollo libre.

En particular, podemos citar los siguientes propósitos que, en cierta manera, pueden contraponerse unos a otros:

- a) Garantizar ciertas libertades básicas comunes a todo software libre (uso, copia, modificación, redistribución).

- b) Imponer algunas condiciones o restricciones (el reconocimiento de autoría, la falta de garantía, etc.).
- c) Procurar que las modificaciones y obras derivadas sean también libres.
- d) Reservarse algunos derechos (por obligación o por voluntad propia).
- e) Mantener el control sobre la evolución del programa.

En la selección de una licencia, los creadores y titulares de los derechos elegirán las condiciones para la protección del software según el grado de libertad y la obligación que quieran imponer a los usuarios y las consecuencias que dichas libertades y obligaciones puedan tener sobre el autor y sobre la evolución del software.

La GPL, por ejemplo, intenta ampliar la cantidad de software libre disponible (*pool*) y maximizar la libertad de los usuarios finales; por lo tanto, impone la cláusula *copyleft*, que obliga a mantener libres, en cualquier distribución secundaria, las obras derivadas. Asimismo, la GPL tiene el efecto práctico de restringir la división de código (*forking*) y de permitir que un equipo de coordinación mantenga cierto control sobre el programa.



Algunas preguntas que el lector puede plantearse son las siguientes:

- ¿Quiero permitir la privatización de obras derivadas y modificaciones?
- ¿Quiero que los desarrolladores devuelvan sus modificaciones a la comunidad libre en general o a mí como autor inicial en particular?
- ¿Quiero permitir que los licenciarios puedan fusionar o enlazar su programa con el mío?
- ¿Quiero una mayor difusión del programa y tratar de establecer un estándar?

- ¿Quiero obtener beneficios de mi programa a partir de su uso comercial u otro, al mismo tiempo que permitir el desarrollo libre?
- ¿La reputación es importante para mí?
- ¿Tengo obligaciones para con terceros en relación con el código incorporado en mi programa?
- ¿Tengo un programa innovador único, o es otro editor de texto (por ejemplo), cuando ya hay “miles” de disponibles, tanto libres como propietarios?
- ¿Mi programa debe ejecutarse con otro en particular? ¿Tiene restricciones?
- ¿Quiero incentivar a otros desarrolladores para que participen en mi proyecto y contribuyan con código u horas de pruebas?
- ¿Mi aplicación es para ser integrada (*embedded*) en un dispositivo, junto con otro software propietario?
- ¿Hay una licencia “predominante” en el sector de mi software en particular (por ejemplo, un lenguaje o bibliotecas)?
- ¿Hay riesgo de que alguien tenga o pida una patente sobre un elemento o aspecto del programa?

La tabla siguiente, que ya es “clásica” (aparece en casi todos los escritos sobre el tema), toma en cuenta las principales licencias libres y asiste en la selección de una licencia.

Tabla 8.

Tipo de licencia	Se puede mezclar con software no libre	Se pueden privatizar las obras agregadas	Se pueden privatizar las modificaciones	Puede ser usado por cualquiera	Privilegios especiales para el autor original	Requiere licenciar patentes
GPL						X
LGPL	X					X

Tipo de licencia	Se puede mezclar con software no libre	Se pueden privatizar las obras agregadas	Se pueden privatizar las modificaciones	Puede ser usado por cualquiera	Privilegios especiales para el autor original	Requiere licenciar patentes
NPL	X	X			X	X
MPL	X	X				X
BSD	X	X	X			
Dominio público (USA)	X	X	X	X		

Los principales actores involucrados en el software libre recomiendan no crear una licencia nueva, sino usar una existente, sobre todo, por temas de compatibilidad y por el deseo de no ver proliferar más licencias libres (que hay que analizar en profundidad para ver si son compatibles con otras).



#### Lecturas para asistir en la elección de una licencia:

- Zooko O'Whielacronx: Quick Reference For Choosing a Free Software License  
[http://zooko.com/license\\_quick\\_ref.html](http://zooko.com/license_quick_ref.html).
- Bruce Perens: The "Open Source Definition", en *Open Sources* (p. 185).
- Donald K. Rosenberg: "Evaluation of Public Software Licenses", en línea en:  
[http://www.stromian.com/Public\\_Licenses.html](http://www.stromian.com/Public_Licenses.html) (última visita: 29 de marzo de 2001).
- Frank Hecker: "Setting Up Shop: The Business of Open Source Software", en línea en:  
<http://www.hecker.org/writings/setting-up-shop.html>.
- Mike Perry: "Open Source Licenses", en línea en:  
<http://fscked.org/writings/OpenSource.html>.
- Brian Behlendorf: "Open Source as Business Strategy", en *Open Sources*.



- Rex Brooks: Open Source Licenses Overview, en línea en:  
[http://www.vrml.org/TaskGroups/vrmlipr/open\\_source\\_overview.html](http://www.vrml.org/TaskGroups/vrmlipr/open_source_overview.html).
- Eric Kidd: "A History of Open Source" (19 de agosto de 2000), en línea en:  
[http://discuss.userland.com/msgReader\\$19844#19889](http://discuss.userland.com/msgReader$19844#19889).
- Estudio POSS / IDA (Unysis para la Unión Europea), pp. 60-65.
- The Mitre Corporation: "Use of Free and Open-Source Software (FOSS) in the U.S. Department of Defense", versión 1.2, 28 de octubre de 2002, p. 15.

Aunque cada iniciativa tendrá sus propios criterios para la elección de una licencia, algunos factores críticos podrían ser:

- a) El grado de independencia relativa al tiempo (versiones, compatibilidades futuras).
- b) El grado de independencia relativa a un producto (hay licencias para un software particular, como la licencia Python).
- c) El grado de independencia relativa a una empresa (por ejemplo, la Common License está muy vinculada a IBM).
- d) Si permite la distribución libre del código.
- e) Si permite la reingeniería del código.
- f) Si permite la creación y distribución de aplicaciones comerciales en binario.

### 8.1.2. Las licencias sobre las contribuciones y autoría

Un elemento esencial a tener en cuenta a la hora de considerar las licencias, es el tema de los autores y las contribuciones a un proyecto

de desarrollo libre. La historia de Netscape muestra las dificultades que uno puede encontrar si se decide a liberar el código o pasar de una licencia libre a otra.

Los problemas que se pueden presentar incluyen:

- a) Código que provenga de una fuente no segura (puede haber sido copiado).
- b) Código al que se ha contribuido bajo otra licencia (una licencia incompatible).
- c) Código restringido por obligaciones preexistentes, ya sea por las licencias de terceros o por compromisos que vinculan al autor-licenciante (el problema de Netscape).
- d) La revocación de la licencia del autor original al proyecto libre (no suele ser un problema, siempre que el proyecto haya declarado los requisitos legales para cualquier contribución).
- e) Restricciones en una licencia anterior que son incompatibles con una nueva licencia deseada. Aunque el autor siempre pueda licenciar de nuevo su código (excepto si se ha comprometido a no hacerlo), esto puede provocar problemas de división de código e incompatibilidad (el problema de UNIX: hay tantas versiones que es difícil saber cuál es la licencia o el origen de una parte de código).
- f) Patentes otorgadas sobre un elemento del código.

### Ejemplo

La FSF exige que cualquier programador que contribuya con más de 10 líneas de código a un proyecto GNU transfiera dicho código a la FSF.

Para los responsables de proyectos de código libre, es necesario realizar un seguimiento cuidadoso del código al que se ha contribuido, mantener un registro de autoría y de versiones o exigir que cualquier contribuyente otorgue una licencia total y exclusiva a la entidad que coordine el proyecto (en el derecho anglosajón, esto constituye una transferencia de titularidad, lo que no es posible en relación con los derechos morales, bajo derecho continental). De esta manera, el coordinador del proyecto puede mantener el control sobre la obra. Esto permite asegurarse que el código provenga de fuentes seguras (por ejemplo, que no sea copiado de otro software) e impedir que cualquier autor original pueda revocar su licencia o solicitar una patente sobre el software.

### 8.1.3. La compatibilidad entre licencias

Hemos hablado varias veces de la compatibilidad de código y de licencias.



Un programa es compatible con otro si se puede mezclar y distribuir sus códigos sin infringir las licencias de uno y de otro.

Por ejemplo, como la licencia BSD permite casi cualquier acción con el código bajo BSD, se puede mezclar o integrar código BSD con casi cualquier otro programa sin infringir las dos licencias correspondientes. El resultado estará regido por la licencia más restrictiva de las dos (la GPL, por ejemplo, o la MPL).

#### Reflexión

Ya hemos clasificado las licencias compatibles con la GPL, como criterio de análisis. En todos los casos de compatibilidad y mezcla de software en código GPL, el código resultante se distribuirá bajo la GPL (si no, no se podría mezclar con el código GPL, dado que sus obras derivadas tienen que distribuirse bajo la misma licencia).

Muchos exponentes del software libre recomiendan usar una licencia compatible con la GPL, sobre todo, porque ella rige casi el 85% del software libre disponible (no necesariamente los programas más usados, hay que tenerlo en cuenta), pero también porque se podrá recibir mayor apoyo de la comunidad de desarrollo libre. En esto hay cierta polémica, porque hay proyectos y desarrolladores que se niegan a aceptar código bajo GPL y otros que aceptan nada más que el código GPL o bajo una licencia compatible (la FSF, en particular).

Otro tema relevante es la compatibilidad por enlace: aun si no se permite mezclar o integrar códigos de diferentes licencias, a lo mejor se pueden enlazar. Ello puede hacerse de diferentes formas; por ejemplo, insertando una API entre un programa y otro o creando enlaces dinámicos que se activan en *run time*.

#### Ejemplo

Varios proyectos se han esforzado en hacerse compatibles con la GPL, por ejemplo Python, Qt y Vim, y hasta Mozilla agregó la cláusula adicional para permitir la licencia dual.

**Reflexión**

La MPL prevé expresamente que se pueda enlazar una aplicación propietaria con un programa bajo esta licencia. El programa propietario debe vincularse con el código bajo MPL a través de una nueva API dentro del programa bajo MPL. La API será una modificación y, por lo tanto, estará bajo MPL.

**8.1.4. Las licencias duales o múltiples**

El autor original no está restringido en las formas de licenciar su código, excepto si ha otorgado una licencia exclusiva o ha dado compromisos de confidencialidad. Por lo tanto, un programa se puede distribuir al mismo tiempo bajo una licencia libre y bajo una licencia propietaria o semipropietaria (de tipo *shared source*) u otra licencia libre.

**Ejemplo**

Hay varios programas que se distribuyen de esta manera. Por ejemplo:

- a) **MySQL**, un motor de base de datos, se distribuye bajo la GPL para usos personales y una licencia propietaria para integrar este programa en productos comerciales.
- b) **eZ publish** es un programa de gestión de contenidos en Internet que también tiene una licencia doble GPL/propietario.
- c) La **MPL** permite distribuir un programa bajo ambas MPL y GPL.

Desde el punto de vista del autor, la posibilidad de usar una segunda licencia propietaria depende del potencial de la comunidad libre para crear un producto similar (que se distribuirá bajo una licencia puramente libre). Asimismo, es esencial mantener un control muy estricto de los derechos sobre el código incorporado en el producto. Por ello, se recomienda:

- a) Usar una licencia *copyleft* para la licencia libre, que impida copiar el programa y privatizarlo para fines comerciales (en competencia con la licencia dual comercial).

- b) Concentrar los derechos de autor sobre el programa en las manos de una sola entidad (como lo hace la FSF, pero también las empresas MySQL AB y Tolltech). Esto incluye obligar a la transferencia exclusiva de título sobre el código (desde los contribuyentes originales), establecer licencias exclusivas con desarrolladores y socios comerciales y controlar el riesgo de patentes. También permite controlar el precio, la calidad y las responsabilidades sobre el código y, eventualmente, liberar el código totalmente.

Otro elemento de control sobre el software son las marcas, una herramienta legal utilizada por Apache pero también por Sun en relación con el uso de Java o Jini.



Desde el punto de vista de los usuarios finales, no hay diferencia: ambas licencias deben permitir un uso libre (notad que una licencia “libre” que no permita el uso comercial, no es libre según la definición OSD).

Los usuarios intermediarios (por ejemplo, empresas de desarrollo que quieran beneficiarse del código) pueden pasar de una licencia a la otra (propietaria), normalmente, pagando los cánones correspondientes.

#### **8.1.5. Cambio de una licencia a otra: problemas y consecuencias**

Finalmente, consideremos brevemente el caso del cambio de una licencia a otra (libre o propietaria). Ya hemos visto algunos aspectos de este tema, cuando comentamos la creación de la MPL y cuando hablamos de la elección de una licencia. En estos casos, hay que tener en cuenta algunas consideraciones:

- 1) En primer lugar, habrá que averiguar el origen y los derechos sobre todos los elementos del código involucrado: ¿quiénes son los autores?, ¿están disponibles por si hace falta pedirles su consentimiento para algo?, ¿qué derechos y obligaciones subsisten en el código?, ¿su licencia original permite cambios en la forma de distribución (recuérdese el *copyleft*)?, ¿hay alguna patente sobre el código?

**Experiencia**

Muchas veces se ha tratado de encontrar a autores contribuyentes de código libre para pedirles autorización para cambiar una licencia o integrar código en otro programa bajo otra licencia. Hay varias notas públicas en los archivos de Sourceforge y en otros foros de software libre donde los coordinadores de proyectos libres buscan a autores desconocidos para avisarles.

2) Segundo, hay riesgos y responsabilidades potenciales relativos a:

- Los contratos comerciales de la empresa con terceros relacionados con el código.
- La posibilidad de que el código demuestre que se han copiado algunos elementos de otro programa (con el código fuente abierto, será más fácil para terceros de reclamar).
- Cualquier patente que pueda existir sobre alguna parte del código.
- Cualquier infracción de los derechos de los autores contribuyentes originales.

3) Por último, hay que elegir la licencia libre y, para ello, remitimos al lector a los apartados anteriores.

### **8.1.6. Licencias libres y la división de software libre (forking)**

El concepto de **división (forking)** viene de la informática multitarea: alude a la división de una tarea o proceso en dos, de modo que, por ejemplo, una puede seguir activa mientras que la otra se detiene.



La **división o bifurcación** de un programa ocurre cuando éste se modifica y esta modificación se desarrolla de manera separada, con otro equipo de coordinación y se distribuye bajo otro nombre, quizás otra licencia.

A veces, la división se ha comparado con engendrar hijos, que luego se independizan. Ejemplos de ello son OpenBSD y NetBSD, divisiones de la UNIX BSD original.



“División [de MySQL] significa dividir el código fuente de la base de datos MySQL en un repositorio mantenido separadamente de tal manera que cualquier desarrollo sobre el código original requiera una operación manual para transferirse al software dividido, o que el software dividido empiece a tener funcionalidades que no estén presentes en el software original.”

Acuerdo de asociación de MySQL

#### Nota

El programa que más ha padecido este fenómeno es UNIX, del que surgieron casi 10 variantes. Algunas variantes de UNIX fueron creadas a medida que sus autores originales (AT&T y la Universidad de California, Berkeley) iban cediendo licencias libres sobre el programa que permitían crear una versión nueva: Unixware de Novell, Open Server SCO, Solaris de Oracle, AIX de IBM, etc. La historia de la división de UNIX se trata en profundidad en la unidad 10, debido a su interés relativo al caso SCO contra IBM y Red Hat.

La posibilidad de división es relevante para los desarrolladores porque provee una indicación de la posible evolución técnica y legal o comercial del software. Desde el punto de vista técnico, las versiones “divididas” tienden a ser incompatibles o no interoperables con los programas originales. Desde el punto de vista legal y comercial, dichas versiones pueden erigirse como competencia del producto original y distribuirse bajo una licencia diferente (libre o comercial), lo que fomenta una incompatibilidad legal.

Hay varias causas de división de un código libre. La razón principal reside en la naturaleza esencial del software libre: los programadores “dividen” el código porque lo pueden hacer. Por ejemplo, la licencia

BSD permite crear una obra derivada del software original y privatizar el código modificado. Este nuevo programa podría no ser más que una variante del original (por ejemplo, OpenBSD en relación con NetBSD) y, por lo tanto, se considerará una “división”. Por ejemplo, una de las versiones principales de UNIX se distribuye bajo la licencia BSD. De esta versión, se han creado nuevas versiones de este sistema operativo: algunas se ejecutan sobre otros tipos de hardware (chips), otras se han desarrollado para crear versiones distribuidas de manera propietaria.

Otra razón para la división, reside en la gestión del equipo de desarrollo y desacuerdos entre los programadores. Por ejemplo, si se identifica la necesidad de una extensión o módulo nuevo y el coordinador no está de acuerdo, es muy probable que alguien cree una versión bifurcada para integrar ese módulo.

Las licencias tienen una influencia directa sobre la posibilidad de división:

- a) No se puede dividir un código bajo licencia propietaria o el *shareware* (no hay acceso al código fuente).
- b) El código bajo una licencia que no permite el uso comercial y exige devolver las modificaciones al autor original tampoco se dividirá, a causa del control centralizador que ejerce el autor (por ejemplo, Ghostscript y la licencia Aladdin).
- c) El código bajo GPL no tenderá a dividirse: hay que mantener la libertad de las obras derivadas y el software original puede por lo tanto incorporar cualquier mejora y hacerla suya.
- d) El código bajo licencias “intermedias” como la Artistic, la MPL o la LGPL podrá dividirse, porque permite crear variantes (propietarias o libres) por “agregación”.
- e) El código bajo licencia BSD y similares se dividirá con facilidad, porque permite distribuciones binarias sin obligación de publicar el código fuente.



Podemos concluir que, aunque una licencia con *copyleft* robusto otorgue mayores libertades a los usuarios finales,



al fin y al cabo, permite mantener un mayor control sobre la evolución de un programa, y evitar divisiones y obras derivadas competitivas.

### 8.1.7. Resumen de potenciales problemas legales de las licencias libres

Ya hemos considerado los distintos elementos de una licencia de software y su ajuste al marco legal, desde los puntos de vista del derecho contractual, de la propiedad intelectual (derechos de autor) e industrial (patentes) y de la protección del usuario en general y del consumidor en particular (garantías, cláusulas abusivas).

En este estudio particular de las consecuencias de las licencias libres, queremos resumir los problemas o dificultades mayores que pueden surgir a la hora de utilizar una licencia libre como licenciante o como licenciatario:

- a) **Naturaleza como licencia o como contrato.** Hay cierta confusión sobre la naturaleza legal de la licencia de software libre. Básicamente, se debe considerar que una licencia no es un contrato hasta que el usuario-contratante lo haya aceptado explícitamente. Hasta ese momento, cualquier obligación sobre el licenciatario es vinculante únicamente si corresponde a los derechos exclusivos reservados por el derecho de la propiedad intelectual e industrial y es razonable. Para ser vinculantes, las condiciones de una licencia tienen que ser “razonables” o corresponder a la buena fe del licenciante (lo que no es necesario para un contrato, en muchos casos, puesto que las condiciones pueden ir más allá de lo razonable). Hay pocas indicaciones sobre lo que es razonable en el ámbito del software libre.

#### Reflexión

La GPL, por ejemplo, reconoce esto y no se considera como contrato hasta la modificación o redistribución del código, considerando dichos actos como la aceptación explícita de las obligaciones.

#### Nota

Ver los distintos elementos de una licencia de software y su ajuste al marco legal en las unidades 4, 5 y 6.

#### Nota

Obsérvese que se trata de un resumen, por lo que, para un estudio más profundo, remitimos al lector a las unidades correspondientes.

- b) **Sanciones en caso de violación de una simple licencia (no de un contrato).** Las sanciones y penalidades por violación de derechos de autor pueden ser diferentes de las aplicadas al incumplimiento contractual. En el derecho anglosajón, por ejemplo, el incumplimiento contractual suele dar lugar únicamente a una indemnización monetaria. Por ello, las demandas relativas a las licencias suelen agregar la violación de derechos de autor, de confidencialidad o de competencia desleal, para permitir la devolución del software y el cese de las actividades ilegales.
- c) **Revocación.** Si el usuario-licenciante no es contratante, el autor puede revocar la licencia en cualquier momento, previo aviso. Asimismo, la **revocación** permite al autor modificar las condiciones de explotación del software, por ejemplo: privatizarlo. Más aún, podría modificar las condiciones con efecto retroactivo. En consecuencia, para cualquier usuario final o intermedio, sería mejor tener un contrato vinculante.
- d) **Aceptación *click-wrap* o similar.** Hay dudas acerca de si este tipo de aceptación es válida o no, y si lo fuera, cuáles son las condiciones incorporadas en el contrato, tomando en cuenta el momento de la notificación de sus términos al contratante (debe ser antes del *click*).
- e) **Obligaciones sobre terceros.** No es seguro que el mecanismo de transmisión y persistencia de la licencia funcione perfectamente, aunque nadie se ha arriesgado a probarlo. Se presume que cualquier sublicenciatario es un nuevo licenciatario directo del autor, sin embargo, no hay ninguna relación fáctica entre estas personas. Quizás, se podría considerar esta persistencia como una renovación del contrato inicial.
- f) **Exclusión de garantías y responsabilidades:** es casi seguro que estas cláusulas no sean válidas, sobre todo en el marco legal europeo. Por un lado, cualquier exclusión de responsabilidad ha de ser razonable para ser vinculante y, por otro lado, el usuario-contratante podría beneficiarse de garantías implícitas provistas por la ley. Asimismo, podrá beneficiarse de indemnizaciones bajo responsabilidades civiles (*tort*, en el derecho anglosajón), sobre todo, por negligencia en la programación. Junto con ello, hay que precisar que varios distribuidores de software libre, como Red Hat, proveen garantías y tienen seguros comerciales contra riesgos de este tipo.
- g) **Autores y distribuidores:** cuando surja una responsabilidad legal, determinar o encontrar a la persona responsable en el entor-

no de desarrollo libre (cuando hay varios autores) puede ser muy difícil. Asimismo, aunque se pueda determinar el responsable, es poco probable que tenga fondos para una indemnización adecuada o un seguro al respecto. Por otro lado, en el contexto del mundo digital, mantener pruebas fehacientes de la proveniencia de un software libre es difícil, aunque los avisos de *copyright* de autoría y de modificación pueden ayudar. Asimismo, algunos sitios de distribución de software libre firman digitalmente el software para garantizar su integridad y autenticidad.

- h) **Derecho aplicable y jurisdicción:** una licencia que no tenga una cláusula de derecho aplicable y una jurisdicción competente causa una cierta incertidumbre acerca del foro pertinente y el régimen legal que se le ha de aplicar en el caso de cualquier litigio sobre el software. Mientras que los consumidores pueden beneficiarse de su propia jurisdicción y las protecciones locales, éste no es el caso de los usuarios comerciales. Además, las licencias se adecuan más al régimen legal americano y algunos conceptos pueden causar dificultades en otras jurisdicciones. Otra dificultad suscitada por la multiplicidad de regímenes legales es la cuestión del derecho que ha de aplicarse a las contribuciones de autores que no están dentro de la jurisdicción americana. Un ejemplo son los derechos morales, que no son reconocidos en EE.UU. o Inglaterra, pero sí en Europa continental.
- i) **Responsabilidades del proveedor de servicios de la sociedad de la información:** gran parte del software libre se distribuye a través de Internet y muchas organizaciones de distribución serán consideradas proveedores de servicios sujetos a las obligaciones de las leyes, implementando la Directiva del Comercio Electrónico y otras. Entre ellas, figuran obligaciones de información y de proceso de contratación que los sitios de distribución habrían de cumplir (por ejemplo, la presentación de los términos de contratación antes de bajar el programa).

## 8.2. Licencias libres y otras ramas de derecho

El estudio de las licencias libres se ha centrado, sobre todo, en el derecho de la propiedad intelectual y en algunos aspectos contractuales. En este apartado queremos presentar y comentar algunas dimensiones de

### Nota

Podéis ver el estudio de las licencias libres en la unidad 7.

estas licencias en relación con otras ramas del derecho: las atinentes a la competencia, a las patentes y a las marcas (propiedad industrial), al secreto comercial y a los estándares.

### 8.2.1. Licencias libres y el derecho de la competencia

Antes de asumir nuevas políticas más abiertas frente al software libre, algunas empresas grandes de software alegaron que el software libre era “anticompetitivo”. Argumentaban que las licencias libres obligan a las empresas a revelar sus secretos confidenciales, que la licencia GPL “infecta” a cualquier código privado y obliga a distribuirlo bajo la misma licencia (sin que el propietario pueda tener beneficios) y que el software libre es una amenaza para las empresas privadas que sustentan sus beneficios en el modelo tradicional de desarrollo (regalías o cánones pagados por la licencia, remunerando la labor de desarrollo y no únicamente el coste de distribución física).

Es importante comentar este tema de la competencia. El derecho de la competencia restringe las actividades anticompetitivas de una o más empresas, sobre todo, con el objetivo de proteger a clientes y consumidores. Brevemente, podemos afirmar que dos tipos de actividades son ilegales:

- Los acuerdos entre empresas que tienen el efecto de distorsionar el comercio.
- El abuso de una posición dominante en un mercado, por parte de una o más empresas, que reduzca la competencia en el mercado.

Estas reglas se aplican también a las administraciones públicas en sus relaciones con empresas públicas y a otras instituciones con derechos especiales acordados por el gobierno.



El marco legal formal del derecho de la competencia incluye:

- A nivel **internacional**, las directrices de la OECD, las reglas de la OMC y otras organizaciones económicas internacionales.

- A nivel **regional**, en la Unión Europea, los artículos 81 y 82 del Tratado de la Unión Europea.
- A nivel **nacional**, cada país miembro de la Unión Europea ha implementado versiones de estos dos artículos en el régimen jurídico nacional. Fuera de Europa, muchos países incorporan reglas similares (sobre todo en EE.UU., pero también en América Latina, Japón, etc.).



¿Qué tiene que ver esto con el software libre? Hay varios vínculos entre las licencias libres y el derecho de la competencia. La cuestión principal es determinar si una licencia libre puede considerarse una práctica anticompetitiva. Hay que tener en cuenta que una licencia es un acuerdo “vertical” entre empresas o personas. Como el licenciante está en una posición más fuerte, podría intentar imponer restricciones sobre el licenciatario que tengan un efecto anticompetitivo.

Respecto de los acuerdos comerciales entre empresas, las siguientes cláusulas en una licencia de software pueden ser declaradas ilegales:

- a) Un pacto que prohíba la descompilación o la corrección de errores, si tiene el efecto de distorsionar la competencia (por ejemplo, impedir la provisión de servicios de soporte y mantenimiento).
- b) Pactos que obliguen a comprar un tipo de software vinculado con un hardware.

En relación con el abuso de una posición dominante, los siguientes términos en una licencia de software pueden tener efectos anticompetitivos:

- a) Una cláusula que obligue a devolver al autor inicial cualquier corrección o modificación del código.

- b) Una cláusula que vincule el software con la compra de otro producto.
- c) Una cláusula que prohíba el uso de otro tipo de software.
- d) La negación de proveer una licencia a competidores o de revelar detalles de una interfaz.
- e) La restricción del uso de un software a un solo equipo.

Ya hemos visto que este tipo de cláusula no existe en las licencias libres. Por el contrario, la mayoría de estas cláusulas figurarían casi exclusivamente en licencias propietarias. Sin embargo, en la licencia seudolibre de Apple (APSL 1.0) existía una cláusula que ilustra el primer caso de la lista –la obligación de devolver las correcciones al autor original–, y la NPL permite a Netscape beneficiarse comercialmente de los desarrollos de contribuyentes de código al software inicial de Netscape (Navigator).

Asimismo, las reglas contra acuerdos anticompetitivos entre empresas y contra el abuso de posiciones dominantes incluyen cualquier acuerdo que aplique a terceros (contratantes) condiciones desiguales para prestaciones equivalentes. Esto puede afectar las licitaciones públicas de los gobiernos, si intentaran imponer un tipo u otro de software (propietario o libre) o, más aún, si indicaran una preferencia por una licencia en particular. Si impusieran una licencia libre, podría considerarse como un término discriminatorio contra empresas de software propietario.

Sin embargo, en dichos casos, los gobiernos podrían beneficiarse de excepciones previstas por la ley y argumentar que el software libre favorece el progreso tecnológico, así como una participación más equitativa de los consumidores en el beneficio. Esta excepción, por ejemplo, es uno de los fundamentos del Sexto Programa de Investigación RTD de la Unión Europea, que favorece la creación de software libre y el uso de licencias libres para la distribución de los resultados de investigación y desarrollo.

#### Reflexión

Como hemos visto en los últimos 15 años a raíz de los problemas legales de Microsoft ante las autoridades res-

ponsables de la competencia, la relación entre el derecho de la competencia y la propiedad intelectual (incluso las licencias) es muy compleja y discutida. A pesar de ello, no hay que confundir las prácticas anticompetitivas como las restricciones y los abusos, y la competencia entre dos modelos de desarrollo diferentes.

### 8.2.2. Licencias libres y licencias de patente

Hemos visto que una patente sobre un proceso o código informático impide no solamente su copia o modificación, tal como hace el derecho de autor, sino también su recreación o reingeniería (lo que los americanos llaman *clean-room development*, es decir, un desarrollo sin acceso a versiones del código original) y el uso y comercialización de la idea o del proceso patentado. Por lo tanto, una patente impediría la expansión del software libre en “clones” de aplicaciones propietarias (como Openoffice.org frente a MSOffice, MySQL frente a Oracle, etc.), las mejoras sobre cualquier código patentado o la reutilización de componentes estándares que fueran protegidos por patentes.

Para minimizar el riesgo que suponen las patentes, las licencias libres más modernas (MPL, IBM Common License, etc.) incluyen cláusulas que otorgan licencias cruzadas de patentes entre contribuyentes y desarrolladores, pactos resolutorios para el caso en que se usara una patente para iniciar un litigio con otro contribuyente, y reservas de distribución en relación con países donde el uso del software podría infringir una patente. Esta práctica es muy común en el mundo del desarrollo propietario.

Es importante considerar el alcance de la licencia de patente otorgada, que debe cubrir todos los actos reservados a sus titulares. Este alcance es muy amplio, e incluye la fabricación, el ofrecimiento, la puesta en el mercado, y la utilización y la importación o la posesión del objeto o procedimiento patentado, o fruto del procedimiento.

Sin embargo, estas cláusulas pueden plantear problemas. En algunas jurisdicciones no se pueden limitar u obligar ciertos actos futuros.

#### Nota

Podéis ver las patentes en la unidad 3.

Por lo tanto, las cláusulas que obligan a licenciar una eventual patente pueden no ser válidas. Para evitar el uso de patentes, quizás sería mejor usar la limitación general de la GPL, que impide imponer sobre cualquier distribución o modificación restricciones mayores que las incluidas en la misma GPL. Esto tiene el efecto práctico de impedir el uso de patentes o, si se obtiene una patente sobre un proceso incorporado en el software o una obra derivada, se deberá otorgar una licencia en términos no más restrictivos que la GPL.

### 8.2.3. Licencias libres y secreto comercial

Otra manera de proteger el software es bajo el derecho del secreto comercial. Esto permite proteger secretos de negocio e información de las actividades comerciales. Los requisitos básicos son:

- a) La información debe tener una cualidad de secreto. Los secretos de negocio se extienden a los algoritmos y a otra información incorporada en el software, tal como los procesos de negocio, las especificaciones técnicas y la arquitectura técnica.
- b) Debe haber una obligación de confidencialidad y secreto (explícita o implícita), sobre todo, por parte de los empleados y los consultores respecto de las empresas contratantes y entre socios de negocio, en las cláusulas de confidencialidad habituales.
- c) Debe haber una revelación o comunicación no autorizada de la información. Esto puede ocurrir a través de la publicación del código fuente de un programa libre.

Las consecuencias de la violación del secreto comercial pueden ser:

- a) Una indemnización a la empresa perjudicada relacionada con el precio de la licencia que hubiera podido cobrar (¿pero quién pagará, aparte del “revelador”, si se puede determinar que fue él?).
- b) La devolución de cualquier beneficio realizado sobre la base del secreto, en el caso de una violación intencional.
- c) La obligación de no usar la información revelada, es decir, el programa en cuestión y, eventualmente, su destrucción.



- d) En algunos países, la información puede considerarse una propiedad que se puede robar, en cuyo caso la revelación intencional de un secreto puede constituir un delito con sanciones penales (para el que comunica el secreto, no para su destinatario).

En este ámbito del derecho surgen varios temas en relación con las licencias y el software libres, por las siguientes razones. Primero, un desarrollador de software libre puede revelar (intencional o negligentemente), a través de su programación, los secretos de negocio de la empresa en que trabaja o ha trabajado (y con la cual sigue relacionado por obligaciones contractuales de confidencialidad). Esto afectará al uso del programa que resulta del desarrollo. Sin embargo, es muy difícil separar “conocimientos del negocio”, que están efectivamente protegidos por el secreto comercial, de las “competencias” que son propias del programador y que él mismo puede usar libremente en el ejercicio de su profesión u otras actividades.

Otra fuente de problemas e incumplimiento respecto del secreto comercial puede surgir del uso por parte de desarrolladores libres de información obtenida a partir de una relación comercial, por ejemplo, relativa a un software propietario comprado a un proveedor, cuyo uso y ejecución estará bajo obligaciones de confidencialidad (relativas al diseño, las especificaciones, la arquitectura, etc.).



Luego, el secreto comercial puede suscitar responsabilidades relativas al origen del código, cualquier uso del software en cuestión y de la distribución de código fuente, abiertamente legible por terceros. Hay que considerar el efecto de las cláusulas de exclusión de responsabilidades contenidas en las licencias libres en relación con esto. Desafortunadamente, no se sabe si estas exclusiones son válidas en este contexto.

#### Ejemplo

SCO ha denunciado a IBM por haber incorporado sus “secretos de negocio” en la última versión de Linux, la versión 2.2. SCO alega que dichos secretos fueron comunicados a IBM en el sistema operativo SCO UNIX

bajo obligaciones contractuales de confidencialidad. IBM argumenta que (1) no hay secretos para revelar, ya que los procesos del software provisto por SCO están abiertamente disponibles a través del código libre de UNIX y (2) no hubo ninguna “incorporación” tal y como alega SCO por las separaciones internas de las divisiones de IBM. Es decir, IBM dice que no hay ningún secreto protegible y, si lo hubiera, no hubo ningún contacto entre el equipo de AIX que trabajaba con el software de SCO y el equipo “libre” de IBM trabajando sobre Linux.

#### Nota

Podéis ver las marcas en la unidad 3.

#### 8.2.4. Licencias y marcas

Otra rama del derecho que puede relacionarse con el software libre y las licencias es el derecho de las marcas. Ya hemos visto que este derecho protege las marcas registradas que indican la conexión entre una persona o empresa y sus bienes y servicios, y los diferencian de los productos de otra persona.



Como los avisos de autoría, las marcas contribuyen a proteger la calidad del software original y, por lo tanto, la reputación de los autores iniciales.

Por ejemplo, Linus Torvalds ha registrado la marca Linux<sup>®</sup> en relación con el software que conocemos como Linux. Nadie más puede usar esta marca sin el consentimiento de L. Torvalds. El registro de una marca sobre un software libre sirve para sostener la autoría y gestionar la evolución del software, independientemente de los permisos acordados por la licencia.

Es importante considerar que la marca registrada es un bien intangible y un derecho separado del software y que cualquier licencia debe incluir pactos al respecto si un distribuidor intermediario quiere usar dicha marca. Las cláusulas suelen establecer un derecho o la prohibición de uso, quizá bajo ciertas condiciones (por ejemplo, si el software ha sido modificado).

