El costo de la propiedad: cómo la propiedad intelectual limita la creatividad en software

Jose David Chaves Mena
Jose Ignacio Madriz Ortiz
Sergio Andres Zuñiga Castillo

Las patentes se ingeniaron como mecanismo de protección a la creatividad e innovación de las personas, el carbón que impulsa el tren del progreso hacia adelante de una manera segura. Pero en los últimos años, este sistema se ha convertido en la arena entre los engranajes del sistema y en particular munición que las compañías grandes utilizan para ofuscar y acabar con la competencia antes de que puedan realizarse.

Como exponen Jaffe y Lerner (2004), el software es una de las áreas donde a las cortes que aprueban patentes se les dificulta aprobar y definir criterios que produzcan patentes de calidad. Gran parte de la fricción viene de la falta de especificidad que las oficinas de patentes requieren en las descripciones del software tal que alguien más con las suficientes habilidades pueda recrear el invento. El resultado de esto fue un entorno de patentes diluidas que no parecen introducir ninguna idea nueva, de patentes que reservan la idea de una implementación sin necesariamente tener realizada la solución. Debido a esto, el dueño de la patente tiene el derecho de excluir a otros de aplicar las mismas ideas aun el dueño no teniendo una implementación siquiera parcial de la idea.

Las oficinas de patentes, en particular para el software, deben aprobar soluciones no triviales, no obvias y lo suficientemente novedosas que requieran que el aplicante describa al software a suficiente detalle, tal que las patentes le pertenezcan a gente que en realidad haya creado algo en lugar de solo tener el concepto de una idea. Esto obstaculiza el desarrollo, pues los innovadores deben gastar recursos enormes en analizar patentes y defenderse, en lugar de en crear. Además, las patentes son costosas, todo el dinero se va en el registro, la vigilancia legal y los litigios implican costos sociales y monetarios significativos que pueden beneficiar a grandes empresas con recursos para litigar en detrimento de desarrolladores independientes.

Debido a esta situación, hay gente que argumenta que el software no es algo patentable en el sentido tradicional y que no debería tratar de serlo: antes de que fuera patentable en los 80s la creación e innovación del software estaba en auge y además la naturaleza de su desarrollo es intrínsecamente iterativa, acumulativa y

cooperativa. Estos dos argumentos demuestran la fuerte desconexión entre el software y el concepto de una patente, y de ahí nació un movimiento que impulsa la idea del software libre: noción que dicta que el software debería ser público y disponible a su uso y modificación de manera indiscriminada, con las esperanzas de que el software de ayer sean los bloques fundamentales con los cuales se desarrolla el software del mañana.

El uso de patentes no es utilizado a lo largo de toda la industria de producción de software. Existen áreas donde las patentes pueden no ser aceptadas o donde no es usual que se implementen patentes por distintas políticas. Sin embargo, es sencillo darse cuenta que algunos software y tecnologías esenciales o estándares para el funcionamiento de algunos dispositivos o programas, ya se encuentran patentados. Estas incluyen tecnologías como conexiones móviles, WiFi, compresión de video, interfaces estándar como USBs, entre otras, y entran en la categoría de SEPs (*Standard Essential Patents*). Este tipo de patente debe ser implementada bajo las condiciones FRAND (*Fair, Reasonable, and Non-Discriminatory*) de manera que licenciar productos que requieran SEPs sea una opción disponible para todos. A pesar de esto, tal como recalca Lloyd (2020), estos términos FRAND son algo ambiguos, además de que la entidad validadora no parece evaluar la “esencialidad” de la patente. Aparte de este tipo de patentes, las áreas más comunes de patentes surgen de los sectores donde hay una alta cantidad manejo de datos: TIC y software empresarial, IA, blockchain, y áreas emergentes relacionadas a las anteriores.

La comunidad del software libre ha sido particularmente activa en señalar este tipo de problemas. Los defensores del software libre argumentan que el software no debería ser patentable, pues el progreso del sector siempre ha sido iterativo y cooperativo, basado en compartir conocimiento previo. Como explica Alevropoulos (2021) de la FSF, las patentes de software crean una “restricción inherente” a la libertad tecnológica al basarse en procesos matemáticos y algoritmos abstractos, rara vez representan “inventos” en el sentido tradicional.

