Efectos de las Patentes de Software

Jens Hardings (jhp@csol.org)
Centro de Software Libre (CSoL, http://www.csol.org/)

Resumen

El tema de patentes en general y sobre todo de patentes de software en particular es sumamente controvertido y es discutido hoy en día en muchas partes del mundo. Convenciones internacionales, tratados bilaterales y otras instancias están determinando qué modelos y tendencias se seguirán a futuro en estos temas, y en las negociaciones priman fuerza política, intereses económicos a corto plazo e intereses de actuales actores importantes en el ámbito económico y político.

1. Introducción

Por lo general, en el contexto de desarrollo de Software Libre se le ha dado mucha importancia al código[1], que es al fin y al cabo lo que se busca crear. Un proyecto se realiza en general porque alguien tiene una necesidad específica y no hay un proyecto existente que esté a la altura de lo que se busca. Hoy en día, el mundo empresarial se ha involucrado cada vez más en el mundo del Software Libre. Primero simplemente usaban el resultado de los proyectos, y luego comenzaron a participar directamente en la extensión, control de calidad, difusión e incluso creación de proyectos de Software Libre.

El éxito que ha tenido el Software Libre, a pesar de no ser la meta ni existir ninguna iniciativa centralizada con alguna meta, lo ha llevado a ampliar su área de acción. Esto incluye no solo el interés por parte de la industria del software que ha quedado clara, sino también el atractivo de este modelo de desarrollo para personas no directamente ligadas con la computación, los cuales han ayudado en

[1] "Show me the code" es una frase atribuida a Linus Torvalds, creador del kernel Linux y que describe muy bien la actitud de atacar problemas reales con soluciones reales que existe en muchas partes de la comunidad.

proyectos de documentación de los programas, traducción con el fin de internacionalizar proyectos existentes, y un sinnúmero de otras actividades relacionadas con difusión y propagación de las ideas detrás de este movimiento.

La propiedad intelectual juega un rol muy importante en el fenómeno del software libre. La primera institución creada que se preocupa explícitamente del tema, la FSF[2] o Free Software Foundation, fue creada precisamente para poner límites al celo exagerado de las grandes empresas que se negaban a compartir la información que permitiría llevar a cabo proyectos interesantes.

En cuanto a los conceptos y las tradiciones legales en torno a propiedad intelectual, se ha vaticinado (cf. Moglen (1999)) que el Software Libre y su forma funcionar tarde o temprano revolucionarán el concepto de propiedad intelectual.

2. Generación de políticas de patentes

Debido a la gran complejidad de los sistemas de patentes, es usual escuchar los argumentos de que es necesario que los abogados especialistas en patentes manejen las políticas al respecto. Esto es precisamente lo que ocurre en organismos como la WIPO (OMPI) [3] o la WTC (OMC) [4]. Los países son muchas veces representados por las oficinas de patentes, abogados especialistas en patentes y otros que no sólo tienen una visión sesgada de cómo se aplican patentes en el mundo real, sino que además tienen un gran interés en que el sistema de patentes de desarrolle de cierta manera (Macdonald, 2003).

Los ingresos de las oficinas de patentes tienen estrecha relación con la cantidad de patentes otorgadas, lo mismo sucede con los abogados de patentes. Incluso, al existir mayor número de patentes, los litigios y las negiciaciones que requieren de abogados especialistas aumenta, con lo cual esa profesión recibe beneficios directos. Es, por lo tanto, un incentivo perverso el permitir que los que más lucran con estos sistemas sean además los mismos que determinen el camino a seguir a futuro.

A modo de ejemplo, es interesante destacar el rol que cumple la oficina de patentes de EEUU, USPTO, en negociaciones tan lejanas como las internas del parlamento europeo. En estas negociaciones, el USPTO ha insistido fuertemen-

[2] http://www.fsf.org/

[3] WIPO es un organismo dependiente de las Naciones Unidas, encargado de la administración del Tratado de Berna, entre otros

[4] WTC es el organismo que maneja el “Trade Related aspects of Intellectual Property rights” (TRIPs o ADPIC).

te en evitar consideraciones de interoperabilidad entre software para no dañar la fuerza de las patentes, a pesar de estar incentivando graves problemas para la industria del software en general con ello. Esto es aún más paradójico si se considera que la constitución de EEUU es la que explícitamente condiciona la existencia de un sistema de patentes solamente a que ello permita un incentivo real al desarrollo de la ciencia y el conocimiento. Por su esquema de jurisprudencia, en EEUU desde 1981 es posible patentar "todo bajo el sol hecho por el hombre"[5].