El modelo de distribución y comercialización del software libre depende en gran medida de las marcas (Red Hat, SuSe, Mandrake, Zope, Apache, Java, etc.) y las licencias libres incluyen términos precisos sobre su uso. Hemos visto que algunas licencias obligan a los usuarios a indicar la autoría y la marca (Apache), otras lo prohíben sin consentimiento expreso del titular de la marca (Zope), otras impiden su uso sobre obras derivadas (la mayoría, incluso la BSD). Se habrá de consultar la licencia a la hora de indicar públicamente el software utilizado en alguna aplicación ("basada en Linux", "programada con PHP", "con tecnología Apache", "extensiones para MySQL").

Obsérvese que las licencias libres no son tan completas como las licencias comerciales de marca, en cuanto al uso de la marca, su presentación, el tamaño del logotipo, las condiciones de uso, territorios, productos y servicios relacionados, etc. Serán los contratos accesorios de comercialización (o principales, según el servicio proporcionado) de los proveedores y consultores de software libre los que incluirán pactos mayores sobre las marcas registradas. Algunos ejemplos incluyen el uso de la marca de Sun<sup>®</sup> o Java<sup>®</sup>, o Red Hat<sup>®</sup>. Este es un aspecto muy relevante para cualquier persona interesada en la comercialización o la publicación de software libre.

#### 8.2.5. Licencias y estándares

Un último asunto por comentar en relación con las licencias libres es el tema de los estándares. El software libre se basa mucho en estándares abiertos, que permiten la interoperabilidad e interconexión de los sistemas informáticos. Pensamos en los protocolos de Internet (HTTP, TCP/IP, etc.), las interfaces para periféricos (*drivers*, etc.) y los algoritmos de cifrado (hash MD-5, SSH, etc).



Mantener la apertura de los estándares es muy importante para el software libre.

Hay una tensión entre la estandarización y la propiedad intelectual e industrial, que son casi opuestas. Ha habido muchas denuncias de las tentativas de patentar o mantener derechos exclusivos de autor

sobre tecnologías aceptadas como estándares por la W3C. Algunos actores comerciales hablan de licenciar bajo términos “razonables y no discriminatorios” (RAND) cualquier tecnología estándar que tengan. Pero RAND no tiene sentido para un desarrollador libre independiente.

Se ha observado que el sistema de desarrollo libre es un proceso de estandarización no formal y que, al final de proceso, el software se erige en estándar para el sector. SendMail, BIND, SMTP, Kerberos, etc. son programas libres cuya especificación ha resultado en un estándar para la industria de Internet. Asimismo, el desarrollo libre es un proceso más rápido que el proceso formal de estandarización, lo que ha fomentado, por ejemplo, un alto nivel de interoperabilidad, en el contexto de Internet.

Las licencias libres favorecen los estándares abiertos. Ya hemos observado que las licencias más flexibles, como la LGPL o la BSD, permiten la difusión rápida de una tecnología y su adopción como estándar tanto en sectores propietarios como libres. Desafortunadamente, también permiten privatizar extensiones que pueden aplicarse o agregarse a los estándares y hacerles no interoperables con el software original (por ejemplo, la versión Microsoft de Kerberos, que se estudiará en la unidad 10).

#### Nota

Los estándares van a ser otro campo de batalla del software libre, junto con las patentes y la extensión del uso propietario de derechos de autor, como en el caso de Kerberos y en el de la iniciativa de Trusted Computing, ambos comentados en la unidad 10.

### 8.3. Los datos personales y la protección de la intimidad

Las nuevas tecnologías de la información y la comunicación (TIC) han causado muchos cambios importantes en el orden jurídico a nivel nacional e internacional. Entre otros, cabe destacar el nuevo régimen legal de la protección de datos personales y la protección de

la intimidad, vinculado con derechos fundamentales a la intimidad (en España, reconocidos en el artículo 18.1º de la Constitución). En este apartado, vamos a considerar la relación entre el software libre y la protección de datos personales. Primero introducimos el concepto de privacidad y el marco legal de la misma, luego comentamos cómo el software libre puede estar afectado por el derecho de la privacidad, y viceversa, cómo los sistemas comerciales y públicos pueden aprovechar el software libre para cumplir con el mismo.

### 8.3.1. Introducción y marco legal

La protección legal de la intimidad y de la vida privada no es tan nueva: ya en los años 70 se desarrollaron varias iniciativas para definir un sistema de protección de la privacidad que dieron lugar a la aprobación de varios textos legales en diferentes países europeos:

- a) En el ámbito del Consejo de Europa, diversos trabajos resultaron en el **Convenio 108**, de 28 de enero de 1981.
- b) En España, en el año 1992, amparándose en el artículo 18.4º de la Constitución, se aprobó la Ley Orgánica 5/1992, de Regulación del Tratamiento Automatizado de Datos Personales (**LORTAD**).
- c) A nivel de la Unión Europea, en 1995 se aprobó la Directiva 95/46, relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos (la **Directiva**).

Actualmente, el régimen español ha sido actualizado para implementar la Directiva y rige la Ley Orgánica 15/1999, de Protección de Datos de Carácter Personal (**LOPD**), junto con otras disposiciones que desarrollan aspectos particulares, como el Real Decreto 1332/1994 o el Real Decreto 994/1999 (por el que se aprueba el Reglamento de Medidas de Seguridad).



Según el artículo 1 de la LOPD, ésta tiene por objeto garantizar y proteger los datos personales, las liberta-

des públicas y los derechos fundamentales de las personas físicas, especialmente su honor e intimidad personal y familiar.

Básicamente, todo fichero que almacene datos personales de personas físicas identificadas o identificables se encuentra dentro del ámbito de aplicación de la ley y está sometido, por tanto, a los principios y requisitos y al régimen establecida por ella. Veamos brevemente estos principios y requisitos.

### **8.3.2. El régimen legal de protección de los datos personales**

Los **conceptos básicos** que son fundamentales para entender el marco legal de la privacidad son los siguientes:

- a) **Datos personales:** cualquier información concerniente a personas físicas identificadas o identificables. Se consideran datos especialmente protegidos los relativos a la ideología, afiliación, religión y creencias, los que hagan referencia al origen racial, a la salud y a la vida sexual, así como los relativos a la comisión de infracciones penales o administrativas.
- b) **Ficheros:** todo conjunto organizado de datos de carácter personal, cualquiera que fuere la forma o modalidad de su creación, almacenamiento, organización y acceso. Se extiende a ficheros no automatizados y a todo tipo de dato personal susceptible de tratamiento. Es suficiente que los mismos estén estructurados u organizados con arreglo a algún criterio lógico que permita la inclusión de dichos datos en un fichero.
- c) **Tratamiento:** cualquier tipo de operaciones y procedimientos técnicos, de carácter automatizado o no, que permitan la recogida, grabación, conservación, elaboración, modificación, bloqueo y cancelación, así como las cesiones de datos que resulten de comunicaciones, consultas, interconexiones y transferencias.
- d) **Interesado:** la persona física titular de los datos que son objeto de tratamiento. Las personas jurídicas están excluidas de protección.

- e) **Responsable de fichero:** la persona física o jurídica, autoridad pública, servicio o cualquier otro organismo que decida sobre la finalidad, contenido y uso del tratamiento (qué, quién, cómo, cuándo y dónde). El responsable del fichero responde administrativa, civil y penalmente, por las posibles infracciones a la LOPD.
- f) **Encargados de tratamiento:** la persona que, sola o conjuntamente con otros, trate datos por cuenta del responsable del tratamiento.
- g) **Responsable de seguridad:** la persona o personas a las que el responsable del fichero ha asignado formalmente la función de coordinar y controlar las medidas de seguridad.
- h) **La Agencia de Protección de Datos (APD):** se establece una autoridad nacional con poderes de sanción para garantizar la protección de datos personales y mantener los registros de ficheros notificados.

**Ejemplo**

Como ejemplos de ficheros con datos personales, podemos citar cualquier conjunto de datos como el historial de los pacientes de un médico (a condición que estén ordenados según un criterio lógico) o como el perfil de los usuarios de un sitio web (clientes, personas registradas, etc.). No importa si el "soporte" de los datos tiene un formato físico o electrónico y tampoco es relevante (en principio) si son o no objeto de un tratamiento automatizado.

El **ámbito objetivo** de aplicación de la LOPD son los datos de carácter personal registrados en soporte físico que los haga susceptibles de tratamiento y a toda modalidad de uso posterior de estos datos por los sectores público y privado. Están excluidos de protección:

- Los ficheros mantenidos por personas físicas en el ejercicio de actividades exclusivamente personales o domésticas (por ejemplo, una agenda personal).
- Los ficheros sometidos a la normativa sobre protección de materias clasificadas (datos relativos a la defensa nacional y protección del Estado) o establecidos para la investigación del terrorismo y de formas graves de delincuencia organizada.

La LOPD se aplica al **ámbito especial** de los ficheros siguientes:

- Cuando el tratamiento sea efectuado en territorio español en el marco de las actividades de un establecimiento del responsable del tratamiento.
- Cuando al responsable del tratamiento no establecido en territorio español, le sea de aplicación la legislación española en aplicación de normas de derecho internacional público.
- Cuando el responsable de tratamiento no esté establecido en territorio de la Unión Europea y utilice en el tratamiento de los datos medios situados en territorio español, salvo que tales medios se utilicen únicamente con fines de tránsito.

### Principios generales

La ley establece varios **principios generales** que se deban respetar. Son los siguientes:

#### 1) La calidad de los datos y la finalidad (art. 4)

Los datos de carácter personal sólo se podrán recoger para su tratamiento, así como someterlos a dicho tratamiento, cuando sean adecuados, pertinentes y no excesivos, en relación con el ámbito y las finalidades determinadas, explícitas y legítimas para las que se hayan obtenido. Estos criterios se determinarán en función del caso en concreto:

- a) Los datos de carácter personal no podrán usarse para finalidades incompatibles con aquellas para las que los datos se hubieran recogido. No se considerará incompatible el tratamiento con fines históricos, estadísticos o científicos. Cuando se ha cumplido la finalidad para la que se recabaron los datos, éstos han de ser cancelados o destruidos o, en caso de que ello no sea posible, bloqueados.
- b) No serán conservados en forma que permita la identificación del interesado durante un periodo superior al necesario para los fines en base a los cuales hubieran sido recabados o registrados.



- c) Asimismo, los datos deben ser actualizados y exactos, de manera que si resultan ser inexactos o incompletos, en todo o en parte, han de ser cancelados o sustituidos.

## 2) Información en la recogida de los datos (art. 5)

Los interesados a los que se soliciten datos personales deberán ser previamente informados de modo expreso, preciso e inequívoco:

- a) De que sus datos van a ser incluidos en un fichero, de la finalidad de la recogida y de los destinatarios de la información.
- b) De la obligatoriedad o no de dar esos datos.
- c) De las consecuencias de la obtención de los datos o de la negativa a suministrarlos.
- d) De la posibilidad de ejercer los derechos de acceso, rectificación, cancelación y oposición.
- e) De la identidad y dirección del responsable del tratamiento o, en su caso, de su representante, para que los afectados puedan ejercer sus derechos.

Se prohíbe la recogida de datos por medio de medios fraudulentos, desleales o ilícitos.

## 3) Consentimiento del interesado (art. 6)

El tratamiento de los datos de carácter personal requerirá el consentimiento inequívoco, expreso o tácito, del interesado, salvo que la ley disponga otra cosa (es decir, una autorización legal, por ejemplo, por orden judicial). El tratamiento de los datos especialmente protegidos requiere el consentimiento expreso y por escrito.

No requieren consentimiento:

- Datos recogidos en fuentes accesibles al público (censo promocional, repertorios telefónicos, listas profesionales).

- Datos que se recojan para el ejercicio de las funciones propias de las administraciones públicas.
- Datos en contratos comerciales o laborales.
- Datos recogidos con la finalidad de proteger un interés vital del interesado.

En estos casos, el interesado podrá oponerse a su tratamiento cuando existan motivos fundados y legítimos relativos a una concreta situación personal.

#### 4) Secreto

Tanto el responsable del fichero como el encargado de tratamiento, así como cualquier persona que intervenga en cualquier fase del proceso, están obligados al secreto profesional respecto de los mismos y al deber de guardarlos. Estas obligaciones subsistirán aun después de finalizar sus relaciones con el titular del fichero o, en su caso, con el responsable del mismo.

#### 5) Comunicación y acceso a los datos

Sujeto a varias excepciones, cualquier comunicación o cesión de los datos a terceros requiere la autorización previa del interesado.

### Derechos y obligaciones

Como consecuencia de estos principios y otras disposiciones de la ley, el interesado se beneficia de varios **derechos** y el responsable está sujeto a una serie de **obligaciones**, que detallamos a continuación:

#### 1) Derechos de los interesados

- a) **El derecho a recibir la información** antes mencionada en el momento de la recogida de datos personales
- b) **El derecho de acceso a los datos**. El interesado puede obtener información sobre sus datos sometidos a tratamiento, el origen de los mismos y las cesiones o comunicaciones realizadas o que se prevean realizar.

- c) **Los derechos de rectificación y cancelación.** El interesado puede exigir que el responsable del fichero cumpla la obligación de mantener la exactitud de los datos, rectificando o cancelando los datos cuando resulten incompletos o inexactos, inadecuados o excesivos para la finalidad de la recogida o cuyo tratamiento no se ajuste a la Ley. En caso de que esta cancelación no sea posible, se procederá al bloqueo del fichero.
- d) **El derecho de oposición.** Cuando no es necesario prestar consentimiento para el tratamiento de los datos de carácter personal (y siempre que una ley no disponga lo contrario), el interesado podrá oponerse a su tratamiento cuando existan motivos fundados y legítimos relativos a una concreta situación personal. En este caso, el responsable del fichero debe excluir del tratamiento los datos relativos al afectado.
- e) **El derecho de impugnación.** El interesado puede impugnar cualquier acto administrativo o decisión privada con efectos jurídicos que implique una valoración de su comportamiento sobre la base de un tratamiento automatizado de datos personales que ofrezca una definición de sus características o personalidad.
- f) **El derecho a consultar el registro** general de la APD (un acceso público y gratuito).

## 2) Obligaciones

El responsable de tratamiento está sujeto a diversas obligaciones:

- a) **La notificación e inscripción registral:** inscripción de los ficheros en el Registro General de la Protección de Datos, con carácter previo a la creación del fichero.
- b) **La información al interesado:** ofrecer la información antes mencionada al interesado cuando se recojan sus datos y revisar los formularios, leyendas y procedimientos que se empleen en la recogida de los datos.
- c) **La obtención del consentimiento del interesado o una autorización legal** (previos) para el tratamiento.
- d) **Asegurar los procedimientos** para permitir a los interesados el ejercicio de sus derechos (principalmente acceso, rectificación y cancelación).

- e) **Documentación:** si en el tratamiento de los datos intervienen terceros (proveedores de alojamiento o empresas de tratamiento – *outsourcing*), se deben documentar estas relaciones en un contrato en el que deben figurar ciertas condiciones mínimas (seguridad, procesos, fines permitidos, accesos, etc.). En particular, la **designación del encargado de tratamiento** deberá realizarse por medio de contrato, que deberá formalizarse por escrito o en cualquier otra forma que permita acreditar la celebración y contenido del mismo.
- f) **Obligaciones de seguridad:** el responsable del fichero debe implementar las medidas de seguridad de índole técnica y organizativas necesarias para garantizar la seguridad de los datos objeto de tratamiento. Esta obligación comporta varios elementos:
- Redactar un documento de seguridad.
  - Realizar una auditoría.
  - Nombrar un responsable de seguridad en relación con un tratamiento de datos que requieran medidas de seguridad de nivel medio o alto.

### Lectura complementaria

En la bibliografía, hay varias referencias que aportan más datos sobre la protección de datos personales.

El **encargado de tratamiento** también debe cumplir con las obligaciones incorporadas en el contrato de tratamiento: desarrollará su actividad por cuenta del responsable del fichero y tratará los datos conforme a las instrucciones que haya recibido. En caso de incumplir las obligaciones que sobre él hace recaer la LOPD, responderá por las infracciones cometidas personalmente.

### Cesiones y transparencias

Una **cesión de datos** es cualquier revelación de datos a una persona distinta del interesado. Esto no incluye el acceso a los datos por un tercero, que sea necesario para la prestación de un servicio al responsable del tratamiento (por ejemplo, el encargado de tratamiento). Los datos de carácter personal sólo podrán ser comunicados a un tercero:

- 1) para el cumplimiento de fines directamente relacionados con las funciones legítimas del cedente y del cesionario; y

- 2) con el previo consentimiento informado del interesado. En relación con esta obligación, hay varias excepciones (obligación o autorización legal, datos públicos, relación jurídica legítima en función de la finalidad, autoridades judiciales, fines históricos, estadísticos o científicos etc.).

La **transferencia internacional de datos** es considerada como el “transporte de datos entre sistemas informáticos por cualquier medio de transporte, así como de datos por correo o por cualquier otro medio convencional”. Las transferencias internacionales están permitidas únicamente:

- 1) Con destino a países que proporcionen el nivel de protección que presta la LOPD. Bajo el régimen de la Directiva europea, esto incluye obligatoriamente a los miembros de la Unión Europea que hayan implementado correctamente la Directiva y a cualquier otro país aprobado por la Comisión Europea y, para España, la APD; o
- 2) En algunos otros casos específicos, por ejemplo, cuando el destinatario haya firmado un contrato que garantice niveles similares de protección de datos o transferencias entre miembros de un grupo empresarial que establezca una política interna adecuada de protección de la privacidad.



Los países aceptados hasta hoy son: Argentina, Canadá, Hungría, Suiza y la isla anglonormanda Guernsey (y EE.UU. bajo los principios de “Safe-Harbour”). Las otras excepciones a la regla general son:

- Cuando la transferencia internacional de datos de carácter personal resulte de la aplicación de tratados o convenios en los que España sea parte.
- Cuando la transferencia se haga a efectos de prestar o solicitar auxilio judicial internacional.
- Cuando la transferencia sea necesaria para la prevención o para el diagnóstico médicos, la prestación

de asistencia sanitaria o tratamiento médico o la gestión de servicios sanitarios.

- Cuando se refiera a transferencias dinerarias conforme a su legislación específica.
- Cuando la transferencia sea necesaria para la ejecución de un contrato entre el afectado y el responsable del fichero o para la adopción de medidas precontractuales adoptadas a petición del afectado.
- Cuando la transferencia sea necesaria para la celebración o ejecución de un contrato celebrado o por celebrar, en interés del afectado, por el responsable del fichero y un tercero.
- Cuando la transferencia sea necesaria o legalmente exigida para la salvaguarda de un interés público.
- Cuando la transferencia sea precisa para el reconocimiento, ejercicio o defensa de un derecho en un proceso judicial.
- Cuando la solicitud de transferencia se efectúe, a petición de persona con interés legítimo, desde un registro público y aquella sea acorde con la finalidad del mismo.

## Seguridad

Ya comentamos que el responsable del fichero y, en su caso, el encargado del tratamiento debe respetar ciertas **obligaciones de seguridad** para la protección de datos personales. La Directiva no predica ninguna medida en particular y los miembros de la Unión Europea han tomado distintas perspectivas sobre el tema, desde la autorregulación (Inglaterra) hasta las medidas detalladas obligatorias (España).

En efecto, España tiene el régimen más estricto (quizá con mayor certidumbre legal pero también más inflexibilidad técnica). Bajo la LOPD

y el Real Decreto de las Medidas de Seguridad (RD 994/1999, de 11 de junio), se establecen tres niveles de protección en función de la información tratada:

- 1) El **nivel básico**: se aplica por defecto a cualquier fichero cubierto por la LOPD.
- 2) El **nivel medio**: se aplica a ficheros que contengan datos relativos a la comisión de infracciones administrativas o penales, Hacienda Pública, servicios financieros y los de solvencia patrimonial y crédito.
- 3) El **nivel alto**: se aplica a ficheros con datos sobre la ideología, religión, creencias, origen racial, salud o vida sexual y los recabados para fines policiales sin consentimiento del afectado (se corresponde, más o menos, con los datos especialmente protegidos de la LOPD y la Directiva).

Todo fichero de datos personales debe tener adoptadas las medidas de seguridad del nivel correspondiente al tipo de datos existentes en dicho fichero. Las medidas de seguridad tienen carácter acumulativo, debiéndose adoptar las medidas correspondientes al nivel de seguridad de que se trate, así como las correspondientes a los niveles inferiores.

Los diferentes niveles de seguridad comportan obligaciones cada vez más onerosas, a implementarse por el técnico encargado de los datos (responsable de seguridad).

Hay que resaltar algunos otros temas importantes relevantes para la seguridad:

- **Documento de seguridad**: se debe elaborar, conforme a las prescripciones legales, un documento de seguridad que se mantenga actualizado y periódicamente revisado.
- **Auditoría**: se debe realizar periódicamente una auditoría (técnico-legal) de las medidas de seguridad implantadas que verifique su cumplimiento.



Las obligaciones de seguridad en España son:

- a) En el **nivel básico**, se deben establecer las siguientes medidas:
- Elaboración e implantación de la normativa de seguridad mediante un documento que deberá contener como mínimo las indicaciones del artículo 8 del Real Decreto.
  - Definición clara y documentada de las funciones que desempeñará y obligaciones que asumirá cada persona con acceso a los datos y sistemas de información, así como mantener una relación actualizada de usuarios con acceso autorizado y el establecimiento de sistemas de identificación y autenticación.
  - Creación de un registro de incidencias.
  - Verificación, por parte del responsable del fichero, de la definición y correcta aplicación de los procedimientos de copias de seguridad.
- b) En el **nivel medio**, a las medidas anteriores, hay que agregar otros requisitos. Particularmente, hay que realizar una auditoría, interna o externa al menos cada dos años. El informe de auditoría estará a disposición de la APD.
- c) En el **nivel alto**, hay que añadir obligaciones relativas a la distribución de soportes, el registro de accesos, la realización de copias de seguridad y la transmisión cifrada a través de redes telemáticas.

En relación con las **sanciones**, la LOPD sistematiza los posibles incumplimientos de la Ley y tipifica las infracciones relativas a la pro-



tección de datos personales. No hay una lista exhaustiva, sino unos criterios que incluyen básicamente:

- a) El descuido del interesado en el ejercicio de sus derechos (acceso, rectificación o cancelación).
- b) La insuficiencia en la información que se le provee en el recabado de los datos.
- c) La falta de colaboración con la Agencia.
- d) La ausencia de notificaciones preceptivas (como las de creación de fichero).
- e) La creación, tratamiento, comunicación, cesión y mantenimiento de ficheros sin la observancia de las prescripciones de la ley.

Hay tres categorías de sanciones:

- Por infracciones leves: multa de 601,01 a 60.101,21 euros.
- Por infracciones graves: multa de 60.101,21 a 300.506,05 euros.
- Por infracciones muy graves: multa de 300.506,05 a 601.012,10 euros.

Se han dado a conocer varias sanciones bajo la LOPD, a pesar de su supuesto secreto (asimismo, en 2003, con el cambio de organización de la APD, se han cambiado también varias políticas de la Agencia): durante el año 2000, por ejemplo, las multas ascendieron a más de 12 millones de euros.

- En 2000, 180 millones de pesetas (1,1 millones de euros) a Zepelín (TV5) por revelar datos de candidatos al programa Gran Hermano (recurrido).
- En 2001, Telefónica de España y Telefónica Data fueron sancionados por un importe de 841.420 Euros por intercambiarse datos de clientes.

- En 2002, la empresa Inlander tuvo que pagar 300.000 euros por tener instalado el servidor en Estados Unidos.

### 8.3.3. Marco legal en otras jurisdicciones

Aunque hayamos comentado en detalle el marco legal español, debemos resaltar que la mayoría de los derechos y obligaciones desarrollados existen con pequeñas variaciones en todos los países de la Unión Europea, por efecto de la Directiva de Protección de Datos Personales de 1995. La mayor diferencia reside en la medidas de seguridad que se han explicitado en España.

Fuera de Europa, notamos que por efecto de estas obligaciones y, sobre todo, de las relativas a la transferencia internacional de datos, la mayoría de los socios comerciales de los países europeos se ven casi “obligados” a establecer marcos legales similares para la protección de la privacidad. Mencionamos Canadá, Hungría, Suiza y Argentina, que han sido aprobados por la Comisión Europea, pero también Japón o Australia.

Un caso particular es el de Estados Unidos, que tiene una protección de privacidad muy menor al marco europeo, y cuya organización es sectorial: sobre todo, para los bancos y servicios financieros y el sector de la salud. Para permitir la transferencia de datos desde la UE, los EE.UU. ha establecido un régimen cuasiprivado, por medio del acuerdo de *Safe Harbour* de julio de 2000. Cualquier empresa que se comprometa a cumplir las obligaciones establecidas por el gobierno americano (el Department of Commerce) y acordadas por la Comisión Europea, puede recibir transferencias de datos de carácter personal desde la UE. Hasta la fecha, unas 50 empresas americanas han firmado el acuerdo. Otras, han firmado contratos estándares obligándose a la protección adecuada.



El marco legal completo de la protección de datos de carácter personal en España:

- Directiva 95/46/CE del Parlamento Europeo y del Consejo, de 24 de octubre de 1995, relativa a la

protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de datos.

- Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.
- Real Decreto 1332/1994, de 20 de junio, por el que se desarrollan determinados aspectos de la Ley Orgánica 5/1992, de 29 de octubre, de regulación del tratamiento automatizado de los datos de carácter personal.
- Real Decreto 994/1999, de 11 de junio, por el que se aprueba el Reglamento de medidas de seguridad de los ficheros automatizados que contengan datos de carácter personal.
- Real Decreto 195/2000, por el que se establece el plazo para implantar las medidas de seguridad de los ficheros automatizados previstas por el Reglamento aprobado por el Real Decreto 994/1999, de 11-6-1999.

#### 8.3.4. Datos personales y software libre: la seguridad

Este resumen del marco legal europeo de la protección de datos indica que la mayoría de las obligaciones y derechos relativos al procesamiento de datos personales no tiene una relación inmediata con el software libre. No es por tener software libre o propietario que una persona cumple o viola este reglamento, sino que son los procesos de negocio y de tratamiento los que influyen sobre el cumplimiento: si se ha obtenido el consentimiento, si se da una oportunidad de revisión y modificación de los datos o si se respetan las políticas declaradas de privacidad.

El ámbito en el que sí que hay un vínculo importante con el software libre (o propietario) está en el área de la seguridad. A nivel europeo,

la Directiva de 1995 obliga a los “responsables de tratamiento” a implementar medidas tecnológicas y organizativas para garantizar la protección de datos y el respeto de los derechos otorgados. Aunque haya diferencias de implementación de la directiva en este sentido (el ordenamiento legal español ha explicitado estas obligaciones en el decreto antes mencionado, mientras que los ingleses las han dejado a la determinación experta de los administradores de sistemas), se puede decir que hay un acuerdo general en que estas medidas deben cumplirse a través de políticas de seguridad adecuadas (a nivel organizativo) y de aplicaciones informáticas seguras (a nivel tecnológico).

Los riesgos de violar estas obligaciones de protección tecnológica son grandes. Surgen dentro de los sistemas corporativos y en las redes públicas. Una lista no exhaustiva incluirá:

- Los accesos no autorizados a partes protegidas de sistemas corporativos que contienen y procesan datos personales (por ejemplo, los directorios del personal, las listas de clientes o de pacientes, datos de cuentas bancarias y de tarjetas de crédito, etc.);
- La interceptación de flujos de datos en las redes públicas (y a veces las VPN);
- La vulnerabilidad a ataques externos por *trojan horses* u otras formas de programas dañinos;
- El robo de identidad (*spoofing*) a través de la cosecha de datos en comunicaciones no seguras;
- La diseminación involuntaria de datos personales (por error de programación o por fallo de sistema).

**Nota**

En septiembre de 2003, debido a un fallo de seguridad, Ya.com permitió el acceso a sus facturas con datos personales incluyendo nombres, NIF, cuenta bancaria, teléfono, etc. Fue penalizada por la APD.



Las consecuencias de un fallo de seguridad que implique la violación de la protección de datos personales son graves, no solamente en la diseminación de datos personales al público (o a empresas privadas que abusan de ellos, como ciertas empresas de marketing en línea) sino en las sanciones administrativas.

Por lo tanto, el tema de la seguridad de las aplicaciones informáticas es primordial en esta área de la protección de datos personales.

### Seguridad informática ¿software abierto o software propietario?

En la esfera de la seguridad, hay un gran debate entre los defensores del software libre y del software propietario. Ambos bandos argumentan que su proceso de desarrollo y el resultado final –abierto o compilado– lleva a mayores niveles de seguridad.

No vamos a entrar en ese debate, nos limitamos a resumir los argumentos:

- a) A favor del software propietario, se arguye que el secreto del código fuente impide que los atacantes descubran de manera sencilla los defectos (*bugs*) de seguridad. Asimismo, es relevante que el proceso de subsanar el código es una tarea difícil e ingrata. Las empresas propietarias tienen la ventaja de poder pagar a expertos para realizar este trabajo, mientras que el desarrollo libre no tenderá a incentivar este tipo de trabajo. Tercer punto: se argumenta que una empresa con responsabilidades se hace cargo del software propietario y coordina y concentra el trabajo del buen diseño original (referente a la seguridad) y la corrección de errores. No hay ninguna garantía de este “cuidado” en el software libre, ni para la distribución de parches. La mera publicación de código en Internet no implica que será examinado desde la perspectiva de la seguridad. Finalmente, nada garantiza la calidad de código contribuido a un proyecto libre que no tenga un coordinador de alto nivel.

#### Nota

Las sanciones administrativas incluyen multas que suman hasta 600.000 euros (por violación, que pueden acumularse) en España o 5.000 libras en el Reino Unido, dependiendo de la gravedad y la frecuencia de la infracción.

#### Lectura complementaria

Hay varios artículos e informes publicados sobre el tema mencionados en la bibliografía.

- b) A favor del software libre, se argumenta que el desarrollo libre se beneficia de “muchos ojos” para descubrir los fallos. Los programas abiertos no usan sus usuarios como “testadores”. Sin embargo, hay pocos proyectos libres, con excepciones como Linux o Apache, que tienen muchos desarrolladores: el promedio para muchos es entre 4 y 6 personas.

Asimismo, una vez descubierto, un defecto es arreglado en muy poco tiempo (horas o días), mientras que con el software propietario, puede haber un tiempo largo entre el descubrimiento de fallos y la distribución de su corrección. Hasta se alega que solamente ante la presión mediática se esfuerzan las empresas “propietarias” por corregir errores. Y se argumenta que el método de diseño abierto es más seguro que el método propietario, citando los RFC (*Requests for Comment*, propuestas de estándares en forma de sugerencia de diseño, como IPsec o S/MIME) de las organizaciones de estándares como W3C e IETF.

- c) Por el lado del usuario, se argumenta que el código abierto permite a los administradores de sistemas abiertos eliminar código indeseado o inseguro, diseñar e incorporar sus propios parches y procesos de seguridad e integrar extensiones de seguridad de terceros.
- d) Ninguna licencia de uso de software, libre o propietario, garantiza una seguridad del 100% y el respeto de las obligaciones de protección de datos almacenados o tratados en sus programas: el cumplimiento de las leyes depende demasiado del diseño del sistema total, de los procesos implementados y de las políticas de seguridad de las empresas.

En términos de estadísticas, tanto Windows como GNU/Linux sufren fallos de seguridad, según la organización CERT, que monitorea e investiga estos temas. No hay ventaja “numérica”, a pesar de los alegatos de la comunidad libre. Desde un punto de vista científico, se argumenta en que en igualdad de condiciones, los riesgos de un modelo y del otro son iguales. Todavía no hay ninguna investigación que lleve a una conclusión fehaciente a favor o en contra de un tipo de software u otro.



Para un comentario sobre el informe CERT, podéis ver:

“Study: Linux’ Security Problems Outstrip Microsoft’s”  
de J. MacGuire, (en: <http://www.newsfactor.com/perl/story/19996.html>).

R Andersen argumenta que las simetrías teóricas entre software abierto y propietario frente a la seguridad están rotas por varios factores que pueden favorecer a un modelo u otro. Podéis ver:

“Security in Open Systems v. Closed Systems – The dance of Boltzmann, Coase and Moore”, en la bibliografía.

## Conclusiones sobre seguridad

Las obligaciones impuestas sobre las personas que procesan datos personales obligan a que se tome la seguridad muy en serio en el momento de elegir, diseñar, desarrollar y/o implementar una plataforma informática. Uno de los argumentos de mayor fuerza de OpenBSD, por ejemplo, es que su versión de Unix es más la segura y fiable, no solamente por las herramientas de cifrado y protección de comunicaciones que trae, sino también porque tiene un equipo dedicado que se ha especializado en resolver los problemas de seguridad de UNIX. Por lo tanto, un análisis de los aspectos de la seguridad es fundamental en la consideración de la implementación de nuevos sistemas informáticos.

Por otro lado, el hecho de que los sistemas libres sean modificables y accesibles por cualquiera que tenga acceso a un equipo (a su funcionamiento interno) significa que se deben cuidar mucho más las autorizaciones, los accesos, las políticas de claves y los registros de control. Esta aclaración está en línea con la filosofía de que la seguridad de un sistema no depende de su nivel de seguridad interno, sino del compromiso de los administradores y responsables para establecer y exigir el cumplimiento de una política de seguridad adecuada. Una obligación impuesta, por ejemplo, por el Real Decreto español, reforzada por una auditoría bianual en casos de datos sensibles que requieren un alto nivel de protección.

### Nota

En la unidad 10 incluimos una actividad relacionada con este debate.

## 8.4. El software libre y los controles sobre los productos de seguridad

Otra área de derecho relevante para el software en general es la de los controles de la exportación de productos de seguridad –sobre todo, los sistemas de cifrado. En este último apartado queremos presentar el por qué de la regulación de estos productos, cómo afecta al software libre y qué se puede hacer (a nivel de licencia) para minimizar los riesgos de incumplir la ley.

### 8.4.1. Seguridad y la sociedad de la información

En la incipiente sociedad de la información, con todas sus amplias capacidades de procesamiento de datos diversos, la multiplicación de las operaciones diarias que se efectúan a través de las redes informáticas y su potencial para la invasión de la vida privada, hay una necesidad creciente de mantener los datos comerciales y privados seguros y confidenciales.

#### Ejemplo

Algunos ejemplos de operaciones que implican riesgos mayores son:

- a) Las comunicaciones privadas personales y de negocios, incluyendo conversaciones telefónicas, mensajes de fax y correo electrónico;
- b) Las transferencias electrónicas de fondos y otras transacciones financieras;
- c) La información confidencial de negocios y secretos comerciales;
- d) Los datos usados en la operación de sistemas críticos de infraestructuras, tales como el control de tráfico aéreo, las redes telefónicas y la energía eléctrica, y
- e) Los informes médicos, financieros, archivos personales y otra información personal.



Los avances en las tecnologías de la información y la comunicación implican que cada vez sea más difícil controlar los flujos de datos y el uso de la información confidencial o saber de dónde viene un mensaje o quién es el destinatario real de una comunicación. La infraestructura de comunicaciones de nuestra sociedad se basa en ordenadores inseguros y redes inseguras y está plagada de problemas de seguridad, ya sea por los protocolos inseguros (en términos de seguridad) en los cuales se basa, como el TCP/IP, SMTP o HTTP, por varios programas perjudiciales como los virus y otras aplicaciones dañinas o por los errores de codificación de los programas de mayor uso que dejan nuestros equipos y archivos abiertos a ataques e intrusiones de terceros.