En la práctica, los efectos restrictivos de las patentes afectan a todo tipo de software, no solo al libre, dado que suelen ser extremadamente amplias y vagas (Alevropoulos, 2021). La consecuencia es que el software puede seguir avanzando sin ellas, pero las patentes a menudo frustran activamente ese avance y, de hecho, existen pruebas sólidas de que la industria del software progresó durante décadas

sin protección por patentes y que introducirlas masivamente puede frenar el crecimiento tecnológico como mencionan Alevropoulos (2021) y Fisher (2001).

Continuando los argumentos a favor del software libre como la manera correcta y más apropiada para el desarrollo del mismo, Fisher (2001) describe los efectos desafortunados de los patentes en la tecnología:

1. Son costosos de administrar: los sistemas de registro, los saldos de los empleados, los recursos necesarios para asegurar el cumplimiento de las patentes y abogados requeridos por los grupos interesados todos suman a que el mundo de la propiedad intelectual no sea accesible y un gran consumidor de recursos sociales.
2. La propiedad intelectual cada vez con más frecuencia es un obstáculo en el camino del desarrollo acumulativo. Puede ser que una patente encarezca, al tener que ser obtenida primero, o del todo impida si así lo desea el dueño la creación de una nueva innovación que busca construir sobre algo preestablecido.
3. El sistema empodera al innovador a sobrecargar más del costo marginal para la replicación de sus productos, lo cual puede empujar a muchos consumidores finales fuera del mercado, lo cual resulta en una pérdida del exceso del consumidor que se pudo haber disfrutado en caso contrario.

Sin mencionar que la tensión existente entre las patentes y el software libre también ha generado conflictos legales a numerosas personas. Un caso es el de la demanda de las patentes de IP Innovation, la cual era una subsidiaria de Acacia Technologies (empresa española que se especializa en sistemas de gestión de almacenes (WMS/SGA)) contra los distribuidores de Linux Red Hat y Novell en 2010. En ese juicio, como menciona Tiller (2010), el jurado determinó que las patentes en cuestión eran inválidas y no habían sido infringidas. Es decir, falló a favor del software libre: patentes dobles y de cuestionable novedad fueron anuladas judicialmente. Este veredicto demostró que los tribunales pueden rechazar las “patentes malas” que simplemente reclaman ideas conocidas, y envió una señal a quienes usan las patentes para intimidar.

Otro punto crítico es que el sistema de patentes puede desalentar la adopción de tecnologías abiertas y estándares compartidos. El software moderno depende fuertemente de la interoperabilidad y la reutilización de componentes, pero la posibilidad de infringir una patente introduce incertidumbre a nivel legal. Según estudios recientes, incluso proyectos de código abierto, los cuales tradicionalmente

han impulsado la innovación, se han convertido en objetivos frecuentes de problemas, precisamente por su visibilidad y adopción masiva (Tran, 2026)

En contraste, numerosos desarrolladores independientes han denunciado casos de litigios injustos que les impiden lanzar productos, mostrando que las demandas de patentes, incluso de conceptos simples, favorecen a grandes corporaciones con mayor poder económico y experiencia legal. Por ejemplo, empresas tecnológicas han denunciado esquemas donde titulares de patentes presentan demandas débiles con el objetivo de forzar acuerdos extrajudiciales, más que defender una innovación real (Reuters, 2024).

Este fenómeno refuerza la idea de que el sistema no necesariamente protege la creatividad, sino que puede convertirse en una herramienta de coerción legal que afecta desproporcionadamente a actores con menos recursos y también evidencian una falla estructural en los criterios de patentabilidad, alineándose con lo expuesto por Jaffe y Lerner (2004) respecto a la baja calidad de muchas patentes de software.

Tristemente, desde una perspectiva crítica, estos ejemplos sugieren que el sistema de patentes ha evolucionado hacia un modelo que prioriza la protección legal sobre la innovación real. En lugar de incentivar la creación, muchas patentes funcionan como activos financieros utilizados para litigios o negociaciones, desvinculándose completamente del desarrollo tecnológico.