Sin embargo, lejos de existir evidencia del tan bullado argumento del incentivo a la producción y al desarrollo en ciencias e investigación, existen numerosos estudios que revelan serios problemas con el sistema de patentes en general, y en relación a patentes de software en particular (Mandeville and Bishop, 1982; Boldrin and Levine, 2001; Merges and Nelson, 1990).

Si bien el artículo 27 de TRIPS obliga a los países miembros a permitir patentes sobre cualquier invención, ya sean productos o procesos, en todos los campos de la tecnología, se pueden exigir requisitos como novedad, paso inventivo y capacidad de aplicación industrial. No es claro según esta definición si el software debe ser considerado como patentable, puesto que se puede considerar que no es un çampo técnico". En Europa se está considerando la posbilidad de exigir que se debe hacer uso controlado de las fuerzas de la naturalez para controlar efectos físicos, y recién así un invento será digno de ser considerado parte de un campo de la tecnología y por lo tanto patentable.

Esto no implica necesariamente que nada que incluya software pueda ser patentado, por ejemplo el concepto de *computer-implemented inventions*, que se refiere a un sistema que usa programas para funcionar pero cumple con esta restricción de ejercer control sobre fuerzas de la naturaleza, puede ser patentado como un todo. Sin embargo, invenciones implementadas con computadores abren un espacio limítrofe en el que se podría argumentar que un programa que de alguna manera causa un efecto visual en una pantalla debiera ser patentable, con lo cual cualquier software sería susceptible de ser patentado, siempre y cuando la formulación de la patente incluya algún aspecto artificial de efectos físicos.

[5]La frase original "everything under the sun made by man" está incluida en la decisión que respalda la extensión de la patentabilidad a muchas áreas hasta entonces consideradas no patentables.

3. Estandarización y Patentes

La inclusión de métodos patentados en estándares es un problema en sí. Incluso se han dado las prácticas por parte de empresas de pasar por todo el proceso de estandarizar un formato o protocolo, para, una vez aceptado el standard y estuviera en proceso de masificación, hacer públicas las patentes sobre esos métodos, sorprendiendo a muchos usuarios de estas tecnologías.

Es por ello que muchas de las instituciones que promulgan estándares hoy en día exigen divulgar la existencia de patentes sobre las propuestas al momento de presentarlas, o de lo contrario perder el derecho de utilizar las patentes después.

En una larga discusión en el "World Wide Web Consortium (W3C)" sobre las políticas respecto de patentes en su proceso de estandarización, se consideró permitir elementos patentados en un standard, siempre y cuando existiera un esquemade licenciamiento razonable y no discriminatorio ("Reasonable And Non-Discriminating", RAND). Esto significa, acceso a las licencias para todos bajo las mismas condiciones, a un precio razonable. Finalmente, después de una decisiva oposición de elementos clave en el movimiento del software libre, se optó por modificar la exigencia de términos RAND por términos que permitan una una implementación sin la necesidad de pagos de ningún tipo, denominada "Royalty Free" (RF). Es posible ver la política de patentes en la estandarización por parte del W3C en http://www.w3.org/Consortium/Patent-Policy-20030520.html [6].

4. Problemas con Patentes de Software

Si bien es cierto que las patentes en general tienden a incentivar el desarrollo tecnológico por la vía de permitir al que invierte gozar de beneficios monopólicos por un tiempo determinado, existe común acuerdo de que en el caso de software las patentes tienden a desincentivar el desarrollo y estancan la industria. Tanto así, que incluso los más acérrimos defensores de hasta los más extremos mecanismos para proteger la propiedad intelectual, la Business Software Alliance (BSA) ha expresado que está contra el principio seguido en EEUU[7], donde se permiten pa-

[6] Última versión disponible el 2 de Noviembre de 2003

[7] Extracto de un artículo de la revista InfoWorld: http://www.infoworld.com/article/03/06/18/EUheaded for limited software patent law (IDGNS)_1.html : *"We support the move towards a harmonized European approach to software patents," said Francisco Mingorance, director of public policy, Europe for the BSA, adding that the BSA doesn't want a U.S.-style system. "We agree that technical effects should be a condition of patentability," he said.* Por otro lado, hay quienes

tentes no sólo de software, sino que incluso de ideas y procesos de negocios. Más información se puede obtener por ejemplo en http://lpf.ai.mit.edu/Patents/.