Podéis ver, por ejemplo, el informe “CyberInsecurity: The Cost of Monopoly” de Computer and Communications Industry Association, en:

<http://www.ccianet.org/papers/cyberinsecurity.pdf> (visitado 24/09/2003).

Este informe ha suscitado un debate feroz sobre los *white papers* y otros informes patrocinados económicamente.



La seguridad toma cada vez más importancia en la vida diaria de cada persona. Usando la tecnología para defenderse de la tecnología, hay una serie de productos informáticos llamados “de seguridad” que se pueden implementar para garantizar, aunque sea parcialmente, esta seguridad.

La mayoría de estas aplicaciones se basan en la criptografía o cifrado, que aparece como el único sistema viable para garantizar la confidencialidad y la seguridad de la información. Basándose en procedimientos matemáticos para “desorganizar” y “reorganizar” datos, permite almacenar y enviar mensajes electrónicos de tal manera que es casi imposible recuperar el texto original por alguien que no sea uno de los destinatarios autorizados y no tenga las claves de descifrado.

Para paliar los problemas más básicos de estas comunicaciones, existen varios protocolos y tecnologías que usan el cifrado y que se implementan en la mayoría de los programas de comunicaciones, como el correo electrónico (S/MIME) y los navegadores de Internet (HTTPS). Estas tecnologías impiden la interceptación de la comunicación y el acceso no autorizado, es decir, preservar la **confidencialidad**. Para garantizar otros aspectos de la seguridad, pueden usarse distintas técnicas criptográficas. Asimismo, favorecen la certeza jurídica (por ejemplo, para un contrato electrónico seguro):

- La **integridad**: asegurar que los datos no hayan sido modificados.
- La **autenticación**: establecer la identidad de las partes de una comunicación.
- El **no-repudio**: garantizar que una parte no pueda negar la existencia o recepción de un mensaje.

El cifrado asimétrico, en base al sistema PKI (*Public Key Infrastructure*, infraestructura de clave pública), es una de estas tecnologías, que garantiza la seguridad de una comunicación entre dos personas con claves diferentes (pública y privada) por cifrado asimétrico.

El cifrado puede tener varios grados de seguridad, porque con esfuerzos masivos (*brute force*) de procesamiento se pueden descubrir las claves. Por lo tanto, lo que se considera “cifrado de alto nivel” (*strong encryption*) variará según los avances tecnológicos. Hasta una época reciente, el cifrado con 40 bits se consideraba suficientemente alto como para justificar restricciones sobre la exportación de los productos que lo incluían. Hoy, el cifrado a 128 bits se considera el mínimo para garantizar una comunicación segura.

**Nota**

El cifrado de alto nivel se usa en los servicios bancarios en línea. Podéis consultar como ejemplo:

[www.banesto.es/banesto/home2/informacion\\_ssl128.htm](http://www.banesto.es/banesto/home2/informacion_ssl128.htm) (visitado 24/09/2003)

### **8.4.2. Diferentes tipos de protocolos y aplicaciones de seguridad**

En un principio desarrolladas en el ámbito militar y en laboratorios de investigación académicos, las tecnologías de criptografía entran ahora en el dominio del gran público y son usadas por todos. Varias aplicaciones y componentes de aplicaciones están disponibles para garantizar los diferentes aspectos de la seguridad en las redes.

#### **1) Algoritmos**

El algoritmo más conocido para el cifrado de alto nivel es el RSA, inventado en los laboratorios del MIT en 1978. Otros algoritmos de cifrado son Triple DES, IDEA y Blowfish. El uso del RSA fue protegido por patente en EE.UU. hasta septiembre de 2000.

La patente sobre el algoritmo RSA es interesante, porque frenó la producción de productos de mayor seguridad para los desarrolladores de soluciones de comercio electrónico y productos de seguridad que no podían pagar los derechos de patente a PKP (Public Key Partners, la empresa formada para comercializar la patente, y RSA Security).

Sin embargo, en Europa el algoritmo fue publicado antes de la petición de patente, lo que significa que no se benefició de la protección de patente y no había restricción legal sobre su uso. Ahora, después del vencimiento de la patente, las aplicaciones que usan el algoritmo RSA proliferan.

#### **2) Protocolos y programas de correo electrónico:**

- a) **S/MIME** (*Secure Multi-purpose Mail Extensions*): un estándar propuesto para el intercambio seguro de correo electrónico.
- b) **PGP** (*Pretty Good Privacy*): para el cifrado asimétrico del correo electrónico (y archivos), implementando el algoritmo RSA. La primera versión fue “liberada” a amigos del autor, Phil R. Zimmermann en 1991, con algunas versiones fuera de EE.UU. Como EE.UU. ya tenían controles sobre la exportación de productos de cifrado, se inició un juicio contra Phil Zimmermann, que fue abandonado en 1996.

- c) **OpenPGP** ([www.openpgp.org](http://www.openpgp.org)) es el estándar actual de cifrado para correo basado en PGP de Phil Zimmermann (estándar propuesto RFC 2440 del IETF).



PGP tiene una historia muy interesante que merece ser leída. Para ver una historia de PGP, consúltase:

<http://www.cypherspace.org/~adam/timeline/> (visitado 24/09/2003)

### 3) Servicios de redes públicas como Internet:

- a) **SSH** (*Secure SHELL*): es un protocolo para crear conexiones seguras entre dos sistemas. Permite el intercambio de datos (nombres de usuario, claves de acceso) sin interceptación en "túneles" seguros por TCP. SSH no cifra solamente la secuencia inicial de acceso al sistema, cifra toda su sesión de trabajo.
- b) **SSL** (*Secure Sockets Layer*): un estándar desarrollado por Netscape Communications para transmitir información de forma segura por Internet. El SSL permite la creación de un canal de comunicación seguro entre el servidor y el navegador de su cliente.

La mayoría de los navegadores actuales (IE, Netscape, Mozilla, Opera, etc.) soportan SSL y cifrado con 128 bits.

### 4) Redes privadas (VPN)

Hay varios protocolos para la seguridad en redes (privadas o públicas). La de mayor importancia actualmente es IPsec, una propuesta de estándar para la IETF ([www.ietf.org/html.charters/ipsec-charter.html](http://www.ietf.org/html.charters/ipsec-charter.html)). Tiene dos modalidades: *Authentication Header* (AH), para el cifrado del contenido de un mensaje y *Encapsulated Security Payload* (ESP), para el cifrado de todo el mensaje, incluso el encabezamiento.

Otros protocolos de seguridad para las redes privadas (VPN, Virtual Private Networks) son *Point to Point Tunelling Protocol*, aplicado con capas de seguridad SSH en redes (PPTP VPN) y L2TP IPsec VPN.

### 8.4.3. Los controles sobre los productos de seguridad

El cifrado no solamente protege la confidencialidad de los datos contra el robo, la interceptación y el acceso no autorizado, sino que también impide que las autoridades gubernamentales como la policía o los oficiales de aduana puedan interceptar y acceder a las comunicaciones.



Con el argumento de que se necesita controlar estas comunicaciones para mantener el orden público y la lucha contra el terrorismo, varios gobiernos han impuesto distintos tipos de controles sobre el uso de los productos de cifrado.



Interpretación de la revista en línea Internautas, accesible en línea en:

<http://www.internautas.org/NOTICIAS/DIC98/11.htm>.

“El Gobierno quiere controlar la criptografía para mantener sus posibilidades de escuchas (esa intención ha sido traicionada por las propuestas fallidas de adopción del chip Clipper por la Administración). La posición del Gobierno de EE.UU. es que las técnicas criptográficas pueden ser usadas por malhechores para madurar sus planes ilegales y la policía tendría mucho más difícil seguir sus huellas. Terroristas internacionales podrían usar correo electrónico cifrado para planear alguna mala jugada en territorio norteamericano y el FBI ni se enteraría.”

A continuación vamos a ver el marco legal internacional de los controles a la exportación, con un enfoque sobre tres de los países con mayores restricciones: EE.UU., Reino Unido y Francia.

Los instrumentos de mayor importancia relativos al control de la distribución de productos de cifrado son:

- Las Directrices de la OCDE, de 1997.

- El Acuerdo Waasenaar, de 1995 (modificado en 1998 y 2000).
- El Convenio sobre el Cibercrimen, del Consejo de Europa de 2001.
- Los Reglamentos de la Unión Europea.
- Las regulaciones administrativas nacionales.

### Las directrices de la OCDE

Las directrices de la OCDE, que no son vinculantes ni sobre los estados miembros de la OCDE ni sobre las personas, enfatizan la necesidad de mantener disponibles productos de cifrado para la privacidad y confidencialidad de los datos (principios 2 y 5), sujeto a medidas proporcionales y efectivas para mantener el orden público (principio 6: el acceso legítimo). Las directrices sugieren, por ejemplo, que estas medidas “podrían” incluir derechos de acceso a datos (es decir, que los gobiernos “podrían” exigir que las claves de cifrado estén a disposición de las fuerzas del orden).

#### Nota

No tratamos aquí el tema del acceso a las claves, *key recovery*, por no ser directamente relevante en el software libre. Podéis consultar en la bibliografía los textos sobre el tema:

H. Abelson et al. *Los riesgos del cifrado*.

Unión Europea. *Green Paper sobre servicios de cifrado en el mercado interno*.

### El Acuerdo Waasenaar sobre el material de defensa y de doble uso

El Acuerdo de Waasenaar es un tratado internacional ratificado por 33 países, para la imposición de controles sobre la exportación de

armas convencionales y bienes y tecnologías de doble uso. Fue firmado en 1995 y entró en vigor en 1996.

En este tratado se crea un marco de cooperación bajo el cual los gobiernos se comprometen a instaurar controles sobre varios productos que pueden usarse contra el orden público, para impedir su transferencia o exportación. El acuerdo determina una lista de productos y tecnologías controlados, entre los cuales figuran tecnologías de criptografía (categoría 5-2) (en el General Software Note). En relación con estas tecnologías, la lista ha sido modificada varias veces:

- a) Hasta 1998, todos los productos de criptografía estaban incluidos, excepto los productos destinados al gran público y los que se encontraban “en el dominio público”.
- b) En 1998, debido a varias campañas a favor de la privacidad y la criptografía, se eliminaron de la lista algunos productos relativos a la autenticación (para la firma digital) y los algoritmos de cifrado de menos de 56 bits. Sin embargo, se amplió la lista con cualquier producto de hardware y software con algoritmos de más de 64 bits.
- c) En 2000, se elimina el límite de 64 bits para el software y hardware para el gran público.

Extractos del Acuerdo Waasenaar:

#### **“Nota sobre el cifrado” (“Cryptography Note”)**

“Artículos 5.A.2 y 5.D.2 no controlan los productos que cumplan con todos los requisitos siguientes:”

“a. Que se halle generalmente a disposición del público, por estar a la venta, sin limitaciones, en puntos de venta al pormenor, de:

1. Transacciones en mostrador;
2. Transacciones por correo;
3. Transacciones electrónicas; o
4. Transacciones por teléfono; y

b. Cuya funcionalidad criptográfica no pueda modificarse fácilmente por el usuario;

#### **Lectura complementaria**

Extracto del Acuerdo Waasenaar, accesible en:  
<http://www.wassenaar.org/>

- c. Que esté concebido para su instalación por el usuario sin asistencia ulterior importante del proveedor; y
- d. Eliminado;
- e. Cuando sea necesario, los detalles de los productos sean accesibles y provistos, a pedido, a la autoridad adecuada en el país de exportación para averiguar que cumplan con las condiciones (a.) a (e.)."

#### **"Nota sobre el software" ("Software Note")**

"La listas no controlaran software:

- a. Que se halle generalmente a disposición del público, por estar a la venta, sin limitaciones, en puntos de venta al pormenor, de:
  - 1. Transacciones en mostrador;
  - 2. Transacciones por correo;
  - 3. Transacciones electrónicas; o
  - 4. Transacciones por teléfono; y
- b. Que esté concebido para su instalación por el usuario sin asistencia ulterior importante del proveedor; o
- c. Que sea 'de conocimiento público' (dícese de la 'tecnología' o 'equipo lógico' (software) divulgado sin ningún tipo de restricción para su difusión posterior.

Nota: Las restricciones derivadas del derecho de propiedad intelectual no impiden que la 'tecnología' o el 'equipo lógico' (software) se consideren 'de conocimiento público'."

En relación con el software libre, este acuerdo tiene varias implicaciones:

- 1) Las implementaciones nacionales del acuerdo afectarían a la distribución de tecnologías de cifrado de alto nivel (mayor a 56 bits) fuera de su país. La distribución de software por Internet ("colgar en la red") está incluida en el concepto de exportación (y, por lo tanto, sujeta a controles).



- 2) Sin embargo, la definición de “dominio público” en estos acuerdos no es una definición legal (correspondiendo a una definición establecida en jurisdicciones como, por ejemplo, España, Inglaterra o EE.UU.), sino que alcanza a tecnología o software que ha sido puesto a disposición del público (divulgado) sin restricciones para su diseminación posterior. La nota mencionada en el texto resalta el hecho de que las restricciones bajo el derecho de propiedad intelectual no se consideran restricciones sobre su diseminación. Esto implica que gran parte del software libre se considera “en el dominio público”. Según la definición OSI y, por ejemplo, bajo los términos de la licencia GPL, los programas libres se distribuyen al público (por Internet) y no se puede limitar su redistribución.

Sin embargo, el acuerdo Waasenaar no es derecho directamente aplicable, cada país lo implementa como quiere. Esto implica que hay varias diferencias en los regímenes nacionales, que estudiaremos más adelante.



La interpretación de la FSF sobre el acuerdo de Waasenaar es interesante. Accesible en:

[www.gnu.org/philosophy/wassenaar.html](http://www.gnu.org/philosophy/wassenaar.html)

## Acuerdos regionales

Dentro los países de la Unión Europea, hay que mencionar el Reglamento 1334/2000 sobre los materiales de doble uso y sus modificaciones en Reglamentos 458/2001 y 2432/2001. Estos reglamentos son directamente aplicables en todos los países de la Unión. Implementan el acuerdo de Waasenaar y establecen listas de productos sujetos a control, con obligaciones de obtener licencias gubernamentales para su exportación:

- a) El Reglamento permite el libre movimiento de productos de criptografía dentro de la Unión Europea e impone restricciones sobre su exportación a terceros países (con licencia).

- b) La exportación a Australia, Canadá, la República Checa, Hungría, Japón, EE.UU., Noruega, Polonia, Suiza y Nueva Zelanda se puede realizar bajo una licencia general.
- c) Se mantiene la misma exención que Waasenaar para productos “en el dominio público”.
- d) En 1999, en línea con Waasenaar, se eliminaron los límites sobre software de cifrado para el gran público.

Dentro de la Unión hay un debate importante entre los defensores de la libertad y la privacidad, que quieren eliminar los controles, y los gobiernos –sobre todo, Inglaterra y Francia– que quieren mantener ciertas posibilidades de acceso a las comunicaciones.



Para más información sobre la posición de la UE, podéis ver el “Green Paper” sobre servicios de cifrado, COM(96)76, de 6 de marzo de 1996, en:

<http://europa.eu.int/en/record/green/gp9603/>  
(visitado 24/09/2003).

La decisión 94/942/PESC definió las listas originales, donde figuran las mismas exenciones que en el Acuerdo Waasenaar:

#### “Nota general para el equipo lógico (NGEL)”

“(La presente nota tiene primacía sobre los controles de la sección D en las categorías 0 a 9.) Las categorías 0 a 9 de esta lista no controlan al equipo lógico que cumpla al menos una de las dos condiciones siguientes:

- a. Se halle generalmente a disposición del público por estar:
  - 1. a la venta, sin limitaciones, en puntos de venta al por menor, por medio de:
    - a. transacciones en mostrador;
    - b. transacciones por correo;
    - c. transacciones por teléfono; y

2.

- a. concebido para su instalación por el usuario sin asistencia ulterior importante del proveedor; o
- b. es 'de dominio público'.

### Definiciones

'De dominio público' (NGT) (NTN) (NGEL): en el marco del presente documento, dicese de la 'tecnología' o 'equipo lógico' (software) divulgados sin ningún tipo de restricción para su difusión posterior [las restricciones derivadas del derecho de propiedad intelectual no impiden que la 'tecnología' o el 'equipo lógico' (software) se consideren 'de dominio público']."

Estas definiciones son retomadas en el ordenamiento español; por ejemplo:

"El anexo no somete a control al 'equipo lógico' ('software') que cumpla al menos una de las dos condiciones siguientes:

1. Se halle generalmente a disposición del público por estar:

a. A la venta, sin limitaciones, en puntos de venta al por menor, por medio de:

- 1. Transacciones en mostrador;
- 2. Transacciones por correos; o
- 3. Transacciones por teléfonos; y

b. Concebido para su instalación por el usuario sin asistencia ulterior importante del proveedor; o

2. Sea 'de conocimiento público'.

'De conocimiento público': dicese de la 'tecnología' o 'equipo lógico' (software) divulgado sin ningún tipo de restricción para su difusión posterior.

N.B.: Las restricciones derivadas del derecho de propiedad intelectual no impiden que la 'tecnología' o el 'equipo lógico' (software) se consideren 'de conocimiento público'."

Real Decreto 491/1998, de 27 de marzo, por el que se aprueba el Reglamento del Comercio Exterior de Material de Defensa y de Doble Uso (BOE 8.4.98), y corrección de errores (BOE 17.6.98).

### A nivel nacional

Debido a la exención de productos “en el dominio público” en la mayoría de los países, no vamos a entrar en muchos más detalles. A continuación, establecemos una tabla que resume de manera breve la regulación de productos de cifrado en algunos países.

**Tabla 9.**

	Régimen	Controles sobre exportación	Exenciones	Condiciones para exención
EE.UU.	Interno + Waasenaar (v. antes de 1998)	EAR - Licencias o notificaciones obligatorias	Dominio público y otros (ved abajo) Gran público	Previa notificación al gobierno (BIS) (ved más adelante)
Canadá	Waasenaar (v. antes de 1998)	Paralelo con EE.UU.	Cifrado hasta 56 bits Exportación a EE.UU. Dominio público Gran público	Control de la reexportación de productos de EE.UU.
Francia	Waasenaar (v. antes de 1998) + UE	Licencia obligatoria	Gran público Cifrado hasta 40 bits	
UK	Waasenaar + UE + internos	Licencia obligatoria	Cifrado hasta 56 bits Dominio público Gran público	
España	Waasenaar + UE	Licencia obligatoria (ved recuadro más adelante)	Dominio público Gran público Cifrado hasta 56 bits	

Hay un análisis detallado en “Crypto Law Survey”, en: <http://rechten.kub.nl/koops/cryptolaw/index.htm>.

Un apunte sobre EE.UU.: en este país ha tenido lugar un gran debate público sobre los controles sobre la diseminación de la criptografía frente a la protección constitucional de libertad de expresión (la distribución del conocimiento referente al cifrado se puede considerar una forma de expresión).

Hasta el año 2000, EE.UU. mantenía un control estricto sobre la exportación de productos de cifrado dentro del régimen ITAR (*International Traffic in Arms Regulations*). Tres decisiones judiciales, Bernstein I, II y III, relativas a un programa que usa criptografía Snuffle, declararon incons-

titucionales algunas partes del ITAR, por ser limitaciones sobre la libertad de expresión.

La Administración de Clinton propuso varias modificaciones, para finalmente instaurar el régimen actual de EAR (*Export Administration Regulations*), manteniendo controles similares. Sin embargo, en enero y octubre de 2000, para implementar Waasenaar, se relajaron varios requerimientos. El régimen actual es el siguiente:

- a) La exportación de cualquier producto de criptografía a usuarios no gubernamentales se debe realizar con licencia, excepto a siete países “terroristas”, donde se prohíbe la exportación: Libia, Corea del Norte, Cuba, Irán, Irak, Sudán y Siria. En octubre de 2000, se eliminó la distinción entre usuario gubernamental y no gubernamental en los países de la UE y ciertos otros países amigos como Australia, Japón y Suiza (la misma lista que en la UE).
- b) Se permite la exportación de productos para “gran público” después de una revisión técnica del gobierno.
- c) Se permite la exportación de software con código fuente a disposición del público (*open source y community source*) sin restricciones y sin revisión técnica. Solamente se debe notificar al *Bureau of Industry and Security* (BIS), incluyendo una copia del código fuente (o su URL). Los países “terroristas” quedan prohibidos. Se puede “colgar” el código en Internet sin tener que controlar quién lo descarga.
- d) Se debe realizar una notificación posterior para ventas de cifrado mayor de 64 bits.

En junio de 2002 la exportación de productos para el gran público fue liberada. Ahora, sólo hace falta una revisión previa del BIS y ninguna notificación posterior. Para la UE y otros estados “amigos”, no hace falta esperar la revisión.

#### Nota

BIS: Antes llamado Bureau of Export Administration (BXA).

#### Nota

Para más detalles sobre el régimen norteamericano, podéis ver TSU – §§740.13(e) en las direcciones siguientes (visitado 24/09/2003):

- <http://www.bxa.doc.gov/Encryption/lechart1.htm>

- <http://www.bxa.doc.gov/Encryption/PubAvailEncSourceCodeNotify.html>

Aunque el régimen norteamericano se haya liberalizado parcialmente, sigue siendo suficientemente complejo y peligroso como para que los desarrolladores de software libre y los responsables de su distribución sigan restringiendo la disponibilidad de productos de cifrado de alto nivel desarrollados en EE.UU. o exigiendo que se desarrollen fuera de dicho país (ni siquiera se permite la contribución de código de desarrolladores de EE.UU.). Asimismo, una lectura pesimista del reglamento indica que cualquier programa notificado al BIS “y sus versiones siguientes, modificaciones y obras derivadas”, caerán bajo las reglas de exportación EAR para siempre.

Por lo tanto, corren el riesgo de una modificación de las reglas por parte del gobierno de los EE.UU. a favor de controles más restrictivos. Esto afectaría a todas las copias del programa distribuidas y sus modificaciones y derivados y cualquier aplicación tercera que incorpore el programa. No es de sorprender que no haya proyectos de software libre con criptografía de alto nivel distribuido desde EE.UU. El OpenBSD, por ejemplo, se distribuye desde Canadá.

#### **8.4.4. Conclusiones sobre el software libre y los controles de seguridad**

En la unidad 9, realizaremos algunas actividades de investigación y debate sobre este régimen legal y su interrelación con el software libre.

### **8.5. Conclusiones**

Aquí termina el segundo bloque del curso de aspectos legales del software libre –el bloque de las licencias. En esta unidad hemos tratado de situar los aspectos legales de las mismas y del software libre en general, en un contexto más práctico, así como de vincular las mismas con otras áreas del derecho no vistas hasta ahora (datos personales, etc.).

En relación con la vida profesional del lector, nos parece importante que pueda evaluar las consecuencias comerciales y tecnológicas de los elementos legales que venimos describiendo y comentando: qué hacer en relación con la eficacia o ineficacia legal de las licencias, cómo aprovechar las posibilidades de licencias múltiples, qué documentación o *check-list* podría ser de utilidad, qué estrategia o táctica hay que tomar frente a una duda legal sobre el software libre y qué proceso de decisión es el más adecuado.

**Nota**

Será interesante para el lector retomar las preguntas y los ejemplos planteados en la unidad 1, para ver cuántos puede contestar, resolver, encontrarle la dificultad legal y establecer una estrategia para superarla. Asimismo, en los estudios de caso veremos un cierto número de situaciones donde el conocimiento adquirido aquí será de gran utilidad.





## 9. Aspectos relevantes para la creación de software (estudio de caso)

Esta unidad y la siguiente están dedicadas íntegramente a estudiar varios casos prácticos de creación, desarrollo, distribución e implementación de software libre, vistos desde el punto de vista legal. Los casos consisten en una serie de experiencias e historias, algunas reales, algunas ficticias (pero basadas en experiencias reales), que describen cómo ciertas empresas y personas han abordado las diferentes actividades vinculadas con el software libre. Sobre la base de estas experiencias –sus actores, sus historias, las tecnologías, los intereses y las opiniones en juego– hemos formulado actividades para investigar y aprehender una serie de aspectos relevantes para nuestro estudio. Estas actividades resaltan las dimensiones jurídicas de los casos bajo estudio, en contraste con sus aspectos económicos o técnicos.

Tras la lectura de los casos y la realización de las actividades asociadas, esperamos que el lector tenga una percepción mayor y un entendimiento más exacto de los aspectos legales prácticos del software libre, como por ejemplo, las dificultades de aplicación e interpretación de las licencias, las ventajas e inconvenientes legales en la migración de un sistema informático propietario a uno “libre” o en cómo organizar una distribución de software libre. Asimismo, el lector tendrá una base empírica para poder formular su propia opinión sobre el marco legal y su aplicación en este sector, y unas herramientas de análisis y de orientación para cuando se enfrente a casos similares en su vida profesional.

Por lo tanto, el objetivo general de estos estudios de caso es ilustrar y fomentar el debate acerca de los conceptos y temas desarrollados en las unidades anteriores y procurar realizar algunas reflexiones sobre los interrogantes, los riesgos y los problemas legales que puedan surgir respecto del software libre. A través de las actividades de investigación, las respuestas a preguntas diseñadas a orientar esta reflexión, el debate y la creación de una documentación parcial o completa relevante para un tema u otro, comprobamos si se ha podido integrar y aplicar el aprendizaje de las unidades anteriores.

Los casos elegidos en las unidades 9 y 10 abarcan desde la creación, el desarrollo y la distribución del software libre hasta su implementación y uso en aplicaciones empresariales, públicas o privadas. Asimismo, los trabajos a realizar consideraran las perspectivas del creador y del desarrollador, del distribuidor y del usuario de las aplicaciones en cuestión.

En esta unidad 9, estudiaremos los casos siguientes:

- En una primera serie de casos –**QDR-Soft, Iván Lasser**– investigaremos los interrogantes y decisiones a los cuales se enfrentan los desarrolladores de software libre y sus empresas: cómo proteger las aplicaciones en cuestión, cómo elegir una licencia en función de varios criterios relevantes, cómo estructurar legalmente las actividades de creación y desarrollo de software libre.
- Luego, en **LibreSolutions** y **MySQL v. Progress Software**, reflexionaremos sobre algunas experiencias frente a la distribución y comercialización de software libre: los riesgos de la distribución, la aplicación de la cláusula 2.b de la licencia GPL de *copyleft*, la licencia doble, el uso de las marcas y los puntos importantes de los contratos de distribución y de prestación de servicios relacionados.
- Finalmente, basándonos en una extensión de los casos anteriores **QDR-Soft** y **LibreSolutions**, investigaremos la **dimensión laboral del desarrollo y de la comercialización de software libre**: cómo articular la relación entre la empresa y los empleados y determinar los pasos convenientes o necesarios para crear y mantener una comunidad de desarrollo.

Cada estudio consiste en un texto principal, la lista de preguntas y actividades correspondientes, una serie de referencias específicas al caso y unas lecturas complementarias sugeridas para completar la presentación de los detalles del caso. Para el mejor aprovechamiento de estos estudios de caso, seguiremos los pasos siguientes:

- Leer detalladamente los casos, analizando los puntos más importantes.

- Realizar una investigación en Internet sobre cualquier aspecto de interés sugerido por el caso.
- Reflexionar sobre los elementos y aspectos legales de los casos, guiado por las preguntas, respondiendo a las mismas.
- Establecer la documentación y las tablas y/o matrices analíticas como herramientas de análisis y reflexión.

Durante estas actividades, convendrá referirnos también a las unidades anteriores del curso, donde se presentan y comentan el marco legal del derecho de la propiedad intelectual e industrial y el análisis de las licencias de software.

Es importante observar que muchas de las situaciones presentadas y las preguntas realizadas no tienen respuestas definitivas, sino que forman una base para fomentar el debate y realizar una reflexión personal y en colaboración sobre estos temas. Esto nos permitirá establecer una metodología y unas herramientas de análisis frente al tema que nos preocupa. Aunque existen varios conceptos y elementos de respuesta que son claves para conocer y entender los temas tratados, otros elementos son de libre discusión y dependen de la contribución y participación activa de los participantes del curso.

A partir de la lectura de los casos y la realización de las actividades asociadas, el lector alcanzará los objetivos siguientes:

1. Tener una percepción mayor y un entendimiento más exacto de los aspectos legales prácticos del software libre, como
  - La protección del software libre.
  - La organización de la distribución de software libre.
  - Las dificultades de aplicación e interpretación de las licencias.
2. Formular vuestra propia opinión sobre el marco legal y su aplicación en este sector, basada en experiencias prácticas y desarrollar argumentos a favor y en contra de distintos usos del software libre.

3. Desarrollar algunas herramientas de análisis y orientación para cuando os enfrentáis a casos similares en vuestra vida profesional.
4. Valorar las distintas aplicaciones prácticas y los efectos de las licencias de software libre.
5. Crear una documentación práctica para la implementación de software libre en vuestra organización.
6. Comprobar el aprendizaje de las unidades anteriores del curso.

### 9.1. QDR-Soft. La selección de una licencia de software libre

Objetivos de estudio:

- Estudiar los criterios de elección de una licencia libre para proyectos nuevos y existentes.
- Anticipar los riesgos legales del desarrollo y distribución de software libre.
- Entender los efectos de la elección de una licencia de software libre sobre la empresa.
- Entender los aspectos y determinantes del proceso de desarrollo de software libre.

Métodos de estudio:

- Lectura de caso.
- Análisis y comparación de varias licencias en función de las características del software y del desarrollo.
- Establecer una matriz o tabla de referencia.
- Discusión.

### 9.1.1. Introducción

En el desarrollo y distribución de una aplicación informática, la elección del tipo de licencia que se usará para su distribución puede ser fundamental para su éxito a corto y largo plazo. Asimismo, debe inscribirse en la estrategia de negocios de la empresa. Las licencias típicas de software de distribución comercial (como por ejemplo, la licencia de usuario final de Microsoft o *end user licence agreement* en inglés), suelen tener restricciones de uso que protegen la propiedad intelectual de la empresa y permiten una explotación comercial de la misma basada en derechos de licencia y regalías. Las licencias de software libre, por el contrario, permiten la modificación y distribución libre pero imponen otras formas de restricciones, lo que resulta en nuevas implicaciones para las empresas desarrolladoras y distribuidoras.

En esta cuestión de la selección de una licencia adecuada, sobre todo, de una licencia libre, se deben tomar en consideración varios elementos, tales como el tipo de mercado y de usuario final, el modelo de desarrollo del software de la empresa y las opciones para la futura extensión modificación, soporte y mantenimiento del mismo.

A través del estudio de este caso del desarrollo y de la distribución de dos tipos de aplicaciones de software, se intenta aprehender las diferentes variables y criterios que pueden ser determinantes en la selección de una licencia.

### 9.1.2. Modelos de desarrollo y licencias

En la literatura y los estudios sobre el software libre, se acostumbra a contrastar el proceso de desarrollo de software propietario (dentro del marco de una organización) y el proceso de desarrollo de software libre. En 1997 Eric Raymond usa las metáforas de la catedral y del bazar para describir y analizar los procesos respectivos. En el primer caso, una estructura jerárquica y una planificación y diseño central (*top-down*) suelen establecer los mecanismos de coordinación necesarios para lograr los objetivos técnicos y comerciales del proyecto. El equipo de desarrollo trabaja según un plan de proyecto y entregas planificadas. En la mayoría de las empresas que desarrollan y comercializan aplicaciones informáticas, este modelo corres-

#### Lectura complementaria

Hay una selección de licencias de Microsoft en:

<http://proprietary.clendons.co.nz/licenses/eula/>

ponde a una selección de licencia que maximice las ganancias de la empresa desarrolladora a través de la protección de su propiedad intelectual, restricciones de uso (sobre todo la copia y distribución a terceros) y el cobro de derechos de licencia y/o regalías.

En el caso de la creación de software libre, suele participar una multitud de contribuidores en el desarrollo de manera voluntaria, entregando distintos elementos y módulos de la aplicación en cuestión según las condiciones, circunstancias y necesidades del proyecto y las ganas de cada contribuyente. Éstos pueden cumplir distintas funciones de analista, programador o *tester* (responsable de pruebas). Para la coordinación eficiente de este desarrollo descentralizado, suele establecerse un equipo de desarrolladores líderes que toman las decisiones claves sobre el diseño, las versiones del software y la licencias aplicables, a veces dando indicaciones de los elementos de programa deseados (como la lista de tareas de GNU), a menudo cumpliendo el papel de monitor de estándares y compilador de versiones estables.

Sin embargo, el modelo de desarrollo de software y el uso de un tipo de licencia u otro, no tienen una relación directa de causa y consecuencia (en un sentido o en el otro), sino que se interrelacionan de manera compleja, según distintas variables y criterios. En función de estas variables, la selección de licencia puede afectar considerablemente el éxito de un proyecto libre o comercial.

### 9.1.3. La empresa QDR-Soft

La empresa QDR-Soft desarrolla aplicaciones informáticas para clientes particulares, comerciales y gubernamentales. Tiene un personal de 50 programadores y analistas, liderados por jefes de proyecto y responsables de clientes de nivel alto. La empresa se creó a principios de los años 1990 y se ha beneficiado enormemente de la explosión de Internet y de las distintas formas de comercio electrónico entre 1999-2001. La empresa trabaja, sobre todo, con aplicaciones “llave en mano”, hechas a medida de los clientes, pero también presta servicios continuos de desarrollo, servicio y mantenimiento a largo plazo para algunos clientes principales. Gracias a estas relaciones, se ha mantenido a pesar de la crisis actual del sector.

## Programas

QDR-Soft se especializa en aplicaciones de gestión de contenidos empresariales y personales. Sus plataformas se ajustan a redes internas y externas de empresas medianas y grandes, por ejemplo las que tienen diferentes centros de operaciones que deben compartir información y procesos de negocio. Tiene un portafolio de aplicaciones que forma parte del núcleo de su oferta técnico-comercial:

- **QDnet:** QDnet es una aplicación para la gestión de contenidos para usuarios internos y externos de las organizaciones comerciales u otras. Tiene un módulo de tipo “Extranet” para la misma aplicación en entorno web. Incluye un servidor de aplicaciones y de web, un motor de bases de datos, y varias herramientas para la gestión de contactos, la planificación y tareas individuales y comunes, varios foros de mensajes y repositorios de documentos, y sobre todo una herramienta de *workflow* para procesos colaborativos y estructurados. Algunas funcionalidades adicionales abarcan buscadores, controles de seguridad y de acceso interno y externo, y herramientas de creación de zonas de trabajo en común para equipos que comparten proyectos a distancia, etc. Es un producto que tiene 5 años de desarrollo y distribución en el mercado, desde las primeras versiones más simples (sitios web básicos en HTML vinculados a bases de datos) hasta las herramientas sofisticadas de carga y descarga de contenidos y de actualización automática, formularios interactivos y módulos para proyectos distribuidos. También tiene componentes e interfaces para la integración con productos comerciales como Microsoft Exchange y productos de LotusNotes y varias bases de datos comerciales, como Microsoft SQL/Access/FoxPro, Oracle y DBase X and IBM DB2.
- **QDweb:** QDweb es una versión simplificada de QDnet, para usuarios finales de menor tamaño (consumidores, profesionales y autónomos, pequeñas y medianas empresas). Se utiliza para la creación y gestión de sitios web y repositorios de contenidos personales medianamente simples, incluso, por ejemplo, la gestión de datos entre sistemas individuales y comerciales (las listas de direcciones, los archivos personales, FTP, etc.).

La aplicación principal y sus diferentes componentes están desarrollados en Java y XML, con JSP y soporte para *Enterprise JavaBeans*

(EJB). Hay versiones para clientes que se instalan sobre varias plataformas y entornos (Windows, Linux, Solaris, UNIX) aprovechando los servidores de aplicaciones libres J2EE/JBoss, Apache y Tomcat (en entorno web), o propietarios como WebLogic (de BEA).

QDR-Soft adapta y personaliza el código de QDnet para sus diferentes clientes, creando soluciones hechas a medida para cada uno de ellos. A través del tiempo, QDR-Soft ha aprovechado la similitud entre los requisitos de desarrollos y personalizaciones para los clientes y, por lo tanto, ha utilizado el trabajo de diseño y análisis de algunas de estas adaptaciones para ampliar la versión actual de los módulos centrales de la aplicación.

### Los clientes y el mercado

A parte de los “pequeños” usuarios, que son consumidores de su producto “gran público”, QDweb, QDR-Soft tiene una variedad de clientes que usan sus productos y servicios. La mayoría de ellos están en el sector comercial privado (hoteles, empresas de logística, supermercados, laboratorios de investigación), pero también los hay en el sector educativo y público: institutos de estudios superiores, laboratorios públicos, organizaciones de asistencia a emprendedores, ministerios de Agricultura y de Obras Públicas.

El mercado para los productos y servicios de QDR-Soft es muy amplio, considerando la versatilidad de su producto en un mundo de negocios cada vez más “en línea” (*e-business*). Asimismo, este mercado es amplio en relación con las varias áreas de actividad de los clientes potenciales (desde el privado hasta el público, desde el sector comercial hasta el educativo y científico). Por el momento, QDR-Soft se destaca en el mercado local por la calidad de sus aplicaciones (velocidad, estabilidad, etc.) y el conocimiento del negocio del cliente, pero las ventas se realizan cada vez más en función de la adaptación al cliente y el soporte, formación y mantenimiento.

Sin embargo, hay una creciente competencia en este mercado de la gestión de contenidos, a medida que productos de empresas competidoras convergen sobre utilidades, funcionalidades y herramientas estándares. Además, hay aplicaciones que aparecen en el mismo



mercado, que se distribuyen bajo licencia de software libre (por ejemplo, Zope, Midgard, OpenCms y Red Hat CCM).

**Nota**

Detalles sobre estas aplicaciones se encuentran en:

- Zope: <http://www.zope.org>.
- MidGuard: <http://www.midgard-project.org/>.
- OpenCMS: <http://www.opencms.org/opencms/en/>.
- Red Hat CCM: <http://www.redhat.com/software/rhea/cms/>.

Éstas, por su naturaleza gratuita, están complicando la oferta comercial de QDR-Soft. Estas aplicaciones libres, aunque provengan de antiguos desarrollos hechos a medida para clientes de las empresas correspondientes (como el producto QDnet de QDR-Soft), se conforman cada vez más con estándares abiertos y se extienden a plataformas y entornos libres (Linux, Apache, Tomcat, etc.). También se ajustan a clientes y sistemas más grandes, complejos y multinacionales: la cuestión del idioma era una barrera de entrada en el mercado de QDR-Soft para varias empresas anglosajonas, ahora se desarrollan versiones libres en diferentes idiomas, como el español, el francés y el alemán.

### Derechos sobre las aplicaciones

QDR-Soft mantiene sus derechos de propiedad intelectual e industrial en los módulos centrales (o núcleo) de las aplicaciones QDnet y en todo el programa QDweb. En los desarrollos hechos a medida para clientes, la empresa cede a sus clientes la propiedad en las adaptaciones y extensiones particulares: los términos contractuales estándares de la empresa otorgan a los clientes la propiedad de cualquier desarrollo específico pagado por ellos (y la entrega del código fuente). QDR-Soft garantiza que el programa no infrinja derechos de terceros, mientras que los clientes se comprometen a respetar los derechos de terceros en relación con los contenidos pro-

cesados por la aplicación (la responsabilidad por contenidos). QDR-Soft otorga una garantía de 6 meses para la corrección de errores y adaptaciones menores.

En contrapartida, los clientes deben mantener la confidencialidad de cualquier código fuente que se les entregue y no se permite la modificación, distribución, licencia o sublicencia de cualquier módulo central de QDR-Soft instalado en los equipos del cliente. Es importante notar que los elementos personalizados de QDnet son de poca utilidad sin el núcleo de la aplicación, con el cual se vinculan estrechamente en la versión compilada y entregada.

Clausulado de un contrato típico con clientes:

#### **“Propiedad industrial e intelectual”**

“El CLIENTE reconoce todos los derechos de propiedad industrial e intelectual de QDR-SOFT sobre los módulos de programas que el QDR-SOFT ha desarrollado fuera de la labor de programación especificada en la Propuesta / Especificaciones y que han sido utilizados por QDR-SOFT como parte de los programas desarrollados exclusivamente para el CLIENTE. En consecuencia, QDR-SOFT otorga a favor de CLIENTE una licencia de uso no exclusiva de los mismos, por todo el periodo de duración de los derechos y para todos los países del mundo. Esta licencia incluye un derecho de instalación, uso y ejecución de los módulos de programas de ordenador que componen el producto en cualquier equipo informático bajo control directo del CLIENTE a los fines necesarios para la utilización del producto. Dichos programas en ningún caso podrán ser objeto de cesión por parte del CLIENTE a terceros, ni tampoco podrán llevarse a término por parte del CLIENTE actos de desmontaje, descompilación, alteración de la ingeniería del sistema, o cualquier otro que implique transformación de los mismos, salvo los que resulten necesarios o convenientes para que el CLIENTE desarrolle en el futuro su actividad. Sujeto a esta licencia y lo previsto en el párrafo siguiente, todos los derechos sobre dichos módulos no específicos permanecen propiedad de QDR-SOFT.”

“Sin perjuicio de lo anterior, una vez se haya efectuado la totalidad del pago y con efectos a partir de la terminación de este contrato o

cualquier renovación del mismo, QDR-SOFT cederá de forma exclusiva los derechos de uso, reproducción, distribución, comunicación pública y transformación derivadas de su específica labor de programación para CLIENTE de acuerdo con el contenido de la Propuesta / Especificaciones, incluyendo los datos, listados, diagramas y esquemas elaborados en la fase de análisis, el manual de aplicación, los restantes datos y materiales de apoyo; con el fin de que CLIENTE pueda realizar copias del mismo, instalarlas en cuantos ordenadores estime oportuno y utilizarlas en su actividad empresarial, así como modificar el código fuente con la finalidad de adaptarlo a sus características o necesidades específicas. Dicha cesión alcanzará a todos los países del mundo y se extenderá por todo el período de duración de los derechos. Respecto a dichos programas exclusivos, QDR-SOFT no podrá ceder el uso de los mismos a terceros ni transmitir ninguno de los derechos que tenga sobre ellos en virtud de este contrato, comprometiéndose a no divulgarlos, publicarlos, ni ponerlos, en todo o en parte, de ninguna otra manera a disposición de otras personas o empresas. Además, QDR-SOFT entregará a CLIENTE el código fuente y el código objeto de dichos programas exclusivos como máximo a los 15 días a contar desde la terminación del contrato, con el fin de que pueda ejercer los derechos cedidos.”

“Las licencias de uso de los sistemas operativos y programas cuya propiedad no se haya mencionado en los párrafos anteriores o que pertenezca a terceros ajenos a este contrato tendrán, en su caso, como beneficiario a CLIENTE, salvo que en la Propuesta / Especificaciones se establezca disposición en contrario.”

“Ninguna de las partes infringirá los derechos de terceros en la creación, desarrollo y mantenimiento de cualquier programa objeto del presente. CLIENTE será responsable de todos los materiales, obras y otros contenidos almacenados en cualquier elemento de programa objeto del presente. Ambas partes se comprometen a indemnizar a la otra parte por cualquier daño que pudiese sufrir a causa de alguna violación de los derechos de terceros.”

#### **“Garantías”**

“QDR-SOFT garantiza al CLIENTE el funcionamiento del producto y el buen fin del servicio prestado durante un período de seis (6) meses

contados a partir de la fecha en que QDR-SOFT perciba la totalidad del precio estipulado. La presente garantía comprende que el pacto que los servicios que presta QDR-SOFT serán de calidad profesional, conforme a los estándares generalmente aceptados en la industria y, en general, respondiendo al grado de pericia, destreza y conocimiento que generalmente cabe esperar en la prestación de servicios de buena reputación.”

### “Confidencialidad”

“Este contrato, sus anexos, cualquier correspondencia y documentación y los detalles técnicos y económicos contenidos en este documento o relacionados con el objeto del mismo son y serán confidenciales, incluso concluido su período de vigencia, y no serán reproducidos, difundidos ni publicados a terceros.”

“Las partes se obligan a tratar confidencialmente y a no reproducir, difundir o publicar la información acerca de las circunstancias, planes, sistemas de gestión, procesos industriales, comerciales o empresariales de la otra parte y/o cualquier otra información a la que hubieren tenido acceso en el curso de este contrato y a no divulgar ni utilizar directamente ni a través de terceras personas o empresas, la información confidencial a la que tengan acceso durante su relación contractual.”

“En especial, las partes se comprometen a no efectuar copia alguna de los documentos o archivos en los que se incluya información confidencial, a excepción de lo estrictamente necesario para la ejecución de este contrato, y a notificar cualquier pérdida de ficheros o información en cualquier formato. Esto mismo lo harán extensivo a las personas a las que les hagan llegar dicha información. En todo caso esta información confidencial será devuelta a la otra parte o destruida, a su elección, cuando una de las partes lo solicite.”

### Personal

La empresa ha sabido mantener su personal y la integridad de los equipos de trabajo a través de una buena gestión de proyectos, cur-

sos de formación continua a distintos niveles (lenguajes, análisis, entornos, administración de sistemas, gestión de proyectos, etc.) y dejando a los programadores participar en diferentes proyectos de software libre (fuera de las aplicaciones de la empresa y fuera del horario de trabajo) como parte de su estrategia de formación y satisfacción del personal.

## Modelos de desarrollo

Hasta hoy, QDR-Soft ha usado distintos procesos de desarrollo para sus proyectos, que pueden dividirse en dos etapas: una primera etapa, “centralizada” o jerárquica y una segunda etapa, más “en red”. En la primera etapa, un equipo central de desarrolladores, liderado por un jefe de proyecto y responsable de clientes, establecía el diseño, arquitectura y otras especificaciones de los proyectos. La programación de las aplicaciones, herramientas y librerías fue realizada por programadores internos y algunos externos (autónomos), según las necesidades del proyecto. El mismo equipo se encargó de la corrección de errores, la instalación y las pruebas en los equipos informáticos de los clientes. Este proceso tuvo problemas con respecto a la velocidad de adaptación de desarrollos hechos a medida, y de extensión entre equipos y clientes (*scalability*). Se generaban funcionalidades extra que el cliente no necesitaba y había demasiados errores de desarrollo.

En una segunda etapa (hasta hoy), la empresa experimentó con un nuevo proceso de desarrollo en red. En este proceso participan más programadores independientes contribuyendo a un repositorio central controlado por un equipo central de gestión del cliente y de la solución personalizada en cuestión. Este equipo central coordina el proceso de desarrollo para el cliente en particular. Hay un segundo equipo interno de “calidad” que monitorea los módulos programados por los diferentes desarrolladores (incluso la calidad y los estándares) y los incorpora en una versión modular de la aplicación central. Este modelo sufre de una falta de coordinación y estandarización (lo que provocaba problemas de interoperabilidad y de compatibilidad) y de sinergia de ideas entre desarrolladores. Resulta en un cierto nivel de duplicación de esfuerzo entre los diferentes miembros del equipo.

#### 9.1.4. Proyecto actual

En este momento de baja actividad, la empresa QDSOFT considera “empaquetar” su producto principal, realizando una limpieza, racionalización y empaquetamiento de los diferentes módulos centrales, periféricos y varias interfaces. Éste involucra un interesante trabajo de reingeniería. Asimismo, la empresa quiere que la última versión se conforme a los estándares SOAP y CORBA, y a los de arquitectura de web-services. Además, hay varios módulos nuevos que se quieren desarrollar e incorporar, por ejemplo, para cumplir con nuevas normas de firma digital, de facturación telemática y de seguridad de datos (por ejemplo, bajo la legislación europea de protección de datos personales). Esta “estandarización” y ampliación de la aplicación nuclear permitiría a QDR-Soft concentrar sus esfuerzos en los procesos de negocio que permiten lograr mayores beneficios, es decir, la adaptación y personalización al cliente y el soporte, formación y mantenimiento de la aplicación.

Este trabajo requiere la estructuración, optimización, “flexibilización” (para la instalación a medida), apertura y pruebas extensivas de distintos módulos actualmente separados de la aplicación principal en diferentes entornos. Por ejemplo, se establecerán módulos estándar para la gestión de distintos tipos de contenidos, nuevas interfaces con aplicaciones y formatos comerciales propietarios (para la importación y exportación de datos), varias herramientas de creación de zonas de trabajo compartido (por ejemplo, plataformas de proyectos distribuidos) y controles de acceso e integridad de contenidos con componentes de firma digital, etc.

El paquete final se destinará a usuarios comerciales, para una instalación por el cliente rápida y eficiente. Además, incorporará varios *tools* y *toolkits* con herramientas de personalización de formularios, *workflow* y otros formatos y contenidos, que puedan manejar el personal interno de los clientes o consultores externos (de QDR-Soft) para que el paquete se comercialice más como un desarrollo “hecho a medida”.

Uno de los objetivos de la empresa es beneficiarse de los conocimientos de programadores externos en estándares y sistemas fuera a la empresa (y aplicaciones paralelas, competidoras o complemen-

tarias), para crear una aplicación que se podrá considerar un estándar o líder en el segmento. Asimismo, alcanzará una utilidad mayor por ser compatible e integrable con otras aplicaciones utilizadas por sus clientes, sean comerciales o de software libre.

Para la nueva versión de la aplicación, la empresa está considerando un modelo de desarrollo libre o abierto. Esto no solamente atraerá programadores con conocimientos pertinentes y una mayor difusión y estandarización, también reducirá los costes de desarrollo para la empresa. Dado que QDnet es uno de los programas líderes en el segmento de la gestión de contenidos, la empresa considera que atraerá a varios programadores “libres”. Sin embargo, la empresa quiere mantener el control y la coordinación del proyecto, erigiéndose en “líder” o “desarrollador principal” de la aplicación y su desarrollo libre –un papel similar al que cumple Linus Torvalds (y su equipo de lugartenientes) en relación con Linux. En esta posición, podrá determinar el código que se incorpore al núcleo existente y decidir el progreso de la aplicación. Asimismo, la empresa estima que tendrá una ventaja competitiva en la venta y distribución del software y cualquier servicio relacionado con el mismo, una vez que el código sea libremente accesible.

En estas circunstancias, QDR-Soft está considerando qué tipo de licencia debería usar para el nuevo paquete.

### 9.1.5. Preguntas y actividades

Preguntas sobre el caso

- 1) Las licencias de software libre se diferencian entre ellas en varios aspectos. ¿Cuáles son los elementos principales de las licencias que se consideran en el momento de la elección de licencia?
- 2) La selección de una licencia para cualquier proyecto o aplicación nueva afectará o tendrá implicaciones importantes para varios aspectos de un negocio. ¿En qué áreas tiene implicaciones esta selección de una licencia en el caso de QDR-Soft? Explicad cómo están afectadas estas áreas.
- 3) Por consiguiente, ¿cuáles son las motivaciones y factores principales que determinan la elección de licencia por QDR-Soft?

- 4) Investigad diferentes aplicaciones de gestión de contenidos de software libre (algunas se mencionan en el texto) para comentar sus licencias de distribución y sus modelos de desarrollo y de negocio.
- 5) ¿A qué riesgos legales se expone la empresa, al distribuir y prestar servicios en relación con una aplicación programada según un modelo de desarrollo libre?
- 6) ¿Qué tipo de licencia (o política de licencia) recomendáis para QDR-Soft? ¿Por qué? ¿Cuáles serán las condiciones más importantes de dicha licencia para QDR-Soft? ¿Y para los nuevos clientes?
- 7) ¿Cuáles son los argumentos a favor o en contra de la creación de una nueva licencia libre (o la modificación de una existente)?
- 8) ¿Qué deberá hacer la empresa para gestionar los derechos en la aplicación en función de la licencia elegida (propietaria o libre)?
- 9) ¿QDR-Soft podría cambiar una licencia por otra, durante la vida de las aplicaciones? ¿Qué complicaciones encontraría?
- 10) ¿Cuál es el efecto sobre los clientes actuales de QDR-Soft si ésta elige una licencia libre como la BSD o la GPL para la nueva versión de la aplicación?
- 11) Si QDR-Soft sigue trabajando y extiende la aplicación QDweb para gran público, ¿cómo será afectado por la utilización de una licencia libre para la aplicación comercial QDnet? ¿Recomendaríais que la empresa libere también el código de QDweb?

### Extensión y generalización

- 12) Desde el punto de vista del desarrollador, se puede considerar que una licencia es restrictiva si prohíbe o impide ciertas acciones de explotación del software (copia, modificación, comercialización). Por ejemplo, la GPL es una licencia restrictiva, porque obliga a mantener cualquier modificación y combinación libre; la BSD es liberal, porque permite cualquier uso y aplicación. ¿Qué tipo de proyecto se prestará a una licencia restrictiva y una más liberal?



- 13) Analizando éste y otros casos, estableced una matriz de características de proyectos que contribuya a la selección de una licencia.
- 14) En el proyecto actual, habrá elementos de código preexistente y módulos y líneas de código nuevo. ¿Qué diferencias hay (en términos de riesgos y estrategia legal) entre el desarrollo y distribución de código nuevo, y la liberación de código de una aplicación existente? ¿Cuáles son los riesgos y los obstáculos legales?
- 15) Si la nueva aplicación QDnet debe ser capaz de integrarse con otras aplicaciones, su licencia debe ser compatible con la de aquel otro software. ¿Qué es el concepto de “compatibilidad entre licencias”? Tomando dos licencias libres comunes, la GPL y la BSD, ¿cuáles son los problemas de uso de una y otra en relación con el software desarrollado por QDR-Soft?
- 16) ¿Cuáles son las consecuencias para QDR-Soft en términos de modelo de desarrollo y de negocio, provocadas por la elección de una licencia u otra? ¿Qué estructura y políticas empresariales recomendaríais?
- 17) Si QDR-Soft utiliza una licencia libre como GPL, explique si recomendaría o no a la empresa que establezca una política especial de conducta para los empleados y colaboradores externos. ¿Qué incluiría en dicha política?

## 9.2. Iván Lasser. Desarrollador

Objetivos de estudio:

- Aprender cómo defender los derechos relativos a la creación y distribución de software libre.
- Considerar el ámbito de los usos permitidos de software libre bajo GPL.

Métodos de estudio:

- Lectura del caso.

- Establecer una estrategia para defender los derechos como autor de programa.
- Redacción de una carta exigiendo el cumplimiento de la licencia.
- Redacción de una defensa, defendiendo los derechos de uso otorgados por la licencia GPL.

### 9.2.1. Introducción

Muchos de los desarrolladores de software libre son individuos que programan por su cuenta: son autores-creadores, y participan de manera independiente en los grandes proyectos libres (Linux, Debian, KDE, Openoffice.org, etc.). No se benefician de la protección y la infraestructura “legal” de una organización comercial o pública para defender sus derechos frente al abuso potencial de sus obras. Por lo tanto, cuando publican código “libre”, lo deben hacer con cierto cuidado y establecer una estrategia para protegerse. Este caso ilustra algunos problemas que puedan surgir e intenta aportar datos para fomentar una reflexión y un trabajo sobre posibles estrategias de protección.

### 9.2.2. Iván Lasser

Iván Lasser es ingeniero en sistemas y desarrollador de software independiente. Ha trabajado en varios proyectos comerciales como empleado de empresas grandes y como autónomo para clientes particulares. En su tiempo libre, estudia programas de software libre como Linux, Mozilla, Sendmail, OpenOffice, JBoss, Samba, etc. y programa para sí mismo.

Particularmente, ha desarrollado una serie de módulos, extensiones e interfaces, en Java y otros lenguajes, que permiten interconectar sistemas operativos diferentes (Windows, Linux, OS2). Iván ha publicado su código en Internet, aplicándole la licencia GPL y reservándose el derecho de distribuir las mismas aplicaciones de manera propietaria o comercial. Ha recibido comentarios y sugerencias de varios programadores de Estados Unidos y Alemania para optimizar el código y un desarrollador de Venezuela le ha enviado algunos

componentes y líneas de código para agregar a sus interfaces. El desarrollo sigue de manera abierta, con aproximadamente 10 personas de diferentes países interesadas y siguiendo el progreso.

### 9.2.3. Controversia

A través de amigos que trabajan en varias empresas del sector, Iván se ha percatado de que dos de estas empresas (empresa A, donde Iván ha trabajado como empleado en el pasado reciente y empresa B) han incorporado partes de su código a aplicaciones propias.

- La primera empresa (A) utiliza varios programas de software libre en sus propios sistemas y el departamento de sistemas de la empresa ha aprovechado el código de Iván publicado en Internet para descargarlo e integrarlo con sus aplicaciones principales. Las extensiones e interfaces de Iván permiten vincular las aplicaciones y los datos del servidor de web (Apache) y el *back office* (un AS-400 con OS2) y transferir datos entre ellas, principalmente relativos a la facturación y al servicio de análisis financiero.
- La segunda empresa (B) ha contratado los servicios de una empresa consultora de informática (C) para desarrollar una nueva extensión de su plataforma de comercio electrónico (Apache, WebLogic, Java), que debe vincularse con bases de datos almacenadas en los servidores centrales (Windows NT, SQLServer). La consultora tiene un contrato de desarrollo en los términos comerciales habituales, otorgando una cesión total relativa al código programado por la consultora y cláusulas de confidencialidad para mantener los secretos de negocio y métodos de desarrollo de la consultora C. Algunas cláusulas del acuerdo de desarrollo entre las empresa B y C figuran a continuación.

Cláusulas relevantes del contrato entre empresas B y C:

#### “OBJETO DEL CONTRATO”

“El objeto del presente contrato es la prestación de los servicios de programación que el cliente (empresa B), con arreglo a los términos y condiciones establecidos en las cláusulas del presente documento,

encarga al prestatario (empresa C), que acepta, para que a través de sus empleados le preste las horas de servicios profesionales que sean necesarias, estableciéndose dicha prestación de servicios mediante la realización de la labor de programación, consistente en la elaboración de la aplicación informática descrita en el anexo I, de acuerdo con las instrucciones y en base a la información y datos que aporte el cliente.”

#### “OBLIGACIONES DEL PRESTATARIO”

- “1) El prestatario se obliga a prestar los servicios contratados por el cliente bajo las instrucciones y supervisión del mismo orientados al desarrollo de una aplicación informática especificada en el anexo I con la máxima diligencia y conforme a las mejores prácticas existentes en el mercado al día de la fecha del presente contrato.”
- “2) Una vez finalizado el trabajo de programación, el prestatario entregará al cliente el código fuente y el objeto de la aplicación informática resultante, así como todo el material previo, incluidos los diseños y todas las transformaciones y actualizaciones de la misma, la descripción detallada y el manual de la aplicación informática, conteniendo las normas de uso en el formato y en las condiciones especificadas en el anexo I.”
- “3) El prestatario se compromete a elaborar y entregar al cesionario o usuario el manual de uso de la aplicación informática.”
- “4) En el caso de que sea necesaria para la elaboración de la aplicación informática la utilización de aplicaciones o componentes informáticos sobre los que recaigan derechos de terceros, el prestatario acreditará que puede usar legítimamente dichos componentes y aplicaciones para la prestación de los servicios objeto del presente contrato.”

#### “PROPIEDAD INTELECTUAL”

- “1) El prestatario reconoce los derechos de propiedad intelectual del cliente sobre el resultado de la labor de programación del prestatario que renuncia a ejercitar cualesquiera derechos sobre el mismo. La titularidad de la aplicación informática alcanza

a todos y cada uno de los trabajos susceptibles de ser objeto de propiedad intelectual e industrial realizados en relación con la aplicación informática. En cualquier caso el prestatario cede al cliente en exclusiva los derechos de explotación que pudieran derivarse de la aplicación para todo el territorio mundial y por un periodo de setenta (70) años.”

- “2) Dichos derechos protegen tanto el programa de ordenador que pueda resultar, como los datos, listados, diagramas y esquemas elaborados en la fase de análisis, el manual de aplicación, los restantes datos y los materiales de apoyo.”
- “3) El prestatario reconoce que el producto de su labor de programación del prestatario pertenece exclusivamente al cliente y se compromete a no poner la aplicación informática de ninguna otra manera a disposición de otras personas, salvo de los empleados de su empresa que tengan que utilizarlo a los efectos de lo previsto en el presente contrato.”
- “4) El prestatario será también responsable del incumplimiento de estas obligaciones por parte de sus empleados o de terceros que accedieran a él por negligencia.”

#### “DURACIÓN Y ALCANCE DE LA GARANTIA”

“La garantía se establece por un plazo de sesenta (60) días naturales a partir de la fecha de la instalación del programa informático en el equipo del cliente. Si el cliente no ha notificado la existencia de defectos al prestatario durante el referido plazo, se considerará que está conforme en todos los aspectos con el funcionamiento del programa, renunciando, a partir de entonces, a cualquier reclamación.”

#### “CONFIDENCIALIDAD”

- “1) Los conocimientos y demás informaciones transmitidas entre las partes constituyen información propia y confidencial del prestatario y del cliente y su protección es de la máxima importancia. Por ello, ambas partes asumen la obligación de guardar dicha información como confidencial, adoptando las medidas apropiadas para asegurar que solamente aquellas personas autorizadas tengan acceso a la misma, entendiendo como personas

autorizadas aquellos empleados de las partes que lo necesiten para el desarrollo de la actividad objeto de este acuerdo.”

- “2) No queda comprendida dentro de la obligación de confidencialidad aquí prevista la información recibida por una de las partes que: (i) ya sea conocida por tal parte antes de su transmisión y tal parte pueda justificar la posesión de la información; (ii) sea información de general o público conocimiento; (iii) haya sido recibida de terceros legítimos titulares de la misma, sin que recaiga sobre ella obligación de confidencialidad; (iv) haya sido desarrollada independientemente por la parte que la recibe sin haber utilizado total o parcialmente información de la otra parte; (v) haya sido su transmisión a terceros aprobada o consentida previamente y por escrito por la parte de la que procede la información; y/o (vi) haya sido solicitada por una autoridad administrativa o judicial.”
- “3) La información confidencial no podrá ser revelada a terceros ni durante la vigencia del presente contrato ni a la finalización del mismo durante un periodo de dos años.”

#### “RELACIÓN”

“La relación entre las partes tiene exclusivamente carácter mercantil, no existiendo vínculo laboral alguno entre el cliente y el personal del prestatario que eventualmente esté prestando sus servicios en el domicilio social de aquél.”

#### “ANEXO I - CONDICIONES ESPECÍFICAS DE LA APLICACIÓN INFORMÁTICA”

“Aplicación informática: datos identificativos, especificaciones técnicas, funcionalidades, características del programa a desarrollar, descripción del servicio, módulos que comprende, etc.”

Iván considera que las tres empresas están haciendo un uso comercial no autorizado de su código, dado que las aplicaciones de las empresas son propietarios y no se publica su código. Según él, la licencia GPL bajo la cual distribuyó el código en Internet exige que cualquier usuario (persona que realiza una explotación del código) mantenga “abiertos” los sistemas que lo incorporan.

#### 9.2.4. Preguntas y actividades

Vosotros sois “abogados” amigos de Iván, consultados por él para defender sus derechos.

- 1) ¿Cuáles son los derechos de Iván en las aplicaciones en cuestión?  
¿Qué derechos ha mantenido y qué derechos ha cedido?
- 2) ¿Hay otros derechos que intervienen en el asunto?
- 3) ¿Qué incumplimientos han acaecido, según vosotros?
- 4) ¿Qué se puede hacer para ayudar a Iván? ¿Qué puede exigir? Estableced una estrategia (plan de acción) para defender los derechos de Iván.

Trabajo de equipo:

- 5) EQUIPO “Iván”: redactad una carta a las empresas A y B por parte de Iván, exigiendo el cumplimiento de sus obligaciones relativas al código de Iván.
- 6) EQUIPO “empresa A”: asesora a la empresa A.
  - (Sin importar que lo consideréis válido o no), redactad la respuesta (defensa) de A, rechazando sus alegatos e indicando por qué consideráis que puede usar libremente el código de Iván.
  - Si A está en incumplimiento de la licencia GPL, ¿cómo podría haber evitado estos problemas?
- 7) EQUIPO “empresa B”: asesora a la empresa B:
  - (Sin importar que lo consideréis válido o no), redactad la respuesta (defensa) de B, rechazando sus alegatos e indicando por qué vosotros consideráis que puede usar libremente el código de Iván.
  - Si B está en incumplimiento de la licencia GPL, ¿cómo podría haber evitado estos problemas?

8) EQUIPO “empresa C”: asesora a la empresa C:

- (Sin importar que lo consideréis válido o no), redactad la respuesta (defensa) de C, rechazando sus alegatos e indicando por qué vosotros consideráis que puede usar libremente el código de Iván.
- Si C está en incumplimiento de la licencia GPL, ¿cómo podría haber evitado estos problemas?

### 9.3. LibreSolutions. La distribución de software libre

Objetivos de estudio:

- Entender los aspectos legales asociados a la distribución comercial de paquetes de software libre y la prestación de servicios relacionados.
- Discernir los riesgos legales de estas actividades y establecer estrategias legales para minimizar estos riesgos.

Métodos de estudio:

- Estudio del caso de la creación de una nueva iniciativa para la distribución e instalación de software libre y la prestación de servicios asociados.
- Análisis de las condiciones, riesgos y obstáculos para esas actividades.
- Creación de la documentación necesaria para realizar esas actividades (contratos, pactos estándares, garantías, protocolos, políticas internas, etc.).

#### 9.3.1. Introducción

En este apartado presentamos la creación de una nueva empresa de distribución de software libre, en paquetes para desarrolladores



y para empresas usuarias. La creación y la organización de esta empresa ponen de manifiesto varios problemas legales. Este estudio de caso se propone ilustrar algunos de los factores y riesgos asociados y ayudarnos a establecer unas pautas o estrategias adecuadas para minimizar los riesgos y maximizar los beneficios (legales y económicos).

### La distribución de software libre

En el principio del desarrollo del movimiento de software libre, pocos tenían fe en la posibilidad de fundar una empresa comercial basada en el modelo de negocio de la libre distribución de software libre. Hasta entonces, la mayoría de las empresas involucradas en el sector de la informática (IBM, Microsoft, Oracle y consultoras de software de todo tamaño, etc.) obtenían sus ingresos vía derechos de licencia y regalías por la venta y distribución de aplicaciones en formato de código binario. Estas distribuciones restringían y todavía hoy restringen la mayoría de los usos de los programas, como la copia, la modificación y la redistribución del código. Las actualizaciones y mejoras periódicas de las aplicaciones permiten a estas empresas mantener a sus clientes y realizar mayores beneficios. La idea de distribuir código que fuera “abierto” o de libre acceso –no necesariamente gratuito, pero libremente modificable y redistribuible– parecía incompatible con ganancias a medio y largo plazo.

Sin embargo, en los últimos años, varias empresas han podido establecerse en este mercado, vendiendo y distribuyendo aplicaciones de software libre cuyos usos no son restringidos: los clientes pueden realizar varias copias de la misma aplicación, instalarla en varios equipos, modificarla para adaptarla a sus necesidades y redistribuirla a terceros. Empresas como Mandrake, SuSe y Red Hat, entre muchas, muestran que el otorgamiento o transmisión de esos derechos a sus clientes no impide la obtención de beneficios y la expansión de su negocio. El hecho de que el cliente pueda revender o distribuir gratuitamente mil veces el mismo programa que haya comprado una vez de Mandrake no implica que Mandrake pierda mil clientes.

Tal como han reconocido estas empresas, la venta y distribución de software no es únicamente cuestión de descargar un programa e instalarlo en un equipo. A menudo hace falta considerar las versiones y

las interdependencias entre distintas aplicaciones para que sean interoperables y se ejecuten sin problemas. Por ejemplo, los programas para actuar en Internet requieren la instalación de un servidor web (Apache, etc.), las aplicaciones comerciales necesitan un servidor de aplicaciones (JBoss, Weblogic) y los programas en Java usan el Java Runtime Engine con su Virtual Machine. Por lo tanto hay toda una infraestructura informática que hay que considerar, descargar, instalar, configurar e integrar antes de poder usar software libre de manera completa. Asimismo, no todas las aplicaciones son estables o están libres de errores, aunque en general el software libre tiende a tener menos errores que el software propietario. Las empresas comerciales con poca experiencia en el campo de la informática no tienen ni el tiempo ni el personal ni las competencias para investigar en Internet a través de los foros y sitios dedicados a software libre para resolver estos problemas y realizar cambios de código para adaptar a sus necesidades los paquetes de software libre disponibles.

#### Lectura complementaria

Podéis leer, por ejemplo: *"The Cathedral and the Bazaar"* de E. Raymond sobre las ventajas del modelo de desarrollo de software libre.

Estas empresas distribuidoras de software libre basan sus ingresos en la prestación de servicios anexos al software libre: asesoramiento, instalación, adaptación, formación y mantenimiento, entre otras actividades. La complejidad actual de muchas de las aplicaciones libres y la frecuente necesidad de realizar adaptaciones para personalizar estos programas en función de los requerimientos particulares del cliente, hacen que estos servicios sean no solamente útiles sino a menudo necesarios para compradores comerciales y particulares. Esa exigencia vale tanto para la distribución de sistemas operativos y programas libres a usuarios finales como para la distribución de aplicaciones y herramientas utilizadas en desarrollos informáticos, destinadas a programadores individuales y empresas consultoras que elaboran aplicaciones para terceros. Por lo tanto hay toda un área de productos y servicios asociados al software libre que se pueden vender y prestar para que el usuario privado o comercial pueda aprovechar las ventajas del mismo.

Sin embargo, estas actividades no están exentas de riesgos: estas empresas basan sus actividades sobre la distribución, la instalación, el uso y la manipulación de software que no es suyo. Asimismo, están restringidas o limitadas por los derechos otorgados y obligaciones impuestos por los titulares de la propiedad intelectual a través de las licencias asociadas al software libre y el derecho de propiedad intelectual no solamente de su país sino de cualquier país donde provenga y se ejecute el

software en cuestión. Por lo tanto, cualquier empresa que se aventure en este sector debe tomar en consideración varios factores legales que puedan afectar a su negocio.