La razón de estas posiciones contrarias a permitir patentes de software se debe a que existe evidencia de que, contrariamente a lo argumentado por defensores de la aceptación de patentes de software, la industria no se beneficia, como se pudo comprobar en EEUU en la década de los ’80 (Bessen and Maskin, 2000), ver también la argumentación económica de Boldrin and Levine (2001). En general, no solamente en la industria del software, existe la percepción de que la mayor utilidad de patentes es para bloquear competencia y para defenderse frente a demandas de terceros por patentes (Hall, 2003).

Entre los argumentos en contra de las patentes de software se cuentan:

- La cantidad de patentes necesarias para producir un solo producto pueden ser del orden de miles. En cambio, en otras áreas de desarrollo tecnológico, donde se justifica la práctica de patentabilidad, suele ser bastante limitada la cantidad de patentes por producto comercializable. Es por ejemplo el caso de la industria farmacéutica, donde para una droga o proceso generalmente es aplicable una sola patente (c.f. Scotchmer, 1991).

  Incluso si se pudiera hacer un estudio acabado de la existencia de potenciales problemas con patentes de software para un producto, es imposible garantizar que no sea necesario licenciar alguna patente. Por un lado, es difícil realizar búsquedas extensas y exhaustivas por la cantidad de potenciales patentes involucradas, y por otro lado es posible que después de terminado un producto se otorgue una patente sobre alguno de los métodos usados, si otra persona o empresa ha desarrollado ese método antes y solicita la patente. Este simple hecho causa una incertidumbre que no existe con otras formas de protección de bienes inmateriales como los derechos de autor. En la literatura al respecto se habla de campos minados en los cuales se mueve el desarrollo del software. Un ejemplo de esto es lo ocurrido con *Unix compress*. El programa fue creado en 1984, y en 1985 se otorgó una patente sobre el algoritmo LZW. Con esto, el *Unix compress* se volvió ilegal de un día a otro hasta no existir una licencia que permitiera su uso.

- La publicación de la patente debe incluir una descripción que permita a alguien entendido en el área respectiva reproducir el proceso o invención. Sin embargo, al no incluirse el código fuente (ver Hardings and Fuentes

aseguran que la condición de *technical effects* no impide patentes de software, solamente obliga a usar más creatividad en describirlas.

(2003)), para reproducir un procedimiento es necesario volver a desarrollar el código fuente, con lo cual el aporte de una patente es en el mejor caso dudoso. Por ejemplo, para el caso de invenciones es necesario proveer planos y diagramas que explican claramente cómo funciona un elemento dentro del sistema que se pretende patentar. En el caso de software, solamente se agregan descripciones vagas de qué es lo que se pretende patentar, y es necesario desarrollar el producto para tener un programa concreto y usable.

También es importante notar que no es necesario que la innovación que se pretende patentar realmente exista, sino que basta que se haya descrito. En el caso de patentes industriales que se refieren a procesos físicos esto no es un problema, puesto que la descripción es lo suficientemente detallada para implementar el invento, de lo contrario la patente no tiene valor porque el proceso no es aplicable industrialmente. Pero en la industria del software, donde para una patente tampoco se requiere una implementación funcional, el texto de la patente puede ser lo suficientemente vago como para ser aplicado a otras ideas o implementaciones que las propuestas por los inventores. En tal caso, el legítimo inventor va a quedar excluido de utilizar y recibir los beneficios económicos que le corresponderían por su invento, ya que la patente ha sido entregada a quien no ha implementado la idea.

- La inversión requerida para crear un software es mínima comparada con la investigación en otros campos como la medicina, mecánica y otras áreas. Para tomar nuevamente el ejemplo de la industria farmacéutica, desarrollar un fármaco generalmente toma años de trabajo, requiere del uso de equipamiento costoso y muchos estudios igualmente costosos. Por otro lado, la imitación en campos como la industria farmacéutica es trivial comparado con la industria del software, en el cual la imitación implica el desarrollo completo del sistema por estar protegido por derechos de autor y por la no disponibilidad del código fuente en general (Hall, 2003).

  El mismo hecho de que no exista disponibilidad del código fuente hace que las ideas de implementación de software tengan una suficiente protección con el mecanismo de propiedad intelectual conocido como secretos industriales (Koboldt, 2003). Así, si una idea es lo suficientemente trivial de ser implementada sin requerir código fuente ni la publicación de las ideas, probablemente nunca debió haber recibido una patente y en el caso del secreto industrial, puede ser usada libremente si alguien logra desarrollar la misma idea en forma independiente.