### 9.3.2. La empresa LibreSolutions: el concepto

Los fundadores de la empresa LibreSolutions se han percatado de que muchas herramientas y aplicaciones de software libre no son fáciles de usar y manejar, sobre todo, para empresas sin personal dedicado a la informática, pero también para departamentos de sistemas de grandes empresas y consultoras informáticas cuyo personal se ha formado en el mundo de los lenguajes, herramientas y aplicaciones propietarias y/o de otra generación (C y C++, Visual Basic, etc.). Por lo tanto, se proponen emprender la distribución de varias aplicaciones y herramientas de software libre para empresas usuarias y consultoras informáticas y proveer servicios relacionados para facilitar el uso de dicho software tanto en el comercio como en el mundo del desarrollo de aplicaciones a medida. Se concentrará en la creación de paquetes de aplicaciones interoperables con herramientas de instalación y personalización de uso fácil y la provisión de servicios de alto valor añadido que se comentan a continuación.

#### Nota

La empresa LibreSolutions se modela parcialmente sobre la empresa real EJB Solutions, en:

<http://www.ejbsolutions.com>

Investigad en Internet otras empresas del mismo estilo y fijaos en sus condiciones de distribución y comercialización.

Por consiguiente, se propone crear un paquete de software o *suite* que genere una arquitectura unificada de aplicaciones y proyectos de aplicaciones libres. Esto servirá como base para una multitud de servicios informáticos para los distintos usuarios. La mayoría de las aplicaciones y los servicios se orientarán sobre todo a Internet, como sitios web, intranets y extranets, comercio electrónico, web-services, etc.). Para sus clientes comerciales, los productos y servicios permitirán la migración de sus aplicaciones a un entorno homogéneo y compatible de software libre. Para los clientes desarrolladores, permitirán crear de manera fácil soluciones informáticas (libres y propietarias) para sus propios clientes.

### Un paquete para desarrolladores: la oferta técnica

En particular, los fundadores quieren crear una “distribución optimizada” de software libre orientado a desarrolladores de aplicaciones, para ofrecerles un entorno de desarrollo integrado. Incluirá un paquete de herramientas de software libre interoperables que facilite el diseño, programación, documentación e instalación de aplicaciones comerciales en Java y otros lenguajes. Incluye módulos para facilitar la integración de proyectos en una arquitectura coherente y para realizar una documentación completa. Por lo tanto, comporta todas las herramientas para crear esta arquitectura.

Esto incluye varias herramientas de desarrollo, como un gestor de *bugs*, un sistema de control de versiones, un analizador de código fuente, una herramienta de modelaje y de documentación (*diagram generator*, *multi-format documentation generator*), un entorno y herramientas de pruebas, *scripts* de compilación e integración (*build and deployment*) y utilidades varias.

Asimismo, el paquete incluye varios componentes para integrar en aplicaciones para clientes: servidores web, de correo y de aplicaciones, un *container* para aplicaciones web, motores de bases de datos, un analizador de base de datos, buscadores (de texto), un registro de actividades web (*web logger*), un sistema caché de objetos distribuidos, herramientas para foros de discusión, para la generación y la gestión de plantillas, sitios *wiki* y RSS, etc.

Estos programas ya están disponibles en Internet: la idea de LibreSolutions es realizar cambios y optimizaciones de los procesos estándares de instalación y configuración para que las aplicaciones y herramientas estén integradas, probadas y listas para su ejecución al finalizar la instalación.

#### Nota

Las direcciones de Internet de los productos enumerados se encuentran citadas al final de este estudio de caso. Investigad algunas de las aplicaciones y sus formas de distribución (licencias, modelos, empresas involucradas, etc.).



En particular, la distribución incluye:

- **Bugzilla** (una herramienta de gestión y corrección de errores).

- **CVS** (una herramienta de gestión de versiones de software).
- **GNAT** (una implementación del lenguaje de programación Ada).
- **Emacs y Vim** (editores de texto).
- **GNU Compiler Collection** (GCC, un paquete de herramientas para la compilación para C, C++ y otros lenguajes).
- **JBoss** (un servidor de aplicaciones compatible con J2EE).
- **TAO** ("The ACE ORB", una implementación de ORB de CORBA).
- **Perl y Python** (lenguajes de programación y para scripts).
- **PHP** (pre-procesador de *hypertext* para el desarrollo de sitios web).
- **Zope** (una herramienta de creación de sitios web).

El paquete incluye otras herramientas desarrolladas por la propia empresa, para "empaquetar" el software distribuido:

- Un instalador gráfico: una herramienta que permite elegir las aplicaciones para instalar y sus dependencias.
- Una base de conocimientos para proyectos (*Knowledge Base*) y otras herramientas para la gestión de proyectos y del desarrollo.
- Documentación, guías rápidas y *tutoriales* (módulos interactivos para la formación rápida del usuario).

Se considera que el paquete LibreSolution Developer aportará varios beneficios para desarrolladores:

- La infraestructura de desarrollo y ejecución se genera instantáneamente: los programadores no deberán luchar con distintas versiones incompatibles o con archivos de configuración incompletos o incorrectos.
- El paquete incluirá las aplicaciones y lenguajes de última generación y de mayor uso: Apache, JBoss, Tomcat, Struts, Ant, PHP, Python, Eclipse, Hibernate, Castor, MySQL, PostgreSQL, etc. (más detalles más adelante).
- Los desarrolladores se ahorrarán varias horas en la fase de integración e implementación de código de proyecto.
- Se podrán incluir proyectos o módulos de ejemplo (sitios web, bases de datos, gestión de accesos, etc.) que sean modificables, para permitir el aprendizaje.
- La empresa consultora podrá establecer foros y aplicaciones de gestión de proyecto, foros de discusión, gestión de versiones y de conocimiento, *weblogger* y sitios *wiki*.

Además, hay varias ventajas para la gestión de cualquier desarrollo de proyecto utilizando software libre:

- El paquete permite la gestión y control del código fuente, el monitoreo y seguimiento de problemas y la fácil creación de documentación.
- Se produce una mayor productividad en los programadores y en el desarrollo de los proyectos por la compatibilidad y la interoperabilidad de las aplicaciones y herramientas y la facilidad de instalación.
- Menor TCO (coste total de la aplicación): no hay costes de mantenimiento y licencias.

## Organización de la oferta comercial

Para los clientes usuarios, el paquete LibreSolution Enterprise (que incluye varias herramientas para la infraestructura empresarial y sus aplicaciones en red y en la red Internet) será una solución de software libre que satisface la mayoría de las necesidades informáticas para las redes empresariales y de las organizaciones públicas. Creará un marco integral para la instalación y la gestión de su infraestructura informática, con funcionalidades de monitoreo, informes y notificaciones relativas a la red (la propia red y los dispositivos, sistemas y aplicaciones instalados en ella). Además incorpora aplicaciones y componentes para varios servicios en la red (correo electrónico, sitios web, gestión de contenidos) y para reforzar la seguridad. Por lo tanto, incluye las aplicaciones enumeradas a continuación.



### Sistemas operativos:

- **GNU/Linux** de Mandrake, SuSe y Red Hat.
- **BSD/Linux** de BDSi.

### Servicios de red:

- **Apache** (servidor web).
- **Bind** (servicio de nombre de dominio).
- **Mailman** (gestión de listas de correo).
- **Samba** (para la interoperabilidad con clientes Windows, estableciendo un servidor de archivos e impresión compatible con Windows).
- **Sendmail y Postfix** (servidores de correo).
- **Jahia** (servidor de portal de Internet y gestión de contenidos en Internet).

Aplicaciones para equipos de sobremesa y comerciales:

- **Evolution** (cliente de correo electrónico y gestión de contactos y calendario).
- **Mozilla** (navegador de web y cliente de correo).
- **OpenOffice.org** (paquete de ofimática con software para un procesador de textos, una planilla de cálculo, y para presentaciones).
- **The GIMP** (editor de imágenes de bitmap).
- **OpenCms** (gestión de contenidos).
- **MySQL y PostgreSQL** (bases de datos).

Infraestructura para la interfaz de usuario gráfica (GUI):

- **GNOME** (entorno de ofimática).
- **KDE** (idem).
- **XFree86** (una infraestructura gráfica que implementa el sistema X Window).

Herramientas de seguridad:

- **Nessus** (un escáner para la seguridad en redes).
- **Nmap** (un escáner para la configuración de red).
- **OpenSSH** (un alternativo seguro de Telnet, RCP y FTP).
- **OpenSSL** (una librería con los protocolos de seguridad SSL y TLS).
- **Snort** (para la detección de intrusiones).



El paquete incluye una herramienta de instalación, configuración y personalización a través de un navegador de web y *tutoriales* para cada aplicación y conjunto de aplicaciones. Todos los productos se comercializarán con 30 días de soporte telefónico gratuito y 60 días de garantía de devolución.

### Servicios adicionales

Para complementar la oferta de software, LibreSolutions considera prestar varios servicios de consultoría y desarrollo, formación y mantenimiento. Se puede distinguir dos tipos de servicios: para clientes finales y para desarrolladores.

Para clientes finales, LibreSolutions se propone ofrecer:

- **Servicios de soporte de diferentes niveles (básico, medio, alto):** soporte para la instalación, una ayuda en línea y por e-mail, la formación y la corrección de errores.
- **Servicios de implementación y ejecución de software libre para clientes que se inician con software libre:** la consultoría sobre la implementación de la infraestructura de software libre en las organizaciones. Análisis de las necesidades, comparación con productos comerciales, elección de las aplicaciones necesarias, asistencia en su instalación y mantenimiento y la migración de programas empresariales.
- **Servicios de ingeniería general y dirección o gestión de proyectos:** proveer experiencia y asistencia para organizaciones ya involucradas en software libre y la migración, adaptación o integración de programas existentes a plataformas y entornos libres. Incluye información sobre las mejoras prácticas, la optimización de los programas instalados y las mejoras o actualizaciones posibles y la solución de dificultades (por ejemplo, *performance*, estabilidad, funcionalidades o configuración) en línea o presencial.
- **El desarrollo de aplicaciones a medida:** servicios de análisis, desarrollo, implementación, documentación y formación para clientes de aplicaciones personalizadas a base de software libre.

- **Servicios de formación:** cursos, seminarios, talleres y otras formas de formación (por ejemplo, en línea) en el tema de software libre para clientes. Incluirá cursos de certificación como “instalador” o “ingeniero” LibreSolutions que permite crear asociaciones con empresas locales para formar una red de distribución y asesoría.

Para los clientes desarrolladores, LibreSolutions ofrece servicios de apoyo relacionados con el paquete “Developer”. Los servicios incluirán:

- Cursos, seminarios y talleres sobre el mismo paquete y las aplicaciones incluidas (para distintos niveles).
- Información sobre cada aplicación y sus usos, dependencias y competidores.
- La documentación completa sobre las aplicaciones de software libre y las de ejemplo.
- Unas tablas (“marco”) de diseño de proyectos.
- Servicios de consultoría sobre todos los aspectos de desarrollo de programas en Java y otros lenguajes, el diseño, planificación y gestión de proyectos.
- La asistencia en la migración a entornos libres y el soporte técnico, formación y mantenimiento.
- Información y asistencia en la aplicación de estándares y las prácticas de calidad.
- Asistencia en la gestión de la productividad, metodologías de desarrollo y mejores prácticas.

### 9.3.3. Conclusión

La empresa LibreSolutions se posiciona en el mercado como un proveedor de soluciones de software libre de tipo “llaves en mano” y “hechas a medida”, para clientes comerciales y desarrolladores de

software. Debido a la complejidad de los sistemas informáticos actuales, su oferta comercial y técnica es bastante complicada: incluye software que se usa en todos los aspectos del negocio electrónico, desde la infraestructura de redes empresariales hasta interfaces y herramientas de sobremesa para usuarios no técnicos.

Tratará de imitar el éxito de empresas de software libre como Mandrake y Red Hat, en su propio mercado local. Sin embargo, a pesar de los logros de dichas empresas, este modelo de negocio no está exento de riesgos legales (y técnicos), cuya consideración es fundamental para el éxito a largo plazo de LibreSolutions.

#### 9.3.4. Preguntas y actividades

- 1) La empresa LibreSolutions se propone distribuir varias aplicaciones en un mismo paquete. Estudiad algunas de las licencias asociadas con las aplicaciones en consideración. ¿Son todas licencias libres “estándares” (dentro de la definición de OSI)? ¿Ellas permiten a la empresa realizar las actividades propuestas? ¿Cuáles son las obligaciones impuestas sobre la empresa?
- 2) Entre las licencias, ¿hay algunas licencias especiales? ¿Cuáles son las cláusulas especiales y por qué se usan?
- 3) Estableced una tabla con las licencias principales y sus compatibilidades e incompatibilidades y particularidades.
- 4) Al distribuir aplicaciones de software libre y prestar servicios relacionados, ¿a qué riesgos legales se expone la empresa? Comentad en detalle los aspectos relevantes a:
  - a) La creación de la empresa, referentes al modelo de negocio, la dirección/gestión de la empresa, el marketing y el personal.
  - b) La distribución de software libre en formato de CD-ROM y en Internet (garantías, responsabilidad, licencias, etc.).
  - c) La prestación de servicios relacionados con el software libre distribuido.
- 5) ¿Cuáles son los riesgos legales para sus clientes? ¿Cómo pueden minimizarse?

- 6) ¿Qué estrategias recomienda para la distribución de los paquetes y para la prestación de servicios?
- 7) ¿Qué efectos tienen las licencias del software considerado sobre el modelo de negocio de LibreSolutions y sus contratos y licencias de propiedad intelectual con los clientes?
- 8) ¿Qué tipo de contrato/licencia usaría LibreSolutions para la distribución de sus paquetes? ¿Cuáles son las cláusulas principales y las razones para ellas?
- 9) ¿Qué contratos se requieren para la prestación de servicio? Haced una lista de las cláusulas más importantes. ¿Qué licencia usaría para los desarrollos propios para clientes?

### Direcciones web de los productos citados

<http://www.bugzilla.org/>  
<http://www.cvshome.org/>  
<http://www.gnat.com/>  
<http://www.gnu.org/software/emacs/emacs.html>  
<http://www.vim.org/>  
<http://www.fsf.org/software/gcc/gcc.html>  
<http://jboss.org/>  
<http://www.theaceorb.com/>  
<http://www.perl.org/>  
<http://www.python.org/>  
<http://www.php.net>  
<http://www.zope.org/>  
<http://www.apache.org>  
<http://www.isc.org/products/BIND>  
<http://sourceforge.net/projects/mailman>  
<http://www.samba.org/>  
<http://www.sendmail.org/>  
<http://www.postfix.org/>  
<http://www.jahia.org/jahia/Jahia>  
<http://ximian.com/products/evolution>  
<http://www.mozilla.org/>  
<http://www.openoffice.org/>  
<http://www.gimp.org/>  
<http://www.opencms.org/>

<http://www.mysql.com/>  
<http://www.postgresql.org/>  
<http://www.gnome.org/>  
<http://www.kde.org/>  
<http://www.xfree86.org/>  
<http://www.nessus.org/>  
<http://www.insecure.org/nmap>  
<http://www.openssh.org/>  
<http://www.openssl.org/>  
<http://www.snort.org/>

## 9.4. MySQL – Progress Software

Objetivos de estudio:

- Entender el ámbito y el efecto de la licencia GPL sobre la distribución.
- Entender lo que es una obra derivada.
- Distinguir entre aplicaciones que deben mantenerse libres y las que pueden comercializarse, usando software bajo GPL.
- Entender el rol de las licencias dobles.

Métodos de estudio:

- Estudio del caso Progress Software / MySQL.
- Investigación en Internet sobre el caso real y licencias similares.
- Preguntas y discusión.
- Presentación de una queja/defensa.

### 9.4.1. Introducción

Mucho se ha dicho del efecto “vírico” de la licencia GNU General Public Licence (GPL), y su naturaleza *copyleft* que la distingue de otras licencias de software libre como las licencias Berkeley Software Distribución (BSD) o Mozilla Public Licence (MPL). Brevemente, *copyleft* es el nom-

**Lectura complementaria**

Podéis ver también la nota de prensa de Free Software Foundation en:

<http://www.gnu.org/press/2002-03-01-pi-MySQL.html>.

Y la declaración de Mogden en:

<http://www.gnu.org/press/mysql-affidavit.html>.

bre que se ha dado al efecto de la cláusula de la licencia GPL, que obliga a mantener libre cualquier código que incorpore o se vincule de manera suficientemente directa con código distribuido bajo esta licencia.

El caso real de MySQL v. Progress Software es un caso típico de la aplicación de esta cláusula en el contexto de la distribución de una aplicación propietaria junto con código bajo licencia GPL e incorporado en un “paquete de aplicaciones” comercial de Progress Software (NuSphere MySQL Advantage). Veamos las circunstancias del caso para investigar los aspectos legales del asunto. La cita legal del caso es: Progress Software Corp. v. MySQL AB, 195 F. Supp. 2d 328, 329 (D. Mass.), 1st Cir., No. 02-1402 (*notice of appeal filed 3/29/02*).

#### 9.4.2. Telón de fondo

##### La licencia GPL

El objetivo de la licencia GPL es crear un “espacio público” de código informático compartido o “libre” que cualquiera puede usar, modificar o ampliar, pero del cual no se puede remover ningún módulo de código libre. La licencia GPL protege por derechos de autor el código en cuestión (“código GPL”) y permite a los usuarios realizar la mayoría de las acciones que están protegidas o reservadas por los derechos exclusivos de los titulares: la copia, la modificación y la distribución libre del código. Sin embargo, para mantener este espacio público de código abierto, la licencia impone ciertas obligaciones. Esta filosofía de apertura o libertad se expresa a través de dos características fundamentales de la licencia GPL:

- La obligación de mantener el carácter libre o abierto de cualquier modificación del código GPL que sea distribuido a terceros. Junto con la modificación compilada, se habrá de distribuir o poner a disposición el código fuente de la modificación.
- La restricción o prohibición en el uso de código GPL (o modificaciones del mismo) en programas y aplicaciones cuya licencia no sea la GPL. Esto se ha llamado el “efecto vírico” o, más positiva-

mente, la “herencia” de la licencia GPL: cualquier programa que usa, combina o integra de manera íntima cualquier código GPL se torna, él mismo, código GPL.

Estas obligaciones son los elementos de la licencia GPL que más preocupaciones dan a los usuarios y desarrolladores de software (no necesariamente libre), que no quieren que su propio código u otras aplicaciones propietarias se vean “infectadas” por el código GPL.

Es interesante notar que estas obligaciones surgen únicamente en el momento de la distribución posterior de código GPL o programas que lo incorporan. Los usuarios finales de código GPL y personas que incorporan o modifican este código para su propio uso en aplicaciones internas a las organizaciones, no están afectados o vinculados por estas obligaciones. Esto significa que la licencia GPL tiene su mayor efecto en un nivel de arquitectura informática que podríamos llamar “intermedio”: en paquetes de código (*drivers*, módulos, interfaces, rutinas y subrutinas, clases de Java, librerías, etc.) y aplicaciones (motores de bases de datos, compiladores e intérpretes, etc.) que se destinan a incorporarse en programas de uso final como los paquetes de ofimática (*office suites*), programas de bases de datos, aplicaciones de gestión de negocio o de procesos industriales, etc. Por lo tanto, son los desarrolladores de aplicaciones de uso final y distribuidores de aplicaciones los que se ven más afectados por el efecto “vírico” o “herencia” de la licencia GPL.

## Las bases de datos

Entre las aplicaciones de mayor uso en el mundo informático figuran las bases de datos, que almacenan y procesan datos de manera estructurada. Las tablas de una base de datos pueden contener las transacciones de un negocio (y cada uno de sus componentes: cliente, dirección, fecha, precio, etc.), las reservas de billetes de avión, las direcciones en una lista de correo o los registros de personal en una empresa. Para procesar lógicamente los datos en las tablas (por ejemplo, agregar, consultar, filtrar, eliminar o modificar datos), se usa un lenguaje estructurado de consulta de datos, a menudo llamado SQL en inglés (*Structured Query Language*). Éste es el “motor” de la base de datos y la parte fundamental de cualquier aplicación de base de datos. Ejemplos de bases de datos comerciales son Access

**Lectura complementaria**

Consultad la página web de MySQL AB en [www.mysql.com](http://www.mysql.com).

**Nota**

El estudio de *benchmarking* de las bases de datos es muy interesante, y se puede consultar en:

<http://www.eweek.com/article2/0,3959,293,00.asp> (del 25 de febrero de 2002).

y SQL Server de Microsoft, Oracle9i de Oracle, DB2 de IBM, etc. MySQL y PostgreSQL son motores de base de datos de software libre.

**9.4.3. MySQL**

MySQL es un motor de base de datos relacional de la empresa MySQL AB, de Suecia. Actualmente es el motor de base de datos libre más popular, con aproximadamente 4 millones de usuarios en el mundo. Fue creado en 1994 por Michael Widenius (ahora Director de Tecnología de MySQL AB) y David Axmark. La aplicación ha recibido varias contribuciones de otros programadores bajo el modelo de desarrollo de software libre, sin embargo M. Widenius controla el desarrollo de manera general y centralizada.

El programa MySQL tiene varias características especiales que lo hacen particularmente eficiente (por ejemplo, el caché para consultas, y la gestión de varias tablas y módulos de almacenamiento múltiples) y en pruebas de *benchmarking*, ha llegado en segundo puesto, después de Oracle, en términos de rendimiento. Un aspecto importante de MySQL es que puede elegir varios módulos de almacenamiento en función de criterios de velocidad, reversibilidad, estabilidad, tamaño, etc. aptos para el sistema informático en cuestión. Estos módulos están integrados o vinculados (compilados) en la aplicación final de MySQL distribuida e instalada en el ordenador del usuario (el programa “mysqld”).

**Las licencias MySQL**

Desde un punto de vista legal, los autores del programa (principalmente los autores iniciales, M. Widenius y D. Axmark) mantuvieron sus derechos de autor en la aplicación. Comparten la titularidad en la propiedad intelectual de los elementos de código contribuido por terceros. En 2001 se creó la empresa MySQL AB para gestionar los derechos en la aplicación entre los autores originales y dichos contribuyentes terceros y mejorar la comercialización del producto y los servicios relacionados.

Con respeto a las licencias de distribución, MySQL es una aplicación de software libre con cierta particularidad: la empresa la distribuye



bajo dos tipos de licencias, un sistema que se ha llamado licencias duales (*dual licensing* en inglés). En el caso de MySQL esto significa que:

- Por un lado, MySQL AB distribuye los elementos centrales del programa bajo licencia GPL para usuarios privados (finales) y desarrolladores de software libre (es decir, software que cumple con la *Open Source Definition* del OSI) para que puedan usar, copiar, modificar y distribuirlo libremente. Algunos otros módulos del programa se distribuyen bajo la LGPL.
- Por otro lado, la empresa distribuye la aplicación bajo licencia comercial para personas y empresas que quieren incorporar el motor de base de datos en otras aplicaciones comerciales o propietarias, o integrarlo en un servicio comercial.

Por el efecto de la licencia GPL (cláusula 2.b) las personas que reciben el programa bajo licencia GPL no pueden redistribuirlo bajo licencia doble, es decir, cobrando alguna forma de derechos de autor.

#### Nota

Los requisitos detallados en la definición o “estándar” de la Open Source Initiative para que el software sea considerado abierto (open source software), en:

[www.opensource.org/docs/definition.php](http://www.opensource.org/docs/definition.php)

#### Reflexión

##### La licencia doble MySQL

Es interesante notar que al principio (a partir de 1996), MySQL se distribuía con su propia licencia dual que permitía:

- La redistribución y el uso libre limitado, con una cláusula *copyleft* que prohibía la redistribución y creación de modificaciones y obras derivadas propietarias, para sistemas operativos de tipo UNIX (incluso Linux).

- Un uso según términos de shareware (código binario solamente, contribuciones voluntarias a los autores) y restricciones de uso y distribución para sistemas operativos Windows.

Esto significa básicamente que tenían una licencia doble para UNIX, y propietaria para Windows. Con el aumento de la popularidad de Linux, la empresa cambió la política de licencias en 2000, distribuyendo el programa bajo GPL para usuarios privados y desarrollos abiertos, y una licencia de tipo “propietaria” para fines comerciales. Abandonó la distinción entre entornos UNIX y Windows.

### Comercialización

Las bases de datos suelen ser productos de mayor uso comercial (los particulares tienen poco uso de una base de datos tan potente) y presentes en sistemas empotrados o integrados (*embedded*). Por lo tanto, actualmente MySQL AB obtiene la mayor parte de sus ingresos de la venta de licencias comerciales para la integración del motor de base de datos en desarrollos de sistemas informáticos propietarios, dentro de las organizaciones privadas o por consultores terceros que desarrollan software para clientes.

#### 9.4.4. Progress Software

Progress Software es una empresa americana de software que firmó en junio de 2000 un acuerdo preliminar con MySQL para la distribución comercial no-exclusiva de MySQL dentro de un paquete de software libre más amplio. Bajo el acuerdo, el equipo de un nuevo centro de negocio, NuSphere proveería soporte comercial y servicios de consultoría y de formación para clientes usuarios de MySQL. El contrato provisorio estipuló que la distribución de MySQL con los productos y servicios de Progress se haría bajo la licencia GPL.

#### Nota

Progress Software se encuentra en Internet en:  
<http://www.progress.com>.

NuSphere en: <http://www.nusphere.com>.

Podéis ver también la nota de prensa de MySQL en:  
<http://www.mysql.com/news/article-23.html>

y en: [www.nusphere.com](http://www.nusphere.com)

El acuerdo era preliminar, en vistas de un acuerdo final posterior, que nunca se firmó. Se comenta también en: <http://www.mysql.com/news/article-75.html>.

El paquete de distribución principal para usuarios finales de Progress era *NuSphere MySQL Advantage* que, como muchas distribuciones de software libre, consiste en una colección de aplicaciones informáticas que habían sido adaptadas para su interoperabilidad para varias plataformas. Entre los programas distribuidos figuraban el servidor de web Apache, algunos lenguajes libres como Perl y PHP, y el motor de base de datos MySQL. Con este paquete de Progress Software, un comprador (usuario final) podía aprovechar la colección de software libre y un módulo de interfaz y otras aplicaciones para la instalación y ejecución de los programas de manera fácil e interoperable o compatible.

En el paquete, Progress incorpora un módulo de su propio software que se llama Gemini. Este es un módulo de almacenamiento para el motor de base de datos MySQL que agrega algunas funciones útiles que no existen en la aplicación principal. Por ejemplo, la gestión de errores y fallas de máquina (*crashes*) y la recuperación de datos corruptos y la reversibilidad de las operaciones (hacer marcha atrás en una operación y recuperar datos o estados anteriores). Como permite entrar nuevos datos en una base nueva o preexistente como subfunción de las tareas ejecutadas por el motor principal, Gemini se integra a la aplicación MySQL al compilarse. Cuando un usuario instala el paquete *NuSphere MySQL Advantage* en su equipo, se crea un solo ejecutable (*mysqld*) que incorpora el módulo Gemini.

En la versión 2.2 de *NuSphere MySQL Advantage*, el código fuente de Gemini no estaba disponible, y el CD-ROM incluía un texto indicando que el código fuente se proveería en el futuro.

#### 9.4.5. El litigio

En junio de 2001, MySQL AB se enteró de que la filial NuSphere de Progress Software había adquirido el dominio de Internet [www.mysql.org](http://www.mysql.org). MySQL consideró que es un uso no autorizado de su marca registrada (lo que resulta en un acto de *cybersquatting*) y pidió a Progress la transferencia del dominio. Progress se negó. Asimismo, MySQL AB fue informado de que la aplicación MySQL se distribuía con un módulo de software propietario sin su código fuente (el módulo Gemini). Para MySQL, NuSphere y Progress estaban en violación del contrato preliminar entre las partes y de la licencia GPL referente al código de la base de datos: MySQL considera que la inclusión de Gemini como software propietario, sin proveer su código fuente, infringía la cláusula 2.b de la licencia GPL sobre obras derivadas. Por lo tanto, MySQL escribió a Progress para que se rectificara el incumplimiento. Progress se negó, alegando que el programa MySQL se distribuía con su código fuente y que, por lo tanto, no había ningún incumplimiento. La empresa sueca revocó el derecho del uso de la marca MySQL.

Como consecuencia de esta revocación, Progress Software inició el litigio el 15 junio de 2001, alegando contra MySQL el incumplimiento del contrato de distribución, la interferencia ilícita en sus contratos y negocios con terceros, y competencia desleal, entre otras responsabilidades. Asimismo, Progress pedía una declaración sobre sus derechos de uso de la marca registrada y otros derechos para la distribución y venta de software MySQL. Los directores alegan que “si se le niega la distribución de los productos de MySQL, la empresa NuSphere sufriría muchos daños en un negocio dónde había invertido millones de dólares.”

En defensa y contrademanda del 11 de julio de 2002, MySQL AB alegó que Progress estaba en violación de la marca registrada (MySQL), incumplía el contrato provisorio de distribución entre las partes del conflicto y violaba la licencia GPL relativo al software MySQL. Asimismo denunciaba las prácticas comerciales de Progress como desleales y engañosas. La base de la contrademanda de MySQL es que el módulo Gemini era una obra derivada de MySQL porque se vinculaba estáticamente con el mismo. Bajo la licencia GPL, hemos visto que cualquier programa que esté vinculado con código GPL se con-

siderará como una obra derivada del código original y deberá distribuirse con la misma licencia GPL y con su código fuente.

En febrero de 2002, la jueza del caso, en una audiencia en sumario, resolvió a favor de MySQL sobre el tema del uso de la marca y prohibió a Progress que distribuyera el software MySQL y usara la marca hasta que se resolviera el litigio principal. Sin embargo, el juez se negó a resolver en juicio sumario el tema de la violación del contrato y de la licencia GPL (la cuestión de la creación de una obra derivada y el efecto *copyleft* de la licencia), declarando que Progress tenía razones suficientes para exigir una audiencia completa sobre el tema.

#### 9.4.6. Conclusiones

Durante el curso del litigio, se liberó la versión 2.3.1 de NuSphere MySQL Advantage con el código fuente del módulo Gemini en “abierto”, lo que técnicamente permite a Progress cumplir con la licencia GPL. Además, el 7 de noviembre de 2002, las empresas llegaron a un acuerdo sobre el litigio y resuelven sus diferencias sobre el uso de la marca y del código del motor de base de datos. Por lo tanto, la licencia GPL no ha sido discutida por los tribunales de manera completa.

Sin embargo, el caso es interesante, no solamente porque es la primera vez que la licencia GPL se considera en los tribunales – aunque el juez se negó a opinar definitivamente sobre la licencia hasta la audiencia completa del litigio (que nunca tuvo lugar)– sino también porque varias personas clave del movimiento libre, como Eben Moglen, contribuyeron con su opinión a las discusiones, dando precisiones sobre la cláusula 2.b y el efecto *copyleft* de la licencia GPL.

#### 9.4.7. Preguntas y actividades

El caso ilustra algunos temas de gran interés para los “intermediarios” del software libre, como los distribuidores y los consultores informáticos que se aprovechan del software libre para interactuar o integrarlo en sus propios programas y distribuciones.

##### Lectura complementaria

Para más detalles, podéis consultar la nota de prensa de NuSphere en:  
<http://www.nusphere.com/releases/2002/110702.htm>

##### Lectura complementaria

Para más detalles, se puede consultar la declaración jurídica completa de Eben Moglen sobre el asunto (en inglés), en:  
<http://www.fsf.org/press/mysql-affidavit.html>

### Análisis del caso

- 1) ¿Quién es titular de los derechos en los módulos de software en cuestión y cuáles son los usos que se hacen de ello? ¿De qué manera se aplica la licencia GPL a la distribución de estos módulos y programas de software por Progress?
- 2) ¿Tiene Progress una defensa legal válida contra MySQL y la aplicación de la licencia GPL al módulo Gemini? ¿Cómo hubiera podido Progress mantener su código Gemini “propietario” (indicad unas estrategias tecnológicas, legales o comerciales si las hay)?
- 3) ¿La definición de “obra derivada bajo la licencia” corresponde a la definición de obra derivada en el marco de derecho de la propiedad intelectual de su país? ¿Bajo el derecho de su país, esta cláusula 2.b de la GPL sería vinculante en las condiciones del caso sobre Progress? ¿Y sobre sus clientes?

### Extensión

- 4) El caso depende, en gran medida, de la cuestión del vínculo y la interacción entre dos programas o módulos de programa. Hay diferentes formas de vincular aplicaciones (en el momento del desarrollo, la compilación, la interpretación, la distribución, la instalación o la ejecución, etc.). En un entorno informático que vosotros elijáis, haced una tabla con estas formas de vínculos (por ejemplo, vínculos estáticos y dinámicos y llamadas API, o a librerías, distribución sobre un mismo medio, etc.) que ayude a determinar cómo una aplicación suya sería afectada (desde el punto de vista legal) por cada tipo de vínculo con un programa bajo la licencia GPL y la LGPL.
- 5) Test de “herencia”: como resultado de la tabla de la pregunta anterior, definid un test para determinar si un desarrollo determinado obliga el uso de la licencia GPL relativa a la aplicación resultante o si se puede establecer la naturaleza propietario de código que interactúa con código GPL.
- 6) Linux es el sistema operativo más popular de software libre y se distribuye bajo licencia GPL. ¿Esto quiere decir que cualquier programa que se ejecute o interactúe con Linux debe distribuirse como software libre? ¿Y si el sistema operativo Linux y aplicacio-

nes que se ejecutan en ello se integran en un producto o dispositivo (*embedded* o empotrado)?

### Licencias dobles

- 7) MySQL ha construido su negocio en torno al software libre, con una estrategia de licencias dobles. Buscad y comentad la política de licencia de MySQL. ¿Cuándo se necesita una licencia comercial? (consultad <http://www.mysql.com/products/licensing.html>) ¿Cuáles son las ventajas e inconvenientes (legales) del *dual licensing*?
- 8) Identificad las cláusulas principales que incluiría la licencia libre y la licencia comercial. Explicad su existencia y funcionamiento relativo al contexto y modelo de negocio.
- 9) Investigad en Internet otros paquetes de software libre distribuidos bajo una licencia doble. Comentad las similitudes y diferencias: el tipo de software, el tipo de mercado, si es posible, el modelo de desarrollo y de coordinación del desarrollo, el tipo de licencia.
- 10) ¿Qué consecuencias tiene una licencia doble para el modelo de desarrollo y negocio de la empresa?

### Defensa de derechos

- 11) MySQL ha tenido que recurrir a los tribunales para defenderse contra la explotación de su software por un tercero, fuera de los límites permitidos por un contrato comercial y la misma licencia GPL. ¿Qué pasos puede tomar un programador de software libre para proteger su software contra la explotación por terceros?

## 9.5. Recursos humanos y comunidades de desarrollo

Objetivos de estudio:

- Aprender los riesgos legales “internos” vinculados con el desarrollo y la comercialización de software libre.

- Preparar la organización de recursos humanos de las empresas involucradas en el software libre.
- Entender las estrategias para mantener la “libertad” de las obras frutos del desarrollo abierto.

Métodos de estudio:

- Lectura de los casos QDRSoft y LibreSolutions (secciones 1 y 3).
- Discusión sobre los riesgos legales.
- Preparación de políticas de conducta, contratos de empleados y de colaboradores.
- Estudio de cláusulas para incorporar en contratos comerciales para el desarrollo y distribución de software libre, y la prestación de servicios relacionados.
- Investigación de comunidades de desarrollo y sus políticas y reglas (legales y para-legales).

#### 9.5.1. Introducción

Ya son muchos los estudios sobre los modelos de desarrollo y distribución de software libre, como nuevo modelo tecnológico y de negocio frente a los modelos tradicionales. Estos trabajos enfocan varios aspectos del negocio de software libre, como pueden ser el modelo del desarrollo mismo del software, la gestión de la coordinación y control de las aplicaciones, las fuentes de ingresos y beneficios para empresas que no pueden “vender” el software en cuestión y las diferencias con modelos comerciales o propietarios.

Al nivel microeconómico, el negocio del software libre, ya sea que se trate del desarrollo o de la distribución, tiene implicaciones sobre varias políticas internas de las empresas. Una de estas áreas más afectadas es el área de los recursos humanos: los contratos y políticas de conducta de los empleados, la relación con los colaboradores exter-



nos, los proveedores y los clientes, los modelos de beneficio e incentivos, etc.

Este estudio de caso trata de elucidar cuales son las dificultades para la organización de la programación y comercialización bajo un modelo de desarrollo libre y la gestión de los derechos correspondientes. Aprovechad los datos de dos casos anteriores, QDR-Soft y LibreSolutions, cuya lectura es necesaria.



Hay un gran número de estudios académicos en:

[http://opensource.mit.edu/online\\_papers.php](http://opensource.mit.edu/online_papers.php).

Además, se destacan las discusiones de E. Raymond en “La catedral y el bazar” en castellano en:

<http://www.sindominio.net/biblioweb/telematica/catedral.html>

Y los escritos de varios autores en “Open Source: Voices from the open source revolution” en inglés en:

<http://www.oreilly.com/catalog/opensources/book/toc.html>

(en proceso de traducción al castellano en:  
<http://es.tldp.org/htmls/proy-opensources.html>).

### 9.5.2. QDR-Soft

La empresa de software QDR-Soft tiene dos productos propietarios relacionados con Internet, para la creación de páginas web, intranets, extranets y la gestión de contenidos (QDnet y QDweb). La primera de estas aplicaciones había sido desarrollada para clientes comerciales de la empresa, durante varios años y con varias adaptaciones y ampliaciones; la segunda es una versión simplificada para “gran público”. Por diversas razones, como la mayor competencia en el sector, el aprovechamiento de conocimientos de desarrolladores

externos y la extensión, la integración y la estandarización de las aplicaciones, QDR-Soft está considerando abrir su código fuente para desarrollar, lanzar y distribuir versiones de software libre: OpenQDnet y OpenQDweb.

Hasta ahora, el código de la aplicación ha sido diseñado, programado y documentado internamente por empleados de la empresa, aunque también han contado con la colaboración de varios programadores externos bajo contratos de servicio. El modelo de contratación con empleados y colaboradores figura a continuación.

### Elementos relevantes del modelo de contrato actual de empleo o con colaboradores (autónomos)

#### “Propiedad intelectual”

“El empleado [colaborador] reconoce que ha sido contratado por EMPRESA para participar en la producción de productos y servicios que redundarán siempre en beneficio de EMPRESA, bajo la dirección, supervisión y control de esta última. En este sentido, dado que el resultado del trabajo en la empresa está destinado a su explotación, el empleado [colaborador] reconoce, asimismo, la titularidad exclusiva de EMPRESA respecto a cualesquiera patentes, derechos de autor, marcas, diseños y secretos industriales, o cualesquiera otros derechos de la misma naturaleza que sean resultado de dicho trabajo.”

“Sin perjuicio de lo anterior, aquellas obras que no pudieran ser consideradas colectivas, se entenderán totalmente cedidas en exclusiva a la empresa, y por tanto el empleado [colaborador] cede de forma exclusiva los derechos de uso, reproducción, distribución, comunicación pública y transformación de las obras derivadas de su específica labor de investigación y programación para la empresa, incluyendo los datos, listados, diagramas y esquemas elaborados en la fase de análisis, los manuales de aplicación, los restantes datos y materiales de apoyo; con el fin de que la empresa pueda realizar copias de las mismas, instalarlas en cuantos ordenadores estime oportuno y utilizarlas en su actividad empresarial, así como modificar el código fuente de los programas desarrollados con la finalidad de adaptarlos a sus características o necesidades específicas. Dicha cesión alcanza a todos los países del mundo y se extiende por todo el periodo

de duración de los derechos. Respecto a dichas obras exclusivas, el empleado [colaborador] no podrá ceder el uso de las mismas a terceros ni transmitir ninguno de los derechos que tenga sobre ellas en virtud de este contrato, comprometiéndose a no divulgarlas, publicirlas, ni ponerlas, en todo o en parte, de ninguna otra manera a disposición de otras personas o empresas. Además, el empleado [colaborador] entregará a la empresa el código fuente y el código objeto de dichos programas exclusivos como máximo a los siete (7) días a contar desde el requerimiento de la empresa, con el fin de que pueda ejercer los derechos cedidos.”

“Para ello, el empleado [colaborador] asistirá a EMPRESA en la medida necesaria para la obtención y refuerzo de patentes, derechos de autor, marcas, diseños industriales y otros derechos aplicables a dichas invenciones en cualesquiera países, incluso una vez finalizado el contrato de trabajo. Asimismo, en la medida en que EMPRESA no consiga la firma del empleado para conseguir el registro o defender judicialmente cualesquiera de los anteriores derechos de autor, patentes, marcas, diseños y secretos industriales, u otros derechos afines; bien por la incapacidad del empleado [colaborador], o bien por cualquier otra causa, el empleado [colaborador] apodera ya desde ahora a EMPRESA para que en su nombre pueda hacer todo lo necesario permitido en derecho para su obtención o defensa.”

### “Confidencialidad”

“El empleado [colaborador] reconoce que EMPRESA posee determinada información confidencial incluyendo, entre otras, información acerca de las circunstancias, sistemas de gestión, procesos industriales, propiedades, secretos comerciales e industriales, planes de empresa, estrategias de negociación, programas de ordenador y otros elementos propios de la explotación mercantil y actividad profesional; tanto de la propia empresa como de otras. Asimismo, el empleado [colaborador] reconoce que toda esta información confidencial le será revelada con la finalidad de poder llevar a cabo sus obligaciones contractuales, según los contratos que firme o haya firmado con EMPRESA. En este sentido, el empleado [colaborador] se compromete a tratar confidencialmente y a no reproducir, difundir o publicar, sin autorización escrita de EMPRESA, toda esta información confidencial, respecto de terceros, así como tomar las precauciones debidas y hacer

todo lo necesario para que los terceros ajenos a la empresa, o no autorizados, no tengan acceso a la mencionada información. En virtud de lo anterior, el empleado [colaborador] no hará ni permitirá copias de cualesquiera documentos, datos o informes de EMPRESA, salvo las necesarias para uso interno de la empresa.”

“Tanto durante la vigencia de los contratos de empleo [colaboración], como extinguido el mismo, el empleado [colaborador] entregará a EMPRESA (o borrará o destruirá, a elección de la misma), en el plazo máximo de siete (7) días desde cualquier requerimiento de la misma o de alguno de sus clientes, toda información confidencial o circunstancias de EMPRESA o de sus clientes que, por exigirlo así el cumplimiento de sus trabajos, hubiere debido registrar en cualquier clase de soporte físico.”

#### **“No concurrencia”**

“Durante el tiempo de la permanencia del empleado en EMPRESA [el plazo del contrato de colaboración], éste no desarrollará la misma actividad por cuenta propia, ajena, o a través de terceros, ya sea directa o indirectamente.”

“Asimismo, el empleado [colaborador] se compromete a que, una vez extinguido el contrato de empleo [colaboración] que haya firmado con EMPRESA, éste no se establecerá por cuenta propia, ajena, o a través de terceros, ya sea directa o indirectamente, prestando los mismos servicios que presta EMPRESA y que tengan relación directa con los mismos proyectos de EMPRESA en que el empleado [colaborador] ha participado o ha tenido conocimiento en EMPRESA, a ninguna empresa cliente de EMPRESA u socio de negocio, durante un periodo de dos (2) años.”

Una vez la distribución del código actual de OpenQDnet esté “abierta”, QDRSoft continuará un desarrollo interno, sobre todo de los componentes nuevos ya identificados y la optimización de módulos existentes. Asimismo, en base al modelo de desarrollo libre, la empresa integrará cualquier contribución, corrección o modificación propuesta por programadores ajenos que considere apropiado. Sin embargo, la empresa quiere mantener el control y la coordinación del proyecto, erigiéndose en *líder* o *coordinador* de la aplicación y de

su desarrollo libre. Asimismo, quiere evitar que la aplicación haga un “fork” (derivación del núcleo principal del código y toma de control de la derivación por un tercero).

### 9.5.3. LibreSolutions

LibreSolutions es una empresa nueva que se propone distribuir varios paquetes de software libre y prestar servicios de consultoría informática relacionados con los mismos. Los dos fundadores principales de la empresa van a crear los dos primeros paquetes de distribución para clientes comerciales y desarrolladores respectivamente. Luego, la empresa contempla reclutar algunos responsables de cliente para comercializar los paquetes, y sobre todo varios desarrolladores de distintos niveles para realizar la prestación de los servicios relacionados con los paquetes contratados por clientes: modificaciones, personalizaciones, adaptaciones y desarrollos a medida del software, cursos de formación y servicios de mantenimiento.

Sin embargo, por razones económicas, financieras y filosóficas (los fundadores son fieles a la filosofía del software abierto y libre), también quieren crear una red de desarrolladores “habilitados” o cualificados con alto nivel de competencias. Por un lado, esto les permite ahorrar costes sociales y de personal. Por otro lado, aprovecharán el conocimiento y experiencia de otros consultores y desarrolladores informáticos y fomentarán una comunidad de alta calidad alrededor de su “distribución”. Además, proveerá a la empresa de una gran flexibilidad para responder a las exigencias de los clientes y del desarrollo de nuevas aplicaciones. La empresa creará un sitio web y foro especializado en la distribución de los paquetes de LibreSolutions. Incluirá no solamente el código fuente y los detalles técnicos de las aplicaciones, sino también materiales de formación y las listas de discusión y de soporte. Sobre todo, se publicarán los procesos para que desarrolladores puedan integrarse a la comunidad LibreSolutions, lo que les permitirá prestar servicios para la empresa.

#### Nota

Para un concepto similar, podéis consultar el DevNet de Expropia descrito en:

[http://www.extropia.com/open\\_source\\_case\\_study.html](http://www.extropia.com/open_source_case_study.html).

#### 9.5.4. Conclusiones

Las dos empresas consideradas contemplan dos actividades técnicas y comerciales importantes relacionadas con el software libre: el desarrollo abierto y la distribución libre. Basándose en nuevos modelos de negocio y de desarrollo, QDR-Soft y LibreSolutions consideran varios procesos organizativos para flexibilizar, optimizar y mejorar la calidad de los productos y servicios ofrecidos. Estos procesos, sobre todo en el ámbito de los recursos humanos y las relaciones con contratantes, requieren mucho cuidado en el momento de la implementación.

#### 9.5.5. Preguntas y actividades

##### En general

- 1) ¿Cuáles son los riesgos legales a los cuales se enfrentan las empresas QDRSoft y LibreSolutions en el área de la gestión de los recursos humanos (en relación con el software libre)? (incorporación de software libre en desarrollos privados, incorporación de código propietario en software libre, pérdida de confidencialidad y secretos de negocio).
- 2) ¿Cuáles son los elementos principales de un contrato de empleo de un desarrollador o jefe de proyecto? Haced una tabla con (1) las cláusulas de mayor interés y un resumen del enunciado de las mismas, (2) una explicación de su objetivo y funcionamiento y (3) el entorno económico comercial que justifique el enunciado típico del pacto.
- 3) Ambas QDRSoft y LibreSolutions quieren establecer redes o comunidades de desarrolladores para el desarrollo de sus aplicaciones y la prestación de servicios contratados por clientes respectivamente. Investigad en Internet y comentad distintas comunidades de desarrollo y de servicios de software libre (dos casos diferentes con aspectos superpuestos). ¿Qué estrategias se usan para la protección de la propiedad en el software?

## QDR-Soft

- 4) ¿Cuáles son los cambios que se habrían de incluir en los contratos de QDRSoft para hacer frente a su cambio de modelo de negocio hacia el desarrollo de software libre y servicios relacionados?
- 5) Redactad (de manera general) los pactos de confidencialidad, de propiedad intelectual y otros dos pactos a elección relevantes a la situación, justificando sus versiones.
- 6) Buscad en Internet y comentad una política declarada relativa al desarrollo de aplicaciones libres (el estándar más conocido es del proyecto GNU, en: [http://www.gnu.org/prep/standards\\_toc.html](http://www.gnu.org/prep/standards_toc.html)).
- 7) Definid los elementos principales (legales) de una política de desarrollo para la aplicación OpenQDnet.
- 8) Delinead los elementos de una estrategia para QDRSoft y la eventual “Comunidad OpenQDnet” para mantener la independencia y “libertad” de su trabajo.

## LibreSolutions

- 9) ¿Cuáles son las cláusulas clave para los contratos de empleo para LibreSolutions? ¿Por qué?
- 10) Si LibreSolutions quiere contratar trabajadores autónomos para realizar algunos desarrollos a medida, o prestar servicios a clientes, ¿qué pactos son importantes para el contrato entre LibreSolutions y el desarrollador? Describid en términos generales estos pactos.
- 11) Para la red de desarrollo y servicios de LibreSolutions, ¿qué estrategia general aconsejáis? ¿Explicad cuáles son los temas de derecho de propiedad intelectual u otro ámbito de derecho relevantes a esta red y a este modelo de prestación de servicios? (Por ejemplo, derechos, de autor, patentes, marca, confidencialidad, etc.)
- 12) ¿Qué documentación recomendáis para esta red de desarrollo? ¿Cuáles son los elementos legales principales?





## 10. Aspectos relevantes para la implantación de software (estudio de caso)

En esta unidad final, vamos a seguir nuestro aprendizaje práctico a través de nuevos estudios de caso. Os referimos a la introducción de la unidad 9 para los objetivos generales de los estudios de caso y la metodología a seguir.

Por un lado, hay muchos temas legales importantes relacionados con el software libre que son relevantes para sus usuarios: el ámbito de los derechos de uso o de modificación, por ejemplo, o las garantías y responsabilidades de los proveedores (o su ausencia). Por otro lado, debemos ilustrar y comentar la relación entre software libre y otras áreas de derecho, como los estándares, la protección de los datos personales y el control del cifrado.

Finalmente, hemos creído interesante incluir un resumen del conflicto legal SCO-Linux, dadas su actualidad e importancia.

Por lo tanto, en esta unidad 10, estudiaremos lo siguiente:

- Presentaremos dos casos de implementación y utilización de software libre, tanto en el ámbito comercial como en el privado (**Hospital Beaumont, Lorena Nadal**). Describiremos el uso de varias aplicaciones libres en un hospital, donde se mezclan plataformas libres y propietarios, y un caso de un consumidor que tiene problemas con una aplicación libre.
- Presentaremos un caso de conflicto entre los derechos de autor, el secreto comercial, el software libre y los estándares: el caso **Kerberos**.
- Realizaremos varias actividades relacionadas con dos temas estudiados en la unidad 8: la protección de **datos personales** y el control de la exportación de **productos de cifrado**.

- Finalmente, presentaremos y analizaremos un caso real, que actualmente conmociona el mundo del software libre, el litigio entre **SCO/Caldera e IBM** y la comunidad del software libre en general (representado o “personificado” por Red Hat). Este conflicto suscita una multitud de cuestiones interesantes para abordar en este temario: cómo se aportan líneas de código en los proyectos libres, cómo se usan las diferentes licencias de distribución de software, cómo se aplica la licencia GPL, para qué se usan las patentes a los fines de la defensa legal (una práctica que viene directamente del mundo del software propietario), y cómo redactar e interpretar contratos, licencias y obligaciones de confidencialidad. Asimismo, nos permite reflexionar más ampliamente sobre los riesgos legales presentados por los modelos de desarrollo y distribución del software libre.

A partir de la lectura de los casos y la realización de las actividades asociadas, el lector será capaz de:

1. Valorar los problemas legales vinculados a la implementación de software libre en las organizaciones.
2. Identificar las ventajas e inconvenientes legales de la migración de un sistema informático propietario a uno “libre”.
3. Conocer la aplicación del derecho del consumidor en el contexto del software libre.
4. Obtener un mayor entendimiento de áreas del derecho asociadas con el software libre: estándares, datos personales y criptografía.
5. Formular vuestra propia opinión sobre el marco legal y su aplicación en este sector, basándoos en experiencias prácticas, y desarrollar argumentos a favor y en contra de distintos usos del software libre.
6. Crear una documentación práctica para la implementación de software libre en vuestra organización.
7. Comprobar el aprendizaje de las unidades anteriores del curso.

## 10.1. El Hospital Beaumont

Objetivo de estudio:

- Considerar los aspectos legales relevantes de la implementación de software libre en una organización.

Métodos de estudio:

- Lectura del caso.
- Investigación en Internet de las aplicaciones referidas.
- Análisis de las licencias en cuestión.

### 10.1.1. Introducción

La introducción de software libre en la organización está llena de desafíos técnicos de compatibilidad, interfaces, GUI, de formación de personal, etc. Y, como veremos a continuación, aparecen algunos problemas legales. Este estudio de caso describe la implementación de varias aplicaciones de software libre en un hospital del sector público, el Hospital Beaumont en Irlanda. El caso describe las aplicaciones instaladas, las dificultades encontradas y aprovecharemos para discutir las estrategias y los pasos requeridos para resolver las dificultades y lograr una implementación exitosa del software libre.

#### Nota

Este estudio de caso se basa en otro estudio realizado por Brian Fitzgerald y Tony Kenny: "Open Source Software can improve the Health of the Bank Balance - The Beaumont Hospital Experience". Accesible en línea en:

<http://opensource.mit.edu/papers/fitzgeraldkenny.pdf>

### 10.1.2. El Hospital Beaumont y sus sistemas informáticos

El Hospital Beaumont de Dublín, Irlanda, abrió en 1987 como resultado de la fusión de tres antiguos hospitales. Funciona como centro

de formación para el Royal College of Surgeons de Irlanda (RCSI) y para la Universidad de la Ciudad de Dublín (UCD). Financiado por fondos públicos, el Hospital se enfrenta a un déficit presupuestario de 17 millones de euros, para 2003 y tiene un personal de 3.000 personas.

Antes de los cambios actuales, el entorno informático del hospital se caracterizaba por una alta heterogeneidad en cuanto a equipos y plataformas informáticas. Respecto del equipamiento informático, había 36 servidores Intel, que trabajaban en un entorno Microsoft Windows. Además, la aplicación clínica principal se basaba en un *mainframe* HP 3000 y los sistemas financieros se ejecutaban sobre un HP UNIX. El sistema informático del hospital también tenía aproximadamente 1.000 ordenadores de sobremesa, varios de los cuales eran casi obsoletos (RAM inferior a 64 MB y CPU de menos de 300 MHz) a causa de presiones presupuestarias. En cuanto a las aplicaciones, la política informática del hospital era mixta, tanto frente a los distintos tipos de aplicaciones como a los proveedores. El hospital había adquirido soluciones informáticas adecuadas en la medida que se disponía de ellas en el mercado y las había modificado (en la medida posible) y había creado nuevas soluciones cuando no se encontraba ninguna adaptada a las condiciones de trabajo del hospital. Como resultado, el hospital tenía un abanico de aplicaciones centralizadas y periféricas. Por ejemplo, a causa de las presiones presupuestarias y en la medida que hubo fondos disponibles, el hospital fue adquiriendo progresivamente paquetes de software tales como MSWord 2 y 6, WordPerfect, QuattroPro, AmiPro, etc. Esto contribuyó a la heterogeneidad de los sistemas informáticos del hospital.

### 10.1.3. Un cambio de enfoque

En 2001, el Departamento Informático del hospital empieza a estudiar el software libre como alternativa al software propietario para varios de los sistemas del hospital. Hasta entonces, el software libre se había considerado apto para las aplicaciones de la infraestructura invisible de las organizaciones, ejecutándose en los servidores del *back-office* (el sistema operativo GNU/Linux, el servidor web Apache, etc.). El Departamento Informático realizó estos estudios a través de listas de discusiones y foros centralizadores de software libre como el Open Source Initiative, Sourceforge y Slashdot, con el fin de exami-

nar cuáles eran las aplicaciones disponibles, cuáles su vigencia y grado de actividad de desarrollo y soporte y cuáles podrían ser los problemas en la implementación.

El director y los responsables del Departamento Informático juzgaron que el software libre empezaba a desarrollarse e implementarse no solamente para el *back-office* sino también en aplicaciones de mayor visibilidad para los usuarios. Los programas de usuario más evidentes eran nuevos paquetes de ofimática (*office suites*) con procesadores de texto y planillas de cálculo, y programas de correo electrónico: OpenOffice.org, SuseMail, MySQL, etc. Por lo tanto, consideraron que el software libre evolucionaba desde sistemas horizontales de infraestructura hacia aplicaciones más “verticales”, orientadas a las unidades de negocio operativas de las organizaciones.

Como consecuencia de estas investigaciones, el Departamento Informático decidió migrar progresivamente (estudiando caso por caso) varios de los sistemas del hospital a plataformas y aplicaciones de software libre. Tres principios fundamentales impulsaron este cambio:

- la eficiencia presupuestaria (rentabilizar el dinero del contribuyente);
- el pragmatismo (frente a las insuficiencias presupuestarias para los sistemas informáticos del hospital), y
- la usabilidad (buscar aplicaciones con funcionalidades y apariencia similares a las aplicaciones comerciales ya en uso, a fin de minimizar los problemas del usuario frente al cambio).

Sin embargo, hay que notar que el hospital ha seguido usando e instalando nuevas aplicaciones comerciales, por lo tanto el cambio al software libre no ha sido una filosofía o doctrina restrictiva.

#### 10.1.4. Las aplicaciones nuevas

Este cambio al software libre se desarrolló en varias etapas:

- 1) **Implementación de un paquete de ofimática.** El hospital empezó instalando StarOffice 5.2 para equipos de sobremesa. A causa de

varias dificultades prácticas y funcionales en esta versión de la aplicación, esta implementación creó muchos problemas, tanto para los usuarios como para el personal técnico del hospital. Luego, se decidió instalar StarOffice 6.0, contando con el soporte de Sun. El Departamento Informático quería un sistema distribuido con clientes *thin*, centralizando el paquete en un solo servidor Linux con acceso por red. Este sistema centralizado provocó problemas técnicos, por ejemplo, la sobrecarga del servidor único y de la red interna del hospital. Por lo tanto, se decidió instalar la aplicación en local para aquellos que lo querían, lo que a lo largo mejoró sustancialmente la satisfacción de los usuarios. Es interesante notar que algunos usuarios mantuvieron paquetes de ofimática comerciales, por razones de fidelidad y de costumbre (un 8%, que corresponde a unos 80 usuarios). Sin embargo, el Departamento Informático no se hizo responsable del mantenimiento y la actualización de dichas aplicaciones. Uno de los aspectos más innovadores de StarOffice es el uso de sus capacidades de XML para la estructuración y procesamiento lógico e interacción dinámica de los distintos archivos (por ejemplo, formularios de pedidos que se envían automáticamente al departamento responsable).

- 2) **Sistema de gestión de contenidos:** para la gestión de todos los documentos y otros “contenidos” del hospital (políticas de personal, protocolos de laboratorio, formularios, documentos de trabajo, historias clínicas de pacientes, mensajes, foros de discusión, etc.), el hospital instaló Zope, que se descarga gratuitamente de Internet. Después de efectuar varios estudios e intentos de configuración internos, el hospital contrató a una empresa local de servicio de software libre para la instalación, la adaptación y el soporte. El programa Zope permite gestionar documentos y otros archivos a través de metatags que identifican y describen un documento, además de permitir la definición de las reglas de acceso y de uso. El sistema de gestión de contenidos (SGC) se integra con el servidor de directorio LDAP del hospital (que contiene todos los datos del personal), para la gestión de los accesos y privilegios al servidor SGC basada en distintas categorías y grupos de personal. Hoy, un equipo propio del hospital define, soporta y mantiene la aplicación.

**Nota**

LDAP es el acrónimo de *Lightweight Directory Access Protocol* (protocolo de acceso a directorio ligero). Es

una base de datos con todas las direcciones de correo y diversos datos de las personas registradas (por ejemplo, usuarios del hospital), que se usa para completar las direcciones de correo en aplicaciones de e-mail, o buscar datos sobre dichas personas.

- 3) **Gestión de radiografías digitales:** actualmente, la mayoría de las cámaras radiográficas modernas permiten crear imágenes digitales, que pueden ser almacenadas y vistas a través de equipos informáticos. El hospital está remplazando progresivamente sus máquinas análogas con equipos digitales. Basándose en estándares internacionales para el procesamiento de imágenes (DICOM), el hospital ha desarrollado con su propio personal una aplicación de procesamiento para el almacenamiento e inspección de radiografías en formato digital y en línea. También contó con la ayuda de Sun, que proveyó un servidor *Sun Fire V880* con 1 TB de memoria. Este desarrollo interno se basa en *scripts* de Perl para recuperar archivos desde el sistema de datos radiográficos en el HP3000. Es interesante destacar que este sistema permite una integración total de los archivos de imagen con otros datos del paciente sobre una misma plataforma, mientras que otros sistemas comerciales deben ejecutarse separadamente y hasta en dos equipos separados.

**Nota**

DICOM: *Digital Imaging and Communications in Medicine* es un método estándar para la transmisión de imágenes médicas y sus datos asociados. Para más detalles, ver <http://medical.nema.org/>. Además, el estándar DICOM facilita el desarrollo informático para ampliar los archivos de imágenes y los sistemas de comunicaciones PACS (*Picture Archiving and Communication Systems*) y su interfaz con sistemas de gestión de la información médica.

- 4) **Servidor de aplicaciones:** el hospital decidió emplear JAVA/J2EE como arquitectura principal de referencia para el desarrollo de software interno, la cual es compatible con la mayoría de las nuevas aplicaciones instaladas. Después de considerar alternativas comerciales como Oracle o WebShere de IBM, se eligió el servi-

dor de aplicaciones de software libre JBOSS. Sin embargo, el hospital no realiza todo el trabajo informático de manera interna, sino que se ha negociado un contrato de servicio con una consultora informática local para varios trabajos de consultoría, instalación y mantenimiento.

- 5) **Correo electrónico:** como muchas organizaciones actualmente, el hospital basa sus comunicaciones internas en el correo electrónico. Anteriormente, tenía una licencia de Lotus Domino para 800 usuarios. Los 3.000 empleados querían acceso al correo electrónico, lo que requería ampliar las licencias de manera prohibitiva. Después de un estudio del software existente, se cambió a SuSe Mail, que provee todas las funciones básicas del correo electrónico requeridas por el personal.
- 6) **E-learning:** finalmente, el hospital ha tenido que considerar la formación continua de su personal médico y no médico. Varios cursos presenciales han resultado deficientes por falta de asistencia de los alumnos, por una preparación inadecuada y a causa de las numerosas cancelaciones de último momento. Junto con otras instituciones académicas (no hay que olvidar que el hospital sirve de centro de formación práctica para la Universidad de la Ciudad de Dublín), el hospital investigó las posibilidades de la formación virtual individualizada y grupal. El precio de varias soluciones comerciales estaba fuera del alcance del presupuesto hospitalario, por lo tanto, se consideraron soluciones de software libre para el e-learning. El hospital eligió Claroline, una herramienta que permite el aprendizaje individual y en grupos, personalizado en cuanto a contenidos y tiempos de aprendizaje para cada alumno.

#### 10.1.5. La situación actual

Como resultado de estos cambios, 22 servidores del Hospital Beaumont ahora usan como sistema operativo GNU/Linux de Red Hat y de SuSe, y 14 usan Windows NT. Para el soporte, el mantenimiento y parte del desarrollo, el hospital tiene varios contratos con empresas locales de consultoría y desarrollo informático. Además, gran parte de los problemas de instalación y ejecución se resuelven a través de las listas de discusión de software libre de Slashdot y Sourceforge y los sitios dedicados a las aplicaciones instaladas.



Esta migración hacia el software libre ha requerido un cambio de mentalidad en la dirección del hospital y, sobre todo, en el Departamento Informático y en los técnicos informáticos y usuarios clave del hospital, acostumbrados a llamar a un servicio de ayuda del proveedor de software propietario.

Asimismo, la dirección del Departamento Informático considera hoy que otro aspecto importante de esta migración, y de los diferentes desarrollos propios realizados en base al software libre, ha sido la experiencia adquirida por el personal técnico del hospital en la manipulación de tecnología de la información, sobre todo, por ejemplo, en el procesamiento de imágenes. Este conocimiento y práctica no se hubieran podido adquirir con un software comercial.

Actualmente, el hospital está considerando la ampliación de su red de comunicaciones para mejorar el acceso a datos y la compra de monitores de alta resolución para permitir el diagnóstico seguro y consistente en línea a partir de las imágenes y radiografías digitales.

#### 10.1.6. Los aspectos financieros (estimados)

Finalmente, es importante considerar los aspectos económicos de los cambios, los cuales se resumen en la tabla siguiente. En total, se estima que el hospital se ahorra unos 13 millones de euros en 5 años.

**Tabla 10.**

<b>Cuadro financiero: costes comparativos indicativos</b>				
<b>En miles de euros</b>	<b>Solución software abierto</b>		<b>Solución de software propietario equivalente</b>	
Aplicaciones ofimática	27,5	34,7	120 p. ej., MS Office	288,5
Gestión de contenidos	20	32,1	126 p. ej., Lotus notes	140
Gestión de imágenes digitales	150	32,1	4.300	7339
Servidores de aplicaciones	10	60,5	302 p. ej., Websphere	595,3
Correo electrónico	1	8,7	110 p. ej., Lotus Domino	175
E-learning	1	4	35	175
<b>TOTAL</b>	<b>209,5</b>	<b>377</b>	<b>4.883</b>	<b>8.713</b>

### 10.1.7. Conclusiones

El director de los sistemas informáticos del hospital considera que esta migración hacia varios sistemas de software libre ha sido exitosa, desde los puntos de vista técnico y económico. No solamente han logrado ahorros de varios millones de euros, también han mejorado los servicios para los usuarios, tienen mayor control sobre sus aplicaciones médicas y han cambiado la mentalidad del personal técnico y usuario dentro del hospital. Asimismo, a partir del software libre utilizado, el hospital ha creado varios programas propios para la administración hospitalaria y la gestión de pacientes y de procesos médicos. Actualmente, se está considerando en qué medida el hospital puede participar y contribuir al movimiento de software libre, como contraprestación (u obligación legal) relativa al software utilizado.

#### Nota

Los programas mencionados en el texto y las condiciones y licencias asociadas, se pueden consultar en las URL siguientes

- OpenOffice.org: <http://www.OpenOffice.org>
- MySQL: <http://www.mysql.com>
- Staroffice:  
<http://www.sun.com/software/star/staroffice/>
- Zope: <http://www.zope.com>
- Claroline: <http://www.claroline.com>
- Jboss: <http://www.jboss.org>
- SuseMail: <http://www.suse.com>

### 10.1.8. Preguntas y actividades

En muchos estudios académicos y comerciales de software libre, su uso se ha considerado sobre todo desde el punto de vista económico

y técnico. Sin embargo, la experiencia del Hospital Beaumont permite considerar también varios temas legales vinculados a este tipo de software. Las preguntas siguientes apuntan desarrollar un análisis general de los aspectos legales de la implementación, sin entrar en los detalles textuales de las licencias, los contratos o cualquier garantía o indemnización, etc. (sin embargo, habremos de considerar sus líneas generales).

## El hospital

- 1) El hospital usa varias aplicaciones de software libre y mezcla éstas con software comercial. Esto requiere un análisis de los derechos y obligaciones del hospital:
  - a) ¿Cuáles son los derechos y las obligaciones del hospital frente al software instalado y el software desarrollado? (Sin tomar en consideración las licencias.)
  - b) Analizad las licencias libres en juego. ¿Cuáles son? ¿Son compatibles entre ellas y con otro software comercial? ¿Hay cláusulas que restringen su uso, modificación o distribución a través del hospital o fuera de él? ¿Cuál va a ser el efecto (vinculante o no) sobre la política informática del hospital en el futuro?
  - c) ¿Creéis que el acceso a los códigos fuentes y el derecho de modificación era y es importante para el hospital como usuario final (investigad el denominado *Berkeley Conundrum*)?
- 2) El Departamento Informático ha debido enfrentarse con varios problemas legales en la selección, implementación, modificación y uso de las nuevas aplicaciones. ¿Qué problemas legales habrán sido éstos y qué soluciones consideraréis que habría aportado el hospital?
- 3) Aparte de sus características económicas y técnicas, el software libre utilizado tiene varias ventajas e inconvenientes legales para usuarios finales como el hospital: analizad estas ventajas y desventajas. Explicad los riesgos y sugerid o definid unas estrategias (legales) para el hospital a fin de minimizarlos. ¿Qué tipos de cláusula exigiríais vosotros en los contratos de soporte y mantenimiento de software libre en una organización (pública o privada)?

### Extensión

- 4) El uso del software libre y sus ventajas y riesgos legales tienen efectos en otras áreas de la gestión del hospital, tales como los recursos humanos, la formación, la comunicación y la competencia con otras instituciones. Precisad estos efectos “extra-legales” y las áreas afectadas (por ejemplo, explicad cuáles son los cambios potenciales para el personal del hospital y, en particular, para el personal del departamento de sistemas) y comentad sus consecuencias. ¿Qué tipo de cambio de mentalidad ha requerido esta experiencia?
- 5) El uso del software libre exige, de cierta manera, una contribución a la comunidad de software libre. ¿En qué aspecto podría (o, en vuestra opinión, ha podido) el hospital realizar esta contribución de manera más beneficiosa o eficiente?

## 10.2. Lorena Nadal

Objetivo de estudio:

- Considerar los aspectos legales relevantes en la implementación de software libre para un particular.

Métodos de estudio:

- Lectura del caso.
- Investigación en Internet de las aplicaciones referidas.
- Análisis de las licencias en cuestión.

### 10.2.1. Introducción

Las obligaciones y responsabilidades de los proveedores de bienes y servicios varían según la naturaleza del cliente, ya sea éste una empresa o sea particular (“consumidor”). En este estudio consideraremos el caso de un individuo que, en su capacidad personal y luego profesional, descarga e instala software libre.

### 10.2.2. Lorena Nadal

Lorena Nadal, contable y auditora, ejerce su profesión como autónoma. Se interesa por el software libre y, tomando consejo de su amigo desarrollador Jordi Puig, se conecta a la página de Debian (<http://www.debian.org>) y a otros sitios de software libre (Sourceforge.net, Mozilla.org, Openoffice.org) para descargar una serie de aplicaciones de software libre, que incluye lo necesario para hacer funcionar su ordenador personal en casa. Entre otros, con la ayuda de J. Puig, instala:

- El sistema operativo Debian/GNU Linux y elementos para establecer un entorno de trabajo gráfico (X Windows, KDE, etc.).
- OpenOffice.org, para un paquete de ofimática.
- Mozilla, para correo electrónico y navegación en Internet.

Un día, Lorena tiene un problema informático y pierde unos datos importantes. J. Puig estudia el problema y considera que se trata de un problema con el paquete de software libre instalado, que tenía un *bug* (error) que provocaba una pérdida de datos en ciertas circunstancias (la causa del problema no es relevante).