En el caso de las patentes de software, detectar si un producto contiene código que está sujeto a patentes por lo general tiene un costo muchas veces superior a la creación del código afectado. Esto se debe por un lado a la gran cantidad de patentes que existen en los países donde son aceptables las patentes de software, además del lenguaje en el que están descritos los procesos protegidos, los cuales se alejan mucho del lenguaje tradicionalmente usado por los programadores y requiere largas interacciones entre abogados y programadores para que ambas partes comprendan qué es realmente lo que está protegido por una patente dada (Mandeville, 1982). Por lo tanto, la práctica de usar una base de datos de patentes como medio de información dentro del proceso de desarrollo de un producto relativo a la informática no sólo no existe, sino que no tiene sentido de ser instaurada por la ineficiencia inherente en el sistema.

- Para PYMES es muy difícil crear una cartera de patentes con los cuales competir en una industria dominada por unos pocos protagonistas que poseen derechos sobre miles de patentes cada uno. Contrariamente a la intención de un sistema de patentes, que pretende proteger al pequeño inventor o innovador, se logra aumentar la barrera de entrada a la industria del software de forma significativa y artificial.

  Incluso el supuestamente simple hecho de litigar, para evitar el pago por una demanda injustificada de licencias por patentes, puede hacer peligrar la existencia a empresas que no tienen los recursos necesarios para esto, por el alto costo que conlleva.

- El tiempo entre la solicitud de una patente y la aprobación en la práctica puede llegar a 5 a 10 años. En una industria como la del software, en donde cada 2 a 3 años surgen nuevas generaciones de productos, claramente el sistema de patentes no calza. Un producto puede haber estado 7 años en el mercado cuando de un momento a otro se convierte en un artefacto ilegal mientras no se negocie una licencia con algún ente que obtuvo la patente respectiva. Más aún, al momento de ser concedida la patente, el invento muy probablemente ya ha pasado a la categoría de trivialidad y es usado en una multitud de programas y productos (Perchaud, 2003). Esto hace que las patentes no sean una herramienta efectiva para el que la solicita, tampoco para la industria y menos para la sociedad, que primero puede acceder a tecnología masivamente y después tiene que financiar el desarrollo.

- En relación con el software libre, el gran problema es que el paradigma en

el cual se enmarca este desarrollo es completamente diferente al paradigma de licenciamiento por uso en el caso de patentes y el software propietario tradicional. Es imposible crear software libre que a su vez incluya elementos patentados sin tener que modificar radicalmente el uso práctico de ese software. Todo el efecto alcanzado gracias al uso liberal que se permite del software libre se pierde en tal caso. Es por eso que la GPL incluye en su párrafo 7 una cláusula que impide el uso de la licencia si no es posible redistribuir el software sin impedimentos. En el caso particular de las patentes de software, eso significa que no es posible distribuir un software bajo la GPL a menos que exista una licencia gratuita para todos los que quieran hacer uso del software. Sin embargo, a pesar que exista esa licencia gratuita, de todas maneras la patente impide el progreso natural que ofrece el esquema de software libre, en el sentido que la licencia puede ser específica a programas que cumplan un standard y no se puede aprovechar el código patentado en otro programa.

- Finalmente, los ejemplos de patentes de software concedidas[8] hablan por sí solos. Queda en evidencia que, a pesar de tomar un tiempo demasiado extenso en revisar los requisitos, la revisión por parte de las oficinas de patentes no es lo prolija que debiera, o bien los revisores no cuentan con la experiencia ni conocimientos necesarios para discernir sobre la trivialidad de una invención.

  Se estima que el 60 % de las patentes relacionadas con Software en EEUU debieran ser descartadas por no cumplir con las exigencias mínimas de patentabilidad. Sin embargo, el proceso de invalidar una patente es caro y muchas veces a las empresas les sale económicamente más atractivo pagar una licencia que enfrentar al beneficiado de una patente en un juicio, con los riesgos y elevados gastos que eso implica.

5. Riesgos para Chile por patentes de software

En general, cambios en sistemas de patentes tienden a favorecer a la industria de otros países en detrimento de la industria local (Lerner, 2002). En el caso de software, es evidente que la industria nacional, lejos de percibir beneficio alguno,

[8] Ver por ejemplo http://swpat.ffii.org/patents/index.en.html
http://www.base.com/software-patents/examples.html
http://www.freepatents.org/examples/.

se verá obligada a pagar licencias por miles de patentes existentes en países como EEUU o Japón, siendo rezagado aún más de lo que está hoy en día.