### 10.2.3. Preguntas y actividades

- 1) Respecto del software libre elegido por Lorena, buscad las licencias relevantes y comentad las cláusulas de exoneración de responsabilidad y garantía.
- 2) En relación con los recursos legales disponibles para Lorena
  - a) ¿Contra quién puede reclamar? (causante del daño; si consideráis que Lorena puede demandar a uno de los autores o distribuidores de software, elegid uno).
  - b) ¿Por qué motivos? (base legal)
  - c) ¿Qué va a pedir? (¿daños y perjuicios?)
- 3) ¿Qué obligaciones/responsabilidades tienen las personas siguientes frente a Lorena?
  - a) El autor de software.

- b) El distribuidor en Internet.
  - c) Jordi Puig, quien le ayuda a instalar el software en cuestión.
- 4) ¿Qué deberían hacer las personas siguientes para protegerse del mejor modo posible (legalmente) contra reclamos potenciales?
- a) El autor de software.
  - b) El distribuidor de software.
  - c) Jordi Puig.
- 5) ¿Qué diferencia habría si:
- a) Lorena hubiera instalado los programas en su ordenador de trabajo y los datos perdidos fuesen importantes para su negocio como contable?
  - b) Lorena hubiera conseguido y comprado la misma serie de programas de un distribuidor comercial de software libre como Red Hat, Mandrake o SuSe (elige uno)?
  - c) Su amigo Jordi Puig hubiera facturado a Lorena sus servicios de consultoría e instalación del software?

### 10.3. Kerberos

Objetivos de estudio:

- Estudiar un ejemplo de desarrollo basado en estándares.
- Entender la relación entre estándares, propiedad intelectual y software libre.
- Analizar la relación entre secreto comercial, libertad de expresión y propiedad intelectual.

Métodos de estudio:

- Lectura del caso Kerberos.
- Investigación en Internet.
- Discusión.
- Redacción de carta de denuncia y de defensa.

### 10.3.1. Introducción

Ya se sabe que Internet no es un “lugar” seguro. Los protocolos de Internet como SMTP, MIME o HTTP no incluyen mucha seguridad, aunque actualmente pueda haber versiones más seguras de éstos como el S/MIME y el HTTPS (con SSL). Uno de los mayores problemas de seguridad reside en el uso de nombres de usuario y claves de acceso para acceder a sitios restringidos y datos confidenciales. Esto ocurre a menudo en aplicaciones distribuidas, en las cuales el servidor frecuentemente se fía en los clientes para la autenticación y la autorización del usuario. Asimismo, estas aplicaciones son vulnerables a programas perjudiciales que buscan las fallas en la transmisión de claves no cifradas en las redes. Además, hay varios sectores para los que es necesario mantener un alto nivel de seguridad en las redes internas o privadas, no solamente por causa de la cyber-criminalidad interna de las organizaciones, sino también más prosaicamente en sistemas en los que los usuarios tienen diferentes privilegios (por ejemplo, los bancos, los hospitales y las universidades).

### 10.3.2. Kerberos

Kerberos es un protocolo de autenticación para equipos organizados en red. Provee una autenticación de usuario de alto nivel (o “robusta”, basada en el cifrado de alto nivel), para aplicaciones servidor/cliente. Básicamente, permite una intercomunicación segura entre equipos clientes y servidores dentro de una red, sobre la base de nombres de usuario y claves de acceso. Por lo tanto, en un sistema distribuido, los clientes autorizados de la red pueden acceder a los recursos y dominios controlados por el servidor.

Kerberos fue desarrollado por el *Massachusetts Institute of Technology* (MIT) durante los años 1980-1990, junto con el X Windows System.

Kerberos genera y distribuye de manera segura claves de acceso (en *Key Distribution Centres*, KDC) a través de *tickets* intercambiados. El sistema permite realizar la autenticación del usuario (la comparación de nombre de usuario con claves actualizadas) y una extensión permite realizar la autorización (la descripción y averiguación de los privilegios de acceso).

#### Lectura complementaria

Para información técnica sobre Kerberos, consultad: MIT: *Kerberos: The Network Authentication Protocol* en bibliografía.

#### Nota

La última versión liberada es Kerberos 5.0, disponible en:  
<http://web.mit.edu/kerberos/www/>

Actualmente, Kerberos es un estándar abierto, adoptado por el Internet Engineering Task Force (IETF) en 1993-1996, para permitir la interoperabilidad de clientes y servidores en la red. Para entender bien este asunto, es importante notar que en la especificación del estándar, el IETF dejó abierto, es decir sin especificación, un campo de datos que se puede utilizar para realizar extensiones personalizadas del protocolo. El Open Group, que desarrolla DCE Kerberos, usa este campo para la identidad del usuario y publica abiertamente el formato para que otros sistemas puedan ser interoperables con su plataforma.

**Nota**

Hubo varias versiones de los estándares del IETF. Se pueden leer en:

<http://www.ietf.org/rfc/rfc1510.txt?number=1510>

<http://www.ietf.org/rfc/rfc1964.txt?number=1964>

**10.3.3. La distribución de Kerberos**

Hay implementaciones del protocolo Kerberos tanto en productos propietarios como en aplicaciones libres. El MIT ha desarrollado una implementación libre que se distribuye en el sitio arriba mencionado, incluido el código, bajo una licencia similar a la de BSD y X Window System. Se usa, sobre todo, en redes que necesitan altos niveles de seguridad, como por ejemplo bancos, sistemas militares y universidades. Sin embargo, su uso está siendo remplazado por la implementación (lenta) de varias plataformas PKI.

**Reflexión**

Ved el procedimiento de acceso y la información sobre Kerberos en el MIT. Notad, sobre todo, el tema de la exportación de productos de cifrado, pero también leed la licencia aplicada al software.

A finales de los años 1990, Microsoft participó en el desarrollo del protocolo e incorporó una implementación en lo que en ese momento se llamaba Windows NT 5.0, hoy llamado Microsoft Windows 2000. Y,



justamente, es esta “*deployment*” de Kerberos en Windows 2000 la que ha causado tantos problemas en la comunidad de seguridad y de software libre en 2000 y 2001. A continuación, explicamos el origen y la evolución del conflicto.



Para una implementación propietaria de Kerberos, podéis ver los enlaces en la página del MIT:

<http://web.mit.edu/kerberos/www/>

Por ejemplo, CyberSafe Limited en:

<http://www.cybersafe.ltd.uk/>

Para una versión libre, podéis ver Heimdal en:

<http://www.pdc.kth.se/heimdal/>

#### 10.3.4. La versión de Windows 2000

La implementación de Windows 2000 sigue la especificación de Kerberos 5, pero en el campo de la especificación dejado abierto incluye un dato de formato no público o “cerrado”. Este campo consiste en datos que forman un certificado de acceso, describiendo los privilegios del usuario (*Privilege Access Certificate*, PAC). Este certificado permite el acceso a los dominios y servicios correspondientes (autorizados) de la red controlada por un servidor con Windows 2000, y por lo tanto acceso a los diversos recursos en la misma (aplicaciones, archivos, *drivers*, etc.). Pero el certificado “cerrado” se agrega a los *tickets* estándar de Kerberos, lo que implica que *tickets* generados por versiones de Kerberos que no son de Windows 2000 no son compatibles. Como el formato del campo es secreto, otros desarrolladores no pueden crear KDC que generen *tickets* compatibles.

El resultado de la extensión propietaria de Kerberos de Microsoft es que otros sistemas operativos de red, como UNIX o Linux, con una implementación de Kerberos diferente, no pueden comunicarse con seguridad con servidores o clientes con Windows 2000. No saben el formato del campo de la extensión y, por lo tanto, no pueden verificar la autorización necesaria. La consecuencia práctica de ello es que a causa de las dificultades prácticas para realizar una interfaz

“manual” entre sistemas cliente/servidor mixtos (es decir, con UNIX en el servidor y Windows 2000 en el cliente, por ejemplo) y los mayores riesgos de la comunicación entre ellos, esta extensión propietaria favorece el uso exclusivo de productos Microsoft.



El efecto de esta práctica se ha llamado *lock-in* (un cerco), combinado con una estrategia de “*embracing, extending and extinguishing*”, es decir acoger, extender y apagar. Una empresa se acoge al estándar abierto para asegurar una interoperabilidad básica y lo extiende con versiones propietarias que instala e integra en sus productos. Cuando esta empresa tiene una posición dominante en un mercado, esto provoca una distorsión del concepto de estándar abierto. Los productos de los competidores basados en el estándar no son más compatibles con los prevalecientes de la empresa dominante, en este caso Microsoft. Por lo tanto se extingue la competencia.

El efecto de cerco reside en el hecho de que los usuarios del producto no estándar deben seguir implementando productos compatibles, normalmente vendidos por la misma empresa o empresas asociadas que han pagado la licencia de propiedad intelectual o industrial (patente, derecho de autor). Esta táctica no es propia únicamente de Microsoft; por ejemplo, IBM la usó en los años 1960, en relación con el micro-código, y AT&T, en relación con los *switches* de telefonía.

En consecuencia, los usuarios que instalan o quieren instalar Windows 2000 (por ejemplo, en cliente, debido a su alta usabilidad y a la disponibilidad de aplicaciones de sobremesa) están “encerrados” y obligados a seguir con plataformas de Windows. Pierden la libertad y la oportunidad de usar alternativas que se podrían considerar más seguros, como UNIX, Solaris o Linux.

#### 10.3.5. El debate

En 1998, Microsoft prometió publicar la especificación de su extensión propietaria. En efecto, en abril de 2000, dos meses después del

lanzamiento de Windows 2000 y justo durante el juicio por abuso de posición dominante, Microsoft publicó la especificación, pero no de manera abierta. Distribuyó la documentación en un paquete que, al instalarse, desplegaba una licencia de uso, con términos de confidencialidad y licencia de propiedad intelectual. Los mismos términos protegían el sitio de Internet donde también se publicó la especificación. La aceptación de estas condiciones legales implicaba que cualquier lector tenía que aceptar una licencia otorgada por Microsoft y no desarrollar productos que pudieran competir con los suyos (es decir, un KDC implementando Kerberos con la extensión de Microsoft).



La parte importante de la licencia declara:

“b. Esta especificación es información confidencial y un secreto comercial de Microsoft. Por lo tanto, usted no puede revelar la misma a nadie (excepto tal como se permite más adelante) y debe tomar las precauciones de seguridad razonables –tan seguras como para sus propios datos confidenciales– para mantener el secreto de la especificación. Si se trata de una organización, usted puede revelar esta especificación a sus empleados de tiempo completo, sobre la base de “need to know” y siempre y cuando se haya pactado con sus empleados acuerdos de confidencialidad adecuados para permitir a la organización cumplir con este acuerdo.”

Este proceso enfureció a la comunidad del software libre. Hasta el equipo de desarrollo del MIT demostró su disconformidad con el hecho de que un estándar abierto se viera comprometido por la combinación de una posición dominante y una licencia propietaria. Se negaron a pagar la licencia para incorporar los PAC de Microsoft (o un generador *clone* –o copia– de claves de PAC) en la versión libre disponible en los servidores del MIT.

Varios desarrolladores de software abierto accedieron a la documentación (aparte de su desdén por la licencia de Microsoft, había una manera de acceder a ella sin tener que aceptar la licencia, a través de la descompresión del ejecutable) y publicaron la especificación en las listas Slashdot. Al darse cuenta de esto, Microsoft envió una carta legal a Slas-

hdot, requiriendo la eliminación inmediata de los mensajes de la lista, so pena de sanciones legales bajo el derecho de propiedad intelectual, apoyadas por la reciente DMCA, y por violación de secreto comercial. Andover.net, dueño de Slashdot, publicó la carta de Microsoft y mantuvo los mensajes, acogándose a la protección otorgada por la constitución americana (la libertad de expresión) y el hecho de que Microsoft había ya publicado la especificación en Internet sin debidas medidas de protección, entre otras defensas.

#### Nota

Bajo la ley DMCA en EE.UU., un interesado puede pedir a un intermediario de Internet (por ejemplo, empresas de *hosting*) que retire de la web determinados archivos o textos que vulneran derechos del interesado (derechos de autor, reputación, datos personales). La notificación se llama *take down notice*, un aviso de descolgar, abreviada en TDN. Una vez avisada, la empresa de *hosting* puede incurrir en daños si no elimina los materiales presuntamente perjudiciales. Esto pone a las empresas de *hosting* en una posición poco envidiable, como jueces de lo que pueda vulnerar (o no) los derechos de terceros, sin tener una ocasión para permitir el ejercicio, por parte de la persona que subió los archivos en cuestión, de su derecho de audiencia.

El IETF consideró cambiar la especificación de Kerberos, para que el producto de Microsoft no cumpliera con el estándar. La idea sería redactar la especificación para que se defina el campo actualmente abierto. Asimismo, hay dudas sobre el origen de la extensión de Microsoft. Como el estándar se desarrolló de manera abierta, hubo muchas discusiones en las listas y BBS correspondientes, sobre distintos mecanismos de autorización posibles. Se ha alegado que la extensión en cuestión es un código derivado de estas discusiones y que, por lo tanto, no está protegido ni por los derechos de autor ni por el secreto comercial.

#### 10.3.6. Conclusiones

Este caso ilustra muchos temas que hemos desarrollado en este curso: los derechos de autor, el secreto comercial, las licencias li-

bres y cerradas, los estándares... Es importante percatarse de que muchos de estos temas están interrelacionados y, sobre todo, de que hay una relación particular entre **apertura legal** y **apertura técnica**, que es lo que intentaremos definir. Una posición en un campo obliga a tomar posiciones en el otro. Sin embargo, esto no significa que no haya matices, compromisos y posiciones intermedias. Muchos de los que trabajan en los foros de estandarización son empleados de las empresas más “propietarias”, como IBM, Intel, Microsoft, etc.

Finalmente, dejamos la historia de Kerberos abierta, porque una de las actividades a realizar consiste en buscar y comentar cómo terminó el asunto (si es que se ha terminado). ¿Qué pasa con la extensión PAC de Microsoft? ¿Y con el estándar IETF? ¿Cómo van las implementaciones de Kerberos en general; tienen un *deployment* amplio?

#### 10.3.7. Preguntas y actividades

- 1) ¿Cómo se aplica el derecho de la propiedad intelectual a las diferentes “obras intelectuales” presentes en este caso (el estándar Kerberos, las implementaciones, la extensión de Microsoft y su documentación, etc.)? ¿Quién es titular de qué? ¿Quién puede proteger qué?
- 2) Buscad y comentad la licencia de acceso a la especificación de PAC (pista: los históricos de Slashdot...). Según vosotros, ¿esta licencia es vinculante? ¿Es válida?
- 3) Redactad (en términos generales) las líneas generales de la carta de Microsoft a Slashdot y la defensa de Slashdot a Microsoft.
- 4) Imaginad que una empresa independiente implemente la extensión PAC de Microsoft en una versión de Kerberos. ¿Sería defendible desde el punto de vista legal? ¿Cuáles serían las consecuencias legales? ¿Cuáles serían los puntos importantes a incluir en una carta que pudiera mandar Microsoft contra esta empresa, por haber utilizado su protocolo....? ¿Y la defensa de la empresa?
- 5) Comentad la relación entre estándares, propiedad intelectual e industrial y código abierto. ¿Cuál es el papel de los estándares? ¿Son compatibles con los derechos de autor o las patentes?

- 6) Considerad la relación entre propiedad intelectual, secreto comercial y libertad de expresión, en vuestra jurisdicción. ¿Cuál prevalece, en las condiciones descritas en este estudio? ¿Se podrá generalizar una regla?
- 7) ¿Cómo consideráis las disposiciones de la Directiva de Derechos de Autor, en la Sociedad de la Información y la DMCA, relativas al amparo de los mecanismos de protección de derechos de autor? ¿Qué efectos pueden tener sobre los estándares? ¿Y sobre el software libre?
- 8) Desde esta perspectiva, investigad y comentad la decisión sobre la publicación de DeCSS (California, 25 de agosto de 2003) y el caso Jon Johanson de Noruega.
- 9) Investigad el asunto de Kerberos y Windows 2000 en Internet. ¿Cuál es la posición actual (pensad en Windows Server 2003)?
- 10) ¿Cuál es, en vuestra opinión, la estrategia de Microsoft relativa al mercado de servidores? ¿Por qué?

#### 10.4. El software libre y la protección de datos

Objetivos de estudio:

- Determinar los factores importantes a tomar en cuenta en el momento de implementar un sistema informático que procese datos personales.

Métodos de estudio:

- Especificación de diseño de sistemas.
- Consideración sobre las ventajas e inconvenientes del software libre en el tema de la seguridad.

##### 10.4.1. Introducción

En la unidad 8, habéis visto un esbozo de la regulación compleja de la protección de datos personales y su relación con el software. En

relación con este tema de los datos personales, vamos a realizar dos ejercicios cortos.

El primero consiste en analizar un breve caso hipotético. En el segundo, en vez de considerar un caso particular, vamos a realizar un ejercicio de especificación y diseño de arquitectura de sistema informático que incorpore los requisitos legales necesarios para cumplir con la LOPD.

#### 10.4.2. Preguntas

- 1) En un hospital, vosotros diseñáis y controláis una base de datos de pacientes, con nombres y otros datos personales. Estos datos se transmiten a través de las redes de Internet a navegadores de los pacientes y médicos que consultan la base.
  - a) ¿Cuáles son los derechos de los pacientes en relación con estos datos?
  - b) ¿Cuáles son las obligaciones que debe cumplir el responsable de tratamiento de datos?
  - c) ¿Qué medidas de protección tomaríais en relación con el proceso indicado? ¿Hay programas de software libre que os puedan ayudar?
- 2) En los casos siguientes, ¿cuáles son los procesos legales específicos que habría que incorporar al diseño del sistema y de su implementación?
  - a) La creación de un fichero o base de datos de miembros de una congregación de una iglesia.
  - b) La recogida de datos de usuario en un formulario de Internet.
  - c) La subcontratación del procesamiento de ficheros que contienen datos personales a una empresa tercera (*hosting, ASP, out-sourcing*).

### 10.5. Trusted Computing o la “informática fiable”

Objetivos de estudio:

- Conocer un ejemplo de Sistema de Gestión de Derechos de Propiedad (DRMS).

- Entender la iniciativa de Trusted Computing (la “informática fiable”), y su relación con el software libre.

Métodos de estudio:

- Presentación y discusión de la iniciativa Trusted Computing.
- Lectura e investigación en Internet.

En la unidad 2 se ha hablado del concepto de derechos de autor y de los nuevos sistemas para sugestión, más conocidos por su abreviación en inglés: DRMS. Se han hecho varias propuestas para implementar este concepto, con sistemas de identificación de obras digitales y, por ejemplo, con la conexión a Internet para obtener el permiso de uso adecuado contra el pago de sumas pequeñas o mayores. El sistema iPod de Apple implementa una versión simple: los usuarios que quieren obtener una canción después de escuchar una demo de 30 segundos deben pagar la suma de 99 céntimos por cada canción que descargan del sitio de Apple. Luego, Apple paga los derechos correspondientes a los autores y a las empresas discográficas.

En este apartado presentaremos y comentaremos una implementación más compleja de este concepto, diseñado en un nivel muy “bajo” en términos informáticos: en el funcionamiento mismo del procesador central (chip) y del sistema operativo del ordenador.

### 10.5.1. El concepto de Trusted Computing (TC)

La iniciativa de la “informática fiable”, apoyada por la Trusted Computing Platform Alliance (usaremos el acrónimo inglés, TCPA), apunta a mejorar la seguridad de los sistemas y las plataformas informáticos. Se trata de establecer estándares para el diseño de hardware y software, además de reglas para la interoperabilidad, que garanticen una mayor seguridad en el procesamiento de datos. El término *fiable* implica que terceros que interactúan con cierta plataforma pueden fiarse de ella, ya sea porque el dueño ha sido identificado y autorizado o porque se puede averiguar (o ya se ha averiguado) que sus sistemas informáticos no han sido modificados “perjudicialmente”. Basado en este concepto, por ejemplo, dos pro-

#### Lectura complementaria

Para más información sobre la TCPA, podéis ver:

<http://www.intel.com/es/home/trends/future/trustedcomputing.htm>  
<http://www.trustedcomputing.org>



gramas en sistemas distribuidos pueden comunicarse entre sí de manera segura.

### 10.5.2. Ejemplos: gestión de derechos y Palladium

El mejor ejemplo de aplicación de la TC es la gestión de derechos de autor. Si los ordenadores integran sistemas TC, los autores y las organizaciones de gestión de derechos de autor pueden asegurarse de que un usuario esté utilizando una obra protegida únicamente de manera legítima (por ejemplo, su lectura o escucha, pero no su copiado ni su transmisión a terceros) o ejecutando un programa legal (o sea, no pirateado). Si se trata de una acción o una versión ilegal, pueden impedir su ejecución. Por ejemplo, se podrá controlar que un usuario no descargue y copie de Internet una obra protegida (un DVD, música, etc.) sin obtener el permiso previo del titular de los derechos. De esta manera, los autores y titulares de derechos pueden proteger sus derechos y controlar el uso y re-uso de sus obras.

Un ejemplo de este modelo es el Palladium, de Microsoft (ahora NGSCB –Next-Generation Secure Computing Base), que puede vincularse con el servicio de gestión de licencias de Microsoft y para fines comerciales y de identificación de usuarios, con el servicio de identidad Passport (de .Net). Por extensión de lo que se puede hacer con obras literarias o musicales específicas, con una gestión similar de accesos, firmas y autorizaciones, los autores de correos y otros documentos electrónicos podrán controlar a los destinatarios y a lectores de los mismos (por ejemplo, para mantener el secreto o la confidencialidad) y regular hasta la duración de su existencia (como la consabida advertencia de “esta carta se autodestruirá en 10 segundos”).



Como lo ha dicho un internauta: “En pocas palabras, en un futuro no demasiado lejano, todos los PC vendrán con un chip, el cual comprobará que la licencia de Windows o del programa que instalemos esté firmada digitalmente por un organismo seguro.”

<http://listas.gula-zale.org/gula/2003-June/001483.html>

#### Lectura complementaria

Sobre NGSCB , podéis ver, por ejemplo:

<http://www.computerworld.com.co/noticias/noti19.htm>

<http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/news/NGSCB.asp>

**Lectura complementaria**

Un comentario interesante en R. Anderson: *Preguntas Frecuentes sobre Informática Fiable* (en bibliografía).

**10.5.3. La relación con el software libre**

La TC ha sido atacada por varias razones, entre otras las relativas al respeto del derecho de la propiedad intelectual (por ejemplo, el control potencial sobre la copia privada o de seguridad o la restricción del derecho de prestar una obra a otra persona). Asimismo, se critica el potencial cambio en el equilibrio de poder en relación con el control y el gobierno de la sociedad actual.



Para más información sobre los efectos de la Trusted Computing, se recomienda leer los escritos de R Andersen (*'Informática de confianza' y política sobre competencia: temas a debate para profesionales informáticos; Preguntas Frecuentes sobre Informática Fiable*

TC / TCG / LaGrande / NGSCB / Longhorn / Palladium / TCPA) y L Lessig (*El futuro de las Ideas*), citados en la bibliografía.

Además de estos debates, la implementación de la TC tiene varias implicaciones para el software libre y para la privacidad.

En primer lugar, la TC sería incompatible con plataformas (sistemas operativos) de software libre, porque sus preconizadores consideran que cualquier código abierto es “modificable” y, por lo tanto, es inseguro. La Alianza TC argumenta que el software libre permite entrar en la “mecánica” de los procesos y desviar flujos de datos protegidos (por ejemplo, para copiar un DVD que se está reproduciendo).

En segundo lugar, la implementación de la TC extenderá el control de los fabricantes de aplicaciones dominantes: Microsoft, Intel, HP, Sun, etc. (un fenómeno llamado *lock-in* en inglés). Éstos podrán:

- Imponer límites a las aplicaciones que se ejecutan en sus equipos o sistemas operativos y a los archivos manipulados por ellos (por ejemplo, únicamente se podrán usar bases de datos DB2 sobre un sistema operativo de IBM).

- Prohibir la apertura de un archivo creado con una aplicación propietaria (por ejemplo, WordPerfect de Corel) con otro programa de tipo “abierto” (por ejemplo, OpenOffice.org).
- Impedir la migración de una plataforma a otra (por ejemplo, de Solaris a Linux) por razones no solamente de compatibilidad, sino también de control de derechos y de accesos, y de regulación de la exportación de archivos.

Las autorizaciones de acceso y uso serán codificadas dentro del mismo archivo y es posible imaginar casos en los cuales, si los autores de los documentos pueden controlar permanentemente los usos de los mismos, cuando una empresa quiere migrar de un sistema a otro tendrá que pedir el consentimiento de todos y cada uno de los autores de la documentación empresarial.

En tercer lugar, estos sistemas de TC están protegidos por las nuevas leyes de propiedad intelectual en la “era de la información”: el DMCA en EE.UU. o la Directiva sobre los Derechos de Autor en la Sociedad de la Información de 2001 en la Unión Europea (EUCD). Como se ha visto en la unidad 2 de este curso, ambas leyes amparan los sistemas tecnológicos de protección de los derechos de autor contra las actividades y los “dispositivos” de elusión y prohíben la comercialización y distribución de dichos dispositivos (art. 6 y 7 de la EUCD). Los sistemas abiertos del software libre permiten modificar el código fuente de las aplicaciones y, como en el ejemplo citado arriba, desviar el flujo de datos de un archivo protegido. Esto implica que los programas libres podrían considerarse dispositivos o programas ilegales.

El hecho de que estos sistemas (de TC) estén vinculados íntimamente con la gestión de la identidad es de mayor relevancia en este rubro sobre los datos personales. Para averiguar la autorización de acceso o de explotación de una obra protegida, estos sistemas exigen mantener bases de datos centralizadas de datos personales (nombre, apellido, datos financieros, autorizaciones, etc.) que la “plataforma fiable” puede consultar para otorgar o no el acceso y el uso.

Finalmente, la TC implica un debate importante en el área de los estándares. La TCPA reivindica varios estándares para la informática

fiable (para la configuración del hardware o para la conexión a redes) que, si bien pueden aumentar la seguridad en las redes y los sistemas informáticos y aportar una mayor protección de la propiedad intelectual, pueden también tener efectos perjudiciales para el movimiento de software libre, en términos de compatibilidad con el hardware, interoperabilidad con aplicaciones de distintos tipos y de exclusión de las redes públicas basadas en aquellos estándares.

#### 10.5.4. Preguntas

- 1) Investigad la iniciativa de Trusted Computing y formulad los argumentos a favor y en contra.
- 2) Comentad el efecto de TC sobre el software libre.

### 10.6. La exportación de productos de seguridad

Objetivo de estudio:

- Esclarecer la relación entre el software libre y el control de productos de cifrado.

Método de estudio:

- Comentario de avisos de restricciones.
- Investigación en Internet sobre diferentes productos de cifrado y la política de su desarrollo.

#### 10.6.1. Introducción

La preocupación por la seguridad como buen elemento del diseño y el desarrollo de aplicaciones y la incorporación de tecnologías de cifrado en los productos libres o abiertos, ha sido un principio de desarrollo de varios programas del movimiento libre. Las siguientes aplicaciones muestran el alcance de los programas disponibles. En este estudio, vamos a considerar la relación entre el software libre y el control de los productos de cifrado por las administraciones públicas.

### 10.6.2. Algunos productos de software libre para la seguridad

Correo electrónico:

- El **PGP**, desde su principio, se ha distribuido con una licencia libre para el uso privado (de tipo *Freeware*).
- Hay una versión GNU de PGP, llamado **GNU Privacy Guard** (<http://www.gnupg.org/>), conforme al estándar **OpenPGP**.
- Se puede agregar varias extensiones a Linux para el cifrado de correo electrónico (por ejemplo, **KMail** usando PGP/GnuPG basado en el estándar OpenPGP).

Servicios de red:

- El **UNIX SSH1** es libre (<ftp://ftp.cs.hut.fi/pub/ssh/> o <http://www.ssh.fi/>). Hay también OpenSSH en: <http://www.openssh.com/es/>.
- Se desarrolla **LSH**, una versión libre de SSH2 (podéis ver en: <http://www.net.lut.ac.uk/psst> y <http://www.lysator.liu.se/~nisse/archive>).
- **Mozilla** soporta el cifrado de alto nivel, con SSL (SSLey).
- El servidor web (**Apache**) soporta SSL.
- Hay varios otros programas para servicios seguros en redes, como **OpenLDAP**, que permite acceder a directorios por SSL.

### Sistemas operativos y aplicaciones

- **Linux**: hasta 2003, Linux no se distribuía en o desde los EE.UU. con cifrado de alto nivel (por las restricciones gubernamentales que veremos a continuación). Hay que incorporar un parche para el núcleo de Linux, el "International Linux Kernel Patch" (podéis ver en <http://www.kerneli.org/>), para que esté conforme a las versiones

**Nota**

podéis ver kerberos en:  
<http://web.mit.edu/kerberos/www/>

europas. Para productos libres integrables con Linux, podéis ver en <http://www.linux.org/apps/all/System/Encryption.html>.

- Otras versiones de Linux fuera de EE.UU., como **SuSe Linux** de Alemania, tienen criptografía de alto nivel.
- El sistema operativo libre con más herramientas de seguridad es el **OpenBSD** (<http://www.openbsd.org/>) que, por ejemplo, se distribuye directamente con componentes para implementar VPNs. OpenBSD se distribuye desde Canadá.
- Asimismo, hay varias aplicaciones, módulos o extensiones para reforzar la seguridad de GNU/Linux, como **Kerberos**, un protocolo de autenticación para arquitecturas cliente/servidor

Como se investigará en las actividades prácticas de esta sección, el uso de cifrado de alto nivel ha provocado varios problemas para el movimiento de software libre, relacionados con la distribución libre de software por Internet. Esto es así porque, como lo hemos visto en la unidad 8, varios gobiernos, entre los cuales el de EE.UU. ha sido el más activo, han intentado controlar la exportación de productos de criptografía.

### 10.6.3. Preguntas y actividades

Los controles de productos de cifrado han levantado quejas y denuncias por parte de toda la comunidad del software libre, de los académicos y, a veces, de los desarrolladores de software propietario. Solamente en el 2002, Netscape y Microsoft pudieron incorporar el cifrado a 128 bits en los navegadores de Internet (Navigator 6 e IE 5.5 y versiones anteriores con parches), para una mayor seguridad de las conexiones, por ejemplo, para las transacciones bancarias o gubernamentales en línea.

- 1) ¿Cuáles son los argumentos a favor y en contra de estos controles a la exportación? ¿Estáis de acuerdo con ellos?
- 2) En relación con el software libre y, sobre todo, con su modelo de desarrollo (considerar el desarrollo de Linux o de Mozilla), ¿cuáles

son los riesgos potenciales y las medidas adecuadas para asegurarse de no violar ningún control?

- 3) Buscad en Internet el sitio de desarrollo de una aplicación que use o desarrolle componentes de cifrado y comentad su política de desarrollo y de distribución.
- 4) Considerad las FAQ y otras páginas de Mozilla sobre la seguridad y el cifrado (en mozilla.org). Haced una lista de las distintas páginas que comentan este tema (hay por lo menos cuatro). Comentad la situación actual para Mozilla.
- 5) Buscad en Internet una declaración o aviso referente a la exportación de productos de cifrado. Comentad su validez actual.

## 10.7. El caso SCO

Objetivos de estudio:

- Comprender los riesgos legales en el desarrollo, distribución y uso del software libre.
- Considerar los efectos de la licencia GPL.
- Evaluar el uso de patentes como herramienta legal en relación con el software.

Métodos de estudio:

- Exposición del caso SCO, IBM y Red Hat, basado en el desarrollo de GNU/Linux V 2.4.
- Análisis del caso.
- Investigación de varios temas relativos al caso (en Internet).
- Preguntas y discusión.

### 10.7.1. Introducción

En marzo de 2003 (modificado –reducido– en junio de 2003), la empresa SCO Group (ex Caldera) inició una demanda contra IBM, por violación de derechos de autor e incumplimiento de contrato, en relación con ciertos módulos de software incorporados a la versión 2.4 de GNU/Linux. Asimismo, SCO envió una carta a varias empresas grandes y OEM, pidiendo derechos de licencia y regalías por el uso de Linux. El 6 de agosto de 2003, IBM contesta e inicia una contra-demanda contra SCO por interferencia en sus negocios, violación de la licencia GNU GPL y violación de patente. Además, debido al efecto negativo de las demandas legales de SCO contra IBM y empresas involucradas en la distribución y soporte del sistema operativo Linux, la principal distribuidora del mismo en EE.UU., Red Hat Inc., inicia una demanda contra SCO por competencia desleal y publicidad engañosa. Así se inició la primera batalla entre el modelo de software propietario y el modelo abierto o libre.

#### Nota

OEM (*Original Equipment Manufacturer*) son empresas que montan ordenadores y dispositivos a partir de componentes y software de terceros. Ejemplos son DELL, Gateway, etc.

Esta confrontación es interesante y útil para estudiar diversos aspectos legales del modelo de desarrollo de software libre y los riesgos de su distribución y uso. Primero, para entender las bases de este conflicto y las demandas entabladas, hay que revisar la historia del desarrollo de los sistemas operativos UNIX y GNU/Linux. Luego, consideraremos los hechos y alegatos de los litigios en cuestión. En este estudio de caso, hemos usado notas al pie para daros las referencias actuales de las citas, las críticas y los documentos originales.

### 10.7.2. Telón de fondo: el desarrollo de sistemas operativos

Para entender el conflicto, se debe tomar en consideración el cambio fundamental que se ha producido en la industria del software “de infraestructura” en los últimos 20 años. El rápido desarrollo del



hardware (que duplica su capacidad cada 18 meses) implica que los sistemas operativos y su complejidad deben modificarse a un ritmo comparable. Esto tiene un efecto profundo sobre la ingeniería de sistemas y sobre los modelos de desarrollo de los sistemas operativos. Han aparecido nuevos modelos eficientes de desarrollo descentralizado de software, iniciados sobre todo por la comunidad UNIX y Linux (Berkeley UNIX, GNU, X Consortium), aprovechando las comunicaciones permitidas por Internet. Estos modelos invierten las premisas del desarrollo de software industrial basado en el control y la jerarquía. Hoy, casi se puede decir que únicamente empresas grandes como Microsoft, IBM y Sun han podido evitar el nuevo modelo de desarrollo. Este nuevo proceso tiene implicaciones legales sin precedentes, que surgen en esta confrontación entre SCO e IBM y Linux.

### 10.7.3. El desarrollo del software UNIX original y de Linux

Uno de los sistemas operativos principales para sistemas informáticos empresariales e industriales es UNIX. Actualmente el término *UNIX* describe una familia de sistemas operativos que comparten elementos comunes de diseño, basados en el primer UNIX desarrollado en los laboratorios Bell de AT&T. A lo largo de los años, se han desarrollado muchas versiones de este sistema.

#### Nota

El estándar UNIX, hoy establecido por The Open Group, está compuesto por la *Single UNIX Specification* y el POSIX.

La evolución de las distintas versiones de UNIX es compleja. Básicamente, UNIX nació en los laboratorios de Bell, en los Estados Unidos, en los años 1960. En su desarrollo y durante muchos años, muchos programadores modificaron y contribuyeron abiertamente (en el sentido de código “abierto”) al desarrollo del sistema operativo y varias organizaciones distribuyeron el código de UNIX sin restricciones. En particular, a partir de 1975, programadores de la Universidad de California en Berkeley (UC Berkeley), contribuyeron considerablemente a la creación del código de UNIX. En 1977, éstos crearon su

propia versión del sistema operativo, llamado el UNIX/BSD (Berkeley Software Distribution).

### Las versiones de UNIX

Como hemos visto, la versión “central” de UNIX, desde la versión 7 hasta la versión “System V”, fue desarrollada por Bell Labs, de AT&T. En 1990, AT&T formó una unidad de negocio y subsidiaria llamada UNIX Systems Labs (USL) para desarrollar el UNIX como producto comercial.

Otras versiones del sistema son:

- **UnixWare:** en 1992, Novell adquirió USL de AT&T y, con ella, los derechos en la versión UNIX de AT&T. Estos derechos incluyeron también los derechos en los contratos de licencia entre AT&T y distintos desarrolladores y usuarios de UNIX (IBM, Sun, etc.). Luego, Novell desarrolló una nueva versión del software llamado UnixWare.
- **XENIX y OpenServer:** a partir del código versión 7, en 1980 la empresa Santa Cruz Operations (SCO/original) desarrolló una versión de UNIX para el chip de Intel, que se llamaría XENIX y luego, en 1995, otra versión llamada SCO OpenServer. Se basa en el UNIX/Intel para chips de bajo rendimiento. Este software incluye varias librerías propietarias denominadas OpenServer Shared Libraries, que se utilizan en las distintas aplicaciones que se ejecutan sobre esta versión de UNIX.
- **AIX:** en 1985, IBM adquirió de AT&T una licencia con varios derechos en el sistema UNIX. En los acuerdos, IBM reconoció los derechos de AT&T sobre el software UNIX y se comprometió a varias obligaciones de confidencialidad y de respeto de derechos de autor. También IBM se comprometió a no transferir el código UNIX a terceros, a restringir el uso de UNIX por terceros y a no permitir sublicencias. IBM desarrolló una versión de UNIX llamada AIX para su uso interno y, posteriormente, para sus clientes. A partir de la versión 4.0, AIX se basa en UNIX System V y se destina a equipos con el chip PowerPC de IBM.

#### Nota

Usamos la nomenclatura SCO/original porque luego los activos y el nombre de la empresa fueron adquiridos por la SCO actual, que anteriormente se llamaba Caldera.

- **BSD/UNIX:** el sistema operativo BSD/UNIX es la versión de UNIX desarrollada dentro de la UC Berkeley, la cual queda disponible y distribuida bajo la licencia abierta BSD, después del litigio con AT&T. La versión BSD/UNIX, que siempre ha sido “código abierto”, es la principal fuente de cualquier elemento de código UNIX en el sistema GNU/Linux.

**Nota**

Hay una historia completa de las versiones de UNIX en:  
<http://www.levenez.com/unix/history.html>

Podéis ver también el “Position Paper” de la OSI sobre el litigio de SCO contra IBM, en:  
[www.opensource.org/sco-vs-ibm.html](http://www.opensource.org/sco-vs-ibm.html)

En 1998, IBM, Intel y SCO/original se asociaron en el Proyecto Monterrey para desarrollar un sistema operativo basado en UNIX para chips Intel de 64 bits. En este proyecto, SCO dio acceso a IBM a varios datos sobre SCO UnixWare. Sin embargo, en mayo de 2001, IBM dejó de participar en el proyecto, con el pretexto de que el proyecto no iba a tener ningún éxito.

## Los derechos en UNIX y sus titulares

Desde el punto de vista legal, hasta los años 1990, AT&T quedó formalmente como titular de los derechos de autor en el sistema y los usuarios debían, teóricamente, obtener una licencia de uso de AT&T. La obtención de esta licencia se consideraba “una formalidad”.

La permisividad de AT&T frente al uso del código sin licencia empezó a mermarse a partir de 1990, cuando crea USL para comercializar el sistema operativo. En particular, en 1993, AT&T entabló un juicio contra UC Berkeley por el uso del código de UNIX en el sistema operativo BSD/UNIX. AT&T perdió el litigio. El tribunal declaró que solamente 3 archivos de BSD/UNIX (entre 18.000) eran claramente de la propiedad de AT&T, los demás podían considerarse de uso y distribución libres. Se sostiene que, con este fracaso jurídico, AT&T también perdió el control de cualquier secreto de negocio y de autoría en los métodos y los materiales originales del sistema operativo UNIX.

A partir de esa fecha, ha habido varios cambios de titular en los derechos de propiedad intelectual sobre las distintas versiones de UNIX:

- En 1992-1993, Novell adquirió de AT&T todos sus derechos en las versiones originales de UNIX (hasta el System V).
- En 1995, SCO/original compró a Novell todos los derechos de UnixWare y los derechos restantes de AT&T sobre el código UNIX. Por lo tanto, desde esta fecha, las versiones originales de UNIX hasta la versión System V además de UnixWare están bajo la titularidad de SCO/original. Como consecuencia, System V, UnixWare y OpenServer eran (y son) tres versiones propietarias de UNIX distribuidas por SCO/original.
- En 2001, cuando SCO/original se divide, una nueva empresa Caldera, formada por ex-desarrolladores de Novell, compró de ella los activos y derechos sobre todos los elementos UNIX de SCO/original (OpenServer, UnixWare y versiones anteriores de UNIX). Luego, en 2002, Caldera cambió su nombre a SCO Group Inc. De aquí en adelante, llamaremos a esta empresa simplemente SCO.

Resulta interesante advertir también que, desde 1992, el término *UNIX* es una marca registrada de The Open Group, una organización creada para definir estándares técnicos y, formalmente, describe cualquier sistema operativo conforme al estándar UNIX publicado (el Single UNIX Specification y el POSIX).

#### 10.7.4. El desarrollo de GNU/Linux

El sistema operativo GNU/Linux es de la familia de UNIX, no porque provenga del código UNIX que hemos comentado, sino porque se modeló sobre el sistema UNIX original y tiene una estructura que cumple aproximadamente con los estándares UNIX de The Open Group. El *kernel* fue creado de manera independiente por Linus Torvald a partir de 1991 y otros elementos (como el GNU C Compiler) fueron agregados más adelante. El “equipo de control” de Linux liderado por Linus Torvald declara que, desde entonces, miles de programadores han aportado líneas de código al *kernel* y al sistema

operativo en general. Desde 1994, cuando se liberó la versión 1.0 junto con elementos del proyecto GNU, GNU/Linux se distribuye bajo la licencia GPL.

**Nota**

Podéis ver Linus Torvald en “Voices” para una descripción más detallada del desarrollo de Linux. Se ha probado que la versión Linux 1.0 cumplía con el estándar Single UNIX Specification, pero actualmente la evolución de GNU/Linux es tan rápida que no se puede decir en un momento determinado si el sistema operativo cumple con este estándar o no. Mantendremos el nombre que le da la Free Software Foundation, dado que el sistema consiste en el *kernel* Linux junto con otros elementos del proyecto GNU.

Una distribución típica de GNU/Linux es una colección de programas que incluyen el sistema operativo, junto con herramientas para instalarlo y la interfaz de usuario, más otras herramientas de administración del sistema. Varias versiones comerciales (Red Hat, Mandrake, SuSe) se distribuyen con aplicaciones de más alto nivel, como programas de correo electrónico, bases de datos, procesadores de texto y planillas de cálculo.

En enero de 2001, se liberó Linux 2.4, la última versión estable de GNU/Linux. Esta versión ha recibido contribuciones de diversas fuentes, desde las versiones anteriores de GNU/Linux (versión 2.2), hasta códigos aportados por SCO/original, Caldera (hoy, SCO), IBM e Intel. En particular, contribuyeron al desarrollo del nuevo *kernel* de Linux los miembros de la división de software libre de IBM.

Es interesante notar que, desde 1994, la empresa que hoy se llama SCO ha apoyado el desarrollo de código libre (o *open source*) y ha distribuido, hasta mayo de 2003, el sistema operativo GNU/Linux y otro software bajo los términos de la GPL, incluido UNIX Versión 7. Esta distribución comprende también la versión GNU/Linux 2.4 bajo litigio. Asimismo, esta empresa, tanto bajo su nuevo nombre SCO como cuando se denominaba Caldera, ha participado en el desarrollo del *kernel* de Linux, incluso en varios módulos mencionados en la demanda contra IBM.

**Ejemplo**

Podéis ver, por ejemplo, MozillaQuest Magazine, The SCOsource IP Matter, Mike Angelo, 05/02/2003 en:

[http://mozillaquest.com/Linux03/SCOsource-01\\_Story01.html](http://mozillaquest.com/Linux03/SCOsource-01_Story01.html)

**Lectura complementaria**

Podéis ver *SCO Establishes SCOsource to License UNIX Intellectual Property* (traducción de los autores):

<http://ir.sco.com/ReleaseDetail.cfm?ReleaseID=99965>

**10.7.5. Los primeros pasos del litigio**

A partir de enero de 2003, SCO hace varias declaraciones públicas que levantan dudas y preocupación en la comunidad de desarrolladores y empresas involucradas con GNU/Linux. Primero, alega que los usuarios de Linux están descargando e instalando elementos de SCO Open Server para ejecutar aplicaciones compatibles con UNIX sobre GNU/Linux.

SCO establece SCOsource para otorgar licencias relativas a la propiedad intelectual en UNIX:

“SCO desarrolla y es titular de SCO UnixWare y SCO OpenServer, ambos programas basados en la tecnología de UNIX System V. SCO es titular de gran parte del núcleo de la propiedad intelectual en UNIX y tiene el derecho de otorgar licencias relativas a esta tecnología y de obligar al cumplimiento de los derechos de autor y de patentes. Varios vendedores de hardware y software piden frecuentemente a SCO el acceso a componentes clave de UNIX. SCO ampliará las actividades relativas a las licencias y ofrecerá a socios y clientes nuevas formas de aprovechar estas tecnologías.”

“La licencia SCO System V para Linux otorgará acceso a las UNIX System Shared Libraries de SCO para su uso con Linux. Los clientes utilizan estas librerías frecuentemente para permitir a aplicaciones de UNIX ejecutarse sobre Linux. En el pasado, las licencias relativas a SCO UnixWare y OpenServer no permitían que dichas librerías se usaran fuera de los sistemas operativos de SCO. A partir de esta nota, los clientes pueden otorgar licencias relativas a estas librerías de SCO para su uso con Linux, sin tener que obtener el sistema operativo entero de SCO. Esto permitirá a los clientes ejecutar miles de aplicaciones de UNIX sobre Linux.”

“Linux es un producto de Open Source y comparte la filosofía, arquitectura y API con UNIX. A partir de [la fecha de publicación], las librerías de SCO estarán disponibles para desarrolladores de aplicaciones, vendedores de sistemas operativos y hardware, proveedores de servicios y usuarios finales. SCO ayudará a sus clientes a combinar legítimamente las tecnologías de Linux y UNIX para ejecutar miles de aplicaciones desarrolladas para UNIX.”

Luego, SCO alega que hay elementos del código de UNIX System V (propiedad de SCO) en el núcleo de GNU/Linux 2.4. Es decir, señala que, en su opinión, el equipo de desarrollo de GNU/Linux ha tomado líneas de código propietario (de SCO) para incorporarlas a la nueva versión de Linux. En marzo, SCO arguye que estas líneas (sin identificarlas) fueron incorporadas por IBM a través de sus diversas actividades con AIX y el Proyecto Monterrey.

Palabras de Chris Sontag, responsable de SCOsource en SCO:

“[IBM] tiene acuerdos con SCO por licencias relativas a la tecnología de UNIX System V: no debe transferir tecnología de AIX [a la Comunidad de Linux], tampoco ninguna obra derivada de AIX. Ni siquiera puede mostrar el código fuente a cualquiera que no tenga una licencia válida otorgada por nosotros para ver el código,” declara C Sontag, responsable de la división SCOsource de SCO. “La comunidad Linux es libre de avanzar con desarrollos independientes. Y creo que esto ha ocurrido de manera justa y razonable. Nosotros alegamos que una parte de lo que IBM ha aportado a la comunidad Linux era ilegítimo y en violación del contrato que tienen con nosotros.”

#### 10.7.6. El litigio entre SCO e IBM

Todo apuntaba a una demanda por violación de derechos de autor contra IBM (y, accesoriamente, contra la comunidad de desarrollo de GNU/Linux), por diversas infracciones de derechos exclusivos de explotación de varias líneas y módulos de código de propiedad de SCO.

#### La demanda de SCO

Sin embargo, cuando se inicia el litigio el 7 de marzo de 2003, SCO no fundamenta su demanda tanto en violaciones de derechos de autor, como en la violación de varios pactos contractuales:

- La violación de varios compromisos contenidos en el contrato de licencia ATT/IBM de 1985/96 y con Sequent en 1986 (una empresa comprada por IBM): IBM ha efectuado la transmisión y la modificación de código UNIX, por ejemplo, permitiendo su

#### Nota

Palabras de Chris Sontag, responsable de SCOsource en SCO, recogidas por Doc Searls, 7 de marzo de 2003:

<http://www.linuxjournal.com/article.php?sid=6706>

#### Nota

La modificación del pleito del 16 de junio de 2003 concentra los alegatos en violaciones contractuales. Texto integral en las direcciones:

<http://www.sco.com/scosource/complaint3.06.03.html>  
<http://www.sco.com/ibmlawsuit/amendedcomplaintjune16.html>

incorporación en el kernel de GNU/Linux. Además, IBM ha otorgado sub-licencias a terceros y ha divulgado secretos de negocio de SCO a terceros.

- La violación de secretos de negocio (en particular en relación con Open Server, librerías Open Server y UnixWare), protegidos por cláusulas de confidencialidad contenidas en contratos entre ATT/IBM y SCO de 1985 y 1996; y también obtenidos como resultado del proyecto Monterrey.
- Competencia desleal, en el uso de métodos, código, materiales y datos confidenciales referentes a UNIX y propiedad de SCO, en distribuciones de Linux.
- La interferencia en contratos con clientes de SCO, por inducirles a modificar y usar el software UNIX de SCO en maneras contrarias a las obligaciones de respeto de derecho de autor incurridas por dichos clientes.

Luego, en junio, SCO pretende revocar la licencia AIX de IBM, exigiendo que deje de usar y devuelva cualquier parte del código de UNIX System V y exige también que IBM y otras grandes empresas usuarias de Linux obtengan una licencia de uso de SCO referente a elementos del sistema operativo GNU/Linux 2.4 (SCO Revokes I.B.M.'s License for Operating System Software, S. Lohr, New York Times, 17 de junio, 2003).

#### Nota

"Los acuerdos de 1985/1996 y de sub-licencia de UNIX contienen cláusulas claras para la protección de código fuente (de SCO), la restricción de su modificación y la creación de derivados y de la divulgación de métodos de trabajo," declara Mark Heise, abogado de SCO.

Boise, Schiller and Flexner, LLP, 16 junio de 2003:

"Al contribuir código de AIX a Linux y usar métodos de UNIX para acelerar y mejorar la operación de Linux



y, por consiguiente, destruyendo a UNIX, IBM demuestra claramente su abuso del código fuente de UNIX y que ha violado sus compromisos con SCO. [...] Hoy, AIX es un derivado no autorizado del código de UNIX System V, y cualquier usuario de AIX está ejecutándolo sin permiso.”

En agosto, SCO revocó la licencia de Sequent, una filial de IBM, relativa a código de UNIX utilizado en otra versión de UNIX distribuida por IBM, llamada Dynix/ptx.

### La defensa y contra-demanda de IBM

En agosto de 2003, IBM contesta la demanda de SCO. En su defensa, IBM niega gran parte de las acusaciones de SCO y alega que:

- No ha violado ningún secreto comercial ni relevado ninguna información confidencial de SCO a terceros.
- La licencia AIX relativa a UNIX es irrevocable y perpetua (lo que fue confirmado por Novell, la empresa que inicialmente otorgó la licencia), lo que permite a IBM seguir con la distribución sin restricciones de UNIX/AIX.
- No ha contribuido ningún elemento de código propiedad de SCO al código de Linux 2.4
- SCO ha violado la licencia GNU GPL referente a GNU/Linux, bajo la cual SCO realizó y aceptó contribuciones para el sistema GNU/Linux y realizó sus propias distribuciones del mismo bajo esta licencia.
- En sus versiones comerciales de UNIX (UnixWare, Open Server, SCO Manager y Reliant HA), SCO ha violado directamente cuatro patentes de IBM referentes a la compresión de datos, la navegación entre programas y menús, y métodos de verificación de la recepción de mensajes electrónicos y de monitoreo de sistemas organizados en *clusters*.



“Por la distribución de productos bajo la licencia GNU/ GPL (por ejemplo, incluido en sus distribuciones de Linux), SCO se ha comprometido a no atribuirse ciertos derechos de autor como los de reclamar regalías o derechos de licencia sobre cualquier elemento de código distribuido bajo los términos de dicha licencia. Tampoco puede restringir distribuciones posteriores de código fuente divulgado por SCO bajo la licencia GPL,” alega IBM.

#### 10.7.7. Los alegatos de SCO contra la comunidad Linux

Otro aspecto del conflicto es la campaña de SCO contra la distribución libre de Linux 2.4 y algunas librerías asociadas. Inicialmente, como hemos mencionado, desde enero de 2003 SCO ha alegado que muchos usuarios de Linux usaban unas librerías UNIX OpenServer de SCO, para ejecutar aplicaciones UNIX sobre Linux. Luego, de marzo a mayo, los directores y empleados de SCO hicieron varias declaraciones públicas alegando que la versión 2.4 de Linux contiene código copiado de UNIX en violación de derechos de propiedad intelectual de SCO.

##### Ejemplo

Por ejemplo, el comunicado de prensa de SCO fechado el 22 enero de 2000 (SCO Establishes SCOsource to License UNIX Intellectual Property) o la carta de SCO a sus clientes del 12 de mayo de 2003 (parcialmente disponible en: <http://www.lemis.com/grog/SCO/index.html>).

El 12 de mayo de 2003, SCO envía una carta a 1.500 clientes que estima que son usuarios de Linux. Alega que: “Linux, de manera sustancial, es una obra derivada no autorizada de UNIX”. Sin ofrecer ninguna prueba (la mayor queja de la comunidad Linux contra SCO), sigue:

“Muchos contribuidores a Linux eran originalmente programadores de UNIX, que tenían acceso al código fuente de UNIX distribuido por

AT&T y estaban obligados por acuerdos de confidencialidad, incluso relativo a los métodos y conceptos involucrados en el diseño de software. Tenemos pruebas de que partes de UNIX System V han sido copiadas en Linux y de que otras partes de UNIX System V han sido modificadas e incorporadas a Linux, aparentemente para ocultar la fuente original del código.”

La carta y los alegatos de SCO han levantado una ola de denuncias, comentarios, burlas y hasta ira en la comunidad de los desarrolladores de Linux.



#### LinuxTag c. SCO en Alemania

SCO ha tenido que retirar esta carta de su página web como consecuencia de la acción legal de LinuxTag en Alemania. Bajo el derecho alemán, cualquiera que haga una declaración pública que pueda dañar el negocio de un tercero debe aportar pruebas de lo afirmado o, si no, retirar las declaraciones. El 23 de mayo de 2003, los abogados de LinuxTag e.V., una asociación que representa a los desarrolladores de Linux en Alemania, denunció las declaraciones de SCO y pidió su retiro o la publicación de las pruebas relevantes antes del 30 de mayo. “Con sus declaraciones sin fundamentos, SCO está dañando a los competidores, intimidando a los clientes de Linux y perjudicando la reputación de Linux como plataforma abierta”, declaró el portavoz de LinuxTag. Cuando SCO no publicó ninguna prueba, el tribunal alemán, en dos decisiones del 28 de mayo y del 5 de junio, obligó a cerrar el sitio web alemán de SCO hasta el retiro de la carta.

Para más detalles, leer *SCO hit by legal action in Germany*, John Blau, 4 junio de 2003 en:

[http://www.infoworld.com/article/03/06/04/HNscosite\\_1.html](http://www.infoworld.com/article/03/06/04/HNscosite_1.html)

Podéis ver también *Rechtswidrige Behauptungen von SCO - Ordnungsverfahren gegen SCO eingeleitet*

<http://www.linuxtag.org/2003/de/press/releases.xsp?id=4>

LinuxTag ha iniciado otra demanda contra SCO, por haber mostrado a sus socios comerciales un hipervínculo a la misma carta después de las decisiones del tribunal. El tribunal alemán impuso una multa de 10.000 euros por no haber suspendido sus alegaciones contra Linux sin fundarlas, una decisión que SCO está apelando. Actualmente, los enlaces de SCO van al sitio [www.sco.com.de](http://www.sco.com.de), que no tiene ninguna alegación prohibida.

Sin embargo, estas declaraciones de SCO han sido difíciles de averiguar, dado que la empresa exige la firma de un contrato de confidencialidad a todos aquellos que quieran ver las similitudes entre los códigos de SCO y de Linux 2.4. Desde el punto de vista de la comunidad Linux, sus portavoces (como B. Perens, E. Raymond y L. Torvald) siguen pidiendo que SCO indique cuáles son los archivos y líneas de código en cuestión. Aceptan que puede haberse integrado código propietario de SCO en los módulos de Linux, debido al propio sistema de desarrollo abierto. Sin embargo, no habrá sido a propósito y declaran que inmediatamente eliminarían cualquier código que consideren que pudiera ser de la propiedad de SCO, para reemplazarlo con código nuevo abierto.

#### Ejemplo

Podéis ver, por ejemplo, la carta abierta y la de respuesta de E. Raymond y B. Perens "An Open Letter to Darl McBride" y "Response to Darl McBride's Open Setter" en las direcciones:

- <http://www.catb.org/~esr/writings/mcbride.html>
- <http://www.catb.org/~esr/writings/mcbride2.html>

En julio de 2004, para sostener sus demandas legales a nivel federal, SCO registró derechos de autor en UNIX System V en el registro federal de derechos de autor de Estados Unidos y ofreció licencias de uso de UnixWare a usuarios de Linux 2.4.

Siguiendo la línea establecida en la carta de mayo a los 1.500 clientes, declarando su intención de exigir licencias de los usuarios

#### Nota

SCO raps Red Hat, sets license prices, D Becker, news.com:

[http://news.com.com/2100-1001\\_3-5060134.html](http://news.com.com/2100-1001_3-5060134.html).

de Linux, en agosto, SCO determinó que los derechos de licencia de Linux serían USD 699 por empresa. Además, SCO pidió a varias empresas OEM que venden equipos informáticos de varios tipos (ordenadores y dispositivos electrónicos como el Playstation, descodificadores de vídeo, etc.) y que tienen GNU/Linux instalado o integrado como sistema operativo, derechos de licencias de USD 32 por equipo o dispositivo.

A mitad de agosto de 2003, SCO mostró en una conferencia algunas líneas de código de Linux que alegaba habían sido copiadas de código UNIX de SCO. Aunque parecían copiadas de UNIX System V, un análisis rápido de B. Perens y otros programadores de la comunidad Linux ha indicado que este código hubiera podido venir de varias fuentes abiertas, como el BSD/UNIX o código de la misma SCO distribuido bajo licencia abierta GPL. A pesar de estos argumentos, SCO declara que la generalidad de sus alegaciones contra IBM y Linux 2.4 no se ve afectada.

Finalmente, en septiembre de 2003, SCO decide facturar directamente a los usuarios comerciales de Linux.



### Críticas al sistema de desarrollo

En una “carta abierta” en septiembre de 2003, D. McBride, director de SCO, ataca el sistema de desarrollo abierto, personificado en el desarrollo de Linux:

“La contribución ilegítima de código de UNIX por SGI a Linux es un ejemplo que revela los defectos estructurales en el proceso de desarrollo de Linux. De hecho, este tema toca al corazón de la cuestión de si código abierto puede ser un modelo de desarrollo confiable para software para las empresas. La propiedad intelectual fundamental de Linux es evidentemente fallida al nivel sistémico, bajo el modelo actual. Hasta la fecha, hemos declarado que más de un millón de líneas de código protegido del UNIX System V han sido aportados a Linux a través de este modelo.”

### Nota

SCO's Evidence: This Smoking Gun Fizzles Out, de E Raymond, en:

<http://www.catb.org/~esr/writings/smoking-fizzle.html>.

Analysis of Linux Code that SCO Alleges Is In Violation Of Their Copyright and Trade Secrets, de B. Perens en:

<http://www.perens.com/SCO/SCOCopiedCode.html>

### Nota

SCO To Invoice Linux Users, NewsFactor Network, 3 Septiembre, 2003 en:

<http://www.newsfactor.com/perl/story/22208.html>

**Nota**

District Court of Delaware, USA, el 4 de agosto de 2003. Podéis ver también “Red Hat files suit against SCO”, CNET News.com, Stephen Shankland, 04/08/2003.

**Red Hat y sistemas integrados Linux**

Red Hat, Inc. es una de las empresas líderes de la llamada “comunidad Linux”, distribuyendo y realizando la instalación y soporte de GNU/Linux para sus clientes. Para defenderse contra las alegaciones de SCO (de que la versión 2.4 de GNU/Linux tiene líneas de código de SCO incorporadas en el sistema operativo sin su permiso), Red Hat interpone una querrela contra SCO por publicidad engañosa y competencia desleal. Argumenta que las declaraciones de SCO relativas a Linux son engañosas y falsas y están afectando el negocio de Red Hat. Asimismo, pide que el tribunal determine que la versión 2.4 de Linux no infringe ningún derecho de propiedad intelectual de SCO.

Los alegatos de Red Hat son los siguientes:

- Una denuncia bajo el *Lanham Act* (derecho federal sobre la propiedad intelectual, sobre todo relativa a las marcas registradas) relativa a la publicidad engañosa y competencia desleal, sobre la base de que las declaraciones de SCO son engañosas y falsas.
- Demandas bajo derecho estatal por competencia desleal, difamación comercial (comentarios adversos contra la marca registrada de Red Hat) por declaraciones falsas e injustas.
- Interferencia ilícita e intencional en los negocios de Red Hat.
- Red Hat pide una declaración del tribunal que confirme que Red Hat no viola ningún derecho de propiedad intelectual de SCO.

Asimismo, reclama una prohibición provisional que impida que SCO continúe sus alegaciones de que Linux infringe derechos de propiedad intelectual de SCO.



“Iniciamos esta denuncia para impedir que SCO haga declaraciones sin fundamentos y falsas, atacando a Red Hat y la integridad del proceso de desarrollo del movimiento Open Source”, dice el abogado de Red Hat.

“Este proceso colaborativo de desarrollo, que creó el sistema operativo Linux, ha sido puesto en duda y amenazado.”

*Red Hat Takes Aim at Infringement Claims*, Declaración de prensa Red Hat, 4 de agosto de 2003, en:  
[www.redhat.com/about/presscenter/2003/press\\_sco.html](http://www.redhat.com/about/presscenter/2003/press_sco.html)

Esta demanda ha sido contestada por SCO en septiembre de 2003, para que se desestime el expediente sobre la base de que SCO no ha iniciado ninguna acción contra Red Hat.



#### **La perspectiva de otros actores del sector: Microsoft, Sun y HP**

En mayo, Microsoft declara que ha firmado un acuerdo de licencia con SCO en relación con varias patentes y código fuente de SCO:

“El acuerdo es representativo del compromiso de Microsoft relativo al respeto de la propiedad intelectual ajena y el intercambio legítimo de código dentro de la comunidad de TI. Permite garantizar la legalidad de las soluciones de Microsoft y apoya nuestros esfuerzos sobre productos y servicios UNIX.”

Brad Smith, asesor legal de Microsoft. *Microsoft to license UNIX code*, CNET News.com, May 18, 2003, en:  
[http://news.com.com/2100-1016\\_3-1007528.html](http://news.com.com/2100-1016_3-1007528.html)

Aunque se puede interpretar como un apoyo a la acción de SCO, dándole crédito, algunos comentaristas de la comunidad Linux acusan a Microsoft de fomentar el litigio entre SCO e IBM y la comunidad Linux, para desprestigiar el sistema operativo Linux que es el competidor más peligroso para sus propios productos

Windows (para más detalles, podéis ver los “Halloween documents” en [www.opensource.org/halloween/](http://www.opensource.org/halloween/)).

“Es una maniobra para aumentar la incertidumbre que SCO ha creado acerca de Linux”, declara B. Perens. “El acuerdo confirma mis sospechas de que el gigante del software es una de las fuerzas detrás de las acciones legales de SCO.”

*Microsoft sends message with UNIX deal*, CNET News.com, 19 de mayo de 2003, en: <http://news.com.com/2100-1016-1007715.html>

Luego, en julio, Microsoft amplía las indemnizaciones contenidas en sus acuerdos de licencia relativas a cualquier litigio de propiedad intelectual sobre los productos Microsoft. Anteriormente, la indemnización estaba limitada al precio de compra del producto (Windows, por ejemplo). A partir de marzo, la cláusula de compensación se extiende a favor de los clientes y, además, se mejoran los plazos de garantías de 90 días a un año.

## Indemnizaciones

Aunque SCO declare que los productos de Sun no están afectados por sus demandas relativas a Linux (Sun tiene acuerdos con SCO sobre el uso de código de UNIX en el sistema operativo Solaris), en septiembre de 2003 Sun anuncia que está considerando agregar una disposición a algunas de las licencias Java para proteger a los licenciarios contra las demandas formuladas por SCO (solamente para usuarios de *Java 2 Micro Edition*, para dispositivos pequeños como los teléfonos móviles). Pero no ofrece esta garantía para usuarios de Linux, ya que Sun prefiere incentivar las ventas de su propio sistema operativo Solaris.

En septiembre de 2003, Hewlett-Packard Co ofrece una indemnización a sus clientes (usuarios de Linux en equipos de HP, con contrato de servicio con HP) en caso de litigio con SCO.

### Nota

Ver “Update: HP to indemnify its corporate Linux users against SCO”, Computerworld, del 24 de septiembre de 2003 en:

<http://www.computerworld.com/softwaretopics/os/linux/story/0,10801,85288,00.html>



Dell, un patrocinador activo de Linux, se ha negado a compensar a sus clientes en relación con cualquier problema legal relativo al uso de Linux en sus equipos.

### 10.7.8. Otras consideraciones y conclusiones

No se puede hablar de conclusiones, dado que el litigio está todavía pendiente de juicio y resolución. Como consecuencia práctica de las alegaciones y demandas de SCO, muchas empresas grandes han mencionado dudas sobre la implementación de Linux hasta que se resuelvan las querellas. Una empresa de la lista Fortune 500 se ha “registrado” con SCO y comprado una licencia para el uso de GNU/Linux 2.4.

Por otro lado, la comunidad de desarrolladores de software libre acusa SCO de iniciar una operación para sembrar “temor, incertidumbre y dudas” (FUD en inglés) entre los usuarios de Linux, apoyado por Microsoft. Varios actores y empresas del sector se han pronunciado en contra, incluso Richard Stallman, Bruce Perens (discurso en Linuxworld, 5 de agosto de 2003), Eric Raymond (fundador del Open Source Initiative), Eben Moglen (profesor de Derecho en la Universidad de Columbia, Nueva York, y abogado de la Free Software Foundation) y el Open Source Development Lab (presentación al Consejo Asesor de Open Source Development Lab, Nueva York, 24 de julio de 2003, y “position paper” de 31 de julio de 2003 –<http://www.osdl.org>) y muchos otros programadores que han participado en el desarrollo de GNU/Linux (<http://www.linuxworld.com/>). Por lo tanto, la gran mayoría de desarrolladores y usuarios de Linux siguen usándolo sin hacer caso a las demandas de SCO.

Esto cambiaría enormemente si SCO saliera ganando cualquiera de los litigios. Las cuestiones legales quedan, por lo tanto, abiertas.

Desde diciembre de 2003, la situación ha evolucionado:

- Un tribunal americano ha exigido que SCO presente pruebas de lo alegado. SCO apenas ha cumplido, lo que llevó a IBM a exigir otra vez que desista en el juicio.

#### Nota

FUD, acrónimo en inglés para *Fear, Uncertainty and Doubt*. Esta expresión fue utilizada por la Comunidad Linux contra las declaraciones de Microsoft relativas al sistema operativo Linux, en los Halloween Documents, dónde pone en duda la validez, calidad y operatividad de Linux.

#### Nota

*Linux community scoffs at SCO's evidence*, CNET News.com, 2 de agosto de 2003, en:  
<http://news.com.com/2100-1016-5066410.html>

- D. McBride ha escrito varias cartas abiertas, defendiendo la política de su empresa y atacando el software libre como anticonstitucional.
- SCO ha demandado Novell en relación con derechos de autor sobre su versión de UNIX.

Sin embargo, no agregamos más aquí, ya que esta evolución será tema de debate durante el curso y una actividad de los estudiantes.

### 10.7.9. Preguntas y actividades

El caso SCO es sumamente complicado y no podemos, en este espacio, hacer más que bosquejar los argumentos principales de los protagonistas. Sin embargo, podemos aprovechar el caso para ilustrar y discutir casi todos los temas estudiados en el curso.

#### Análisis de la situación

- 1) ¿Cuáles son los derechos asociados con los diferentes programas en cuestión? ¿Quién es titular de estos derechos? (Haced una tabla, si preferís, asociando las versiones de código y sus titulares.)
- 2) ¿Cuáles son los escenarios legales posibles en el asunto? (SCO tiene razón, IBM tiene razón, etc.)
- 3) ¿Cuáles podrían ser los motivos de las distintas posiciones en este conflicto?
- 4) ¿Qué efecto (sobre la posición de SCO y el uso de GNU/Linux en general) tiene el uso de la licencia GPL para el *kernel* y otros elementos de GNU/Linux?
- 5) ¿Cuál es el estado actual de este asunto?

#### Análisis de las demandas

##### IBM

- 6) ¿Por qué la demanda inicial de SCO se basa en violación de contrato y no se basa en derechos de autor?

- 7) ¿Qué tendría que probar SCO para sostener los diferentes argumentos de su demanda contra IBM?
- 8) IBM alega que SCO está violando la licencia GPL relativa a Linux. ¿Cuáles son las bases de este alegato? ¿Estáis de acuerdo? ¿A qué se arriesga IBM, involucrando la licencia GPL en un asunto que *a priori* no trataba de esta licencia?
- 9) ¿Qué pensáis del uso por IBM del derecho de las patentes para defenderse y atacar a SCO? ¿Qué implicaciones podría tener para el movimiento de software libre en general y la comunidad GNU/Linux en particular?

### Red Hat

- 10) ¿En qué argumentos funda Red Hat su demanda?
- 11) ¿Cuáles son los efectos, las ventajas y los riesgos de la denuncia de Red Hat contra SCO?

### Análisis de las consecuencias

- 12) ¿Qué implicaciones o lecciones tiene este asunto para el desarrollo de software libre?
- 13) ¿Qué riesgos tiene una empresa u organización que:
  - distribuye GNU/Linux como parte de un servicio comercial?
  - instala o usa GNU/Linux, ahora o en el futuro?
- 14) ¿Qué estrategia podría seguir para minimizar estos riesgos?
- 15) ¿Cuál es el “peor escenario posible” si SCO tiene razón? Para desarrolladores y usuarios?
- 16) ¿Cuáles son las consecuencias para SCO si hay elementos de Linux incorporados en el código de SCO (OpenServer o UnixWare)?



## GNU Free Documentation License

GNU Free Documentation License  
Version 1.2, November 2002

Copyright (C) 2000,2001,2002 Free Software Foundation, Inc.  
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA  
Everyone is permitted to copy and distribute verbatim copies  
of this license document, but changing it is not allowed.

### 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

## 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent.

An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition.

Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.



If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material.

If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

#### 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number.

Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit.

When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form.

Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the

translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

### ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.