En Europa existen actualmente aproximadamente 30.000 patentes de software, a pesar que la legislación actual no lo permite. En los últimos meses en Europa se ha estado discutiendo la harmonización de las legislaciones respecto de patentes de software. Si bien el proceso está aún inconcluso, existen voces claras contra el modelo que intenta imponer EEUU a través de su oficina de patentes, USPTO. Una campaña masiva permitió modificar radicalmente el proyecto de ley votado en el parlamento europeo, corrigiendo así errores que permitirían patentar cualquier tipo de software en toda europa.

A pesar de las constantes críticas al esquema seguido por EEUU[9], se sigue imponiendo este esquema a países en todo el mundo a cambio de beneficios económicos. Esto ya fue observado en 1994 (Barlow, 1994) y sigue ocurriendo hasta el día de hoy, ya sea a través de tratados de libre comercio[10], el Acuerdo sobre los Aspectos de los Derechos de Propiedad Intelectual relacionados con el Comercio / Trade Related aspects of Intellectual Property rights(ADPIC / TRIPs)[11], así como otros acuerdos internacionales que contienen elementos que intentan forzar esta política respecto de propiedad intelectual en el resto del mundo. Por lo general, estos acuerdos se realizan sin el conocimiento y menos la participación de los principales afectados. Sin embargo, ya comienzan a escucharse voces que apuntan en la dirección contraria, argumentando que no es necesario contar con una protección a la propiedad intelectual para alcanzar avances en varios ámbitos. Un ejemplo de esto es una carta[12] dirigida a WIPO (World Intellectual Property Organization), en la cual se hace notar la importancia de tomar en cuenta las experiencias exitosas de desarrollos de bienes públicos en forma colaborativa.

Referencias

Barlow, J. P. (1994). The Economy of Ideas. http://www.wired.com/wired/archive/2.03/economy.ideas_pr.html.

Bessen, J. and Maskin, E. (2000). Sequential Innovation, Patents, and Imitation. http://www.researchoninnovation.org/patent.pdf.

[9] Incluso en el reporte de la *Federal Trade Comission (FTC)* acerca de la innovación se (DeSanti et al., 2003) reconocen los errores cometidos al aceptar patentes de software.

[10] http://www.direcon.cl/html/acuerdos_internacionales/estadosunidos_12.php

[11] http://www.wto.org/english/tratop_e/trips_e/trips_e.htm

[12] http://www.cptech.org/ip/wipo/kamil-idris-7july2003.pdf

Boldrin, M. and Levine, D. K. (2001). Perfectly Competitive Innovation. Available online at http://levine.sscnet.ucla.edu/papers/pci23.pdf.

DeSanti, S. S., Cohen, W. E., Levine, G. F., Greene, H. J., Bye, M., Wroblewski, M. S., Moore, R., Barnett, M., Gorham, N., Kohrs, C., Scheffman, D., Frankena, M., Levy, R., Abbott, A. F., Michel, S., Pidano, P., and Lubell, K. (2003). To Promote Innovation: The Proper Balance of Competition and Patent Law and Policy. Technical report, Federal Trace Commission.

Hall, B. H. (2003). Business Method Patents, Innovation and Policy. Competition Policy Center. Paper CPC03-039.

Hardings, J. and Fuentes, A. (2003). Introducción al Software Libre. http://www.hardings.cl/publications/hardings2003intro.pdf.

Koboldt, C. (2003). Much Pain for Little Gain? A Critical View of Software Patents. *Journal of Information, Law and Technology*.

Lerner, J. (2002). 150 Years of Patent Protection.

Macdonald, S. (2003). Bearing the Burden: Small Firms and the Patent System. *Journal of Information, Law and Technology*.

Mandeville, T. (1982). *Engineers and the patent system: Results of a survey of members of the Institution of Engineers*. In Mandeville and Bishop (1982).

Mandeville, T. and Bishop, J. (1982). *Economic effects of the patent system: Results of a survey of patent attorneys*. Australian Government Publishing Service, Canberra.

Merges, R. P. and Nelson, R. R. (1990). On the complex economics of patent scope. *olumbia Law Review*.

Moglen, E. (1999). Anarchism Triumphant: Free Software and the Death of Copyright. *First Monday*, 4(8).

Perchaud, S. (2003). Software Patents and Innovation. *Journal of Information, Law and Technology*.

Scotchmer, S. (1991). Standing on the Shoulders of Giants: Cumulative Research and the Patent Law. *The Journal of Economic Perspectives*, 5(1):29–41.