Otra parte negativa del mundo de las patentes es la existencia de ciertos NPEs (*non-practicing entities*), aquellos quienes poseen ciertas patentes, pero no generan nuevas tecnologías o más desarrollo con ellas. Algunas entidades como universidades obtienen patentes por el reconocimiento o prestigio que les da, sin embargo, hay un grupo más problemático. Los PAEs (Patent Assertion Entity) o conocidos vulgarmente como “*patent-trolls*” son aquellos quienes se centran de forma desproporcionada en las patentes, especialmente de software, y otros negocios similares. Estos se aprovechan de la ambigüedad o baja calidad de los estándares o las patentes y sus definiciones para crear litigios. Tal como lo mencionan Bessen et al. (2011) en su investigación, alrededor de 62% de las patentes NPE son en software y alrededor de 41% de los problemas legales alrededor de patentes de software vienen de algún NPE. Según este equipo de investigadores, el origen del crecimiento exponencial en las patentes de software se debe a las aún presentes “fronteras difusas” que presentan estas patentes y permite a estas entidades encontrar fácilmente infracciones comunes.

Como se ha analizado a lo largo del texto, las patentes han causado avances y daños al mercado de producción de software, por lo que es importante conocer algunas alternativas para proteger y monetizar este tipo de productos. La más usual es el Copyright: según TRIPS (*Agreement on Trade-Related Aspects of Intellectual Property Rights*) y el derecho internacional, se reconoce la protección por derechos de autor a programas de ordenador como “obras literarias”. Lamentablemente, tal como notan Mujtaba y Zia-Ul-Haq (2024), la definición de *obra literaria* para un producto de software protege al código concreto, por lo que la misma idea puede ser implementada en un software distinto, por lo que el uso de Copyright es insuficiente para la protección del software. Una opción más moderna y centrada en software es la que fue mencionada en el caso de Alice v. CLS y analizada por Yi (2019) protección *sui generis*. Este es un sistema específico para software que busca combatir los “*patent-trolls*”, agilizar y crear un mercado de producción de software más sano. La protección sui generis corresponde a una protección tipo “*utility model*” de 10 años para software, centrada en aspectos técnicos de implementación y no solamente en funciones abstractas, además de que tiene requisitos de divulgación técnica más altos. Sui generis fue diseñada para cubrir el constante vacío legal, las fronteras difusas y casos poco específicos que presentaban otros métodos de protección de derechos de autor para soluciones de software.

Para concluir se quiere apelar al lector no solo como desarrollador sino también como consumidor y usuario porque cuando la innovación se ve limitada por barreras legales, no solo se afectan los desarrolladores, sino también los usuarios finales. Tecnologías potencialmente útiles pueden retrasarse o no llegar al mercado debido a conflictos legales, reduciendo el bienestar general y el acceso a nuevas soluciones. En este sentido, el problema de las patentes de software trasciende lo técnico o económico, para ser un tema de interés público que afecta directamente el progreso tecnológico de la sociedad.

Referencias

Alevropoulos, P. (2021, 15 sept.). *La amenaza de las patentes de software persiste*. Free Software Foundation. https://www.fsf.org/es/blogs/la-amenaza-de-las-patentes-de-software-persiste

Bessen, J. E., Meurer, M. J., & Ford, J. L. (2011). The private and social costs of patent trolls. *SSRN Electronic Journal.* https://doi.org/10.2139/ssrn.1930272

Fisher, W. (2001). *Intellectual property and innovation: Theoretical, Empirical and Historical Perspectives. Berkman Klein Center For Internet & Society. https://share.google/FB5xBn2lsSBMvrcIh*

Jaffe, A. B., & Lerner, J. (2004). *Innovation and Its Discontents: How Our Broken Patent System is Endangering Innovation and Progress, and What to Do About It*. Princeton University Press. http://www.jstor.org/stable/j.ctt7t655

Li, Y. (2019). The current dilemma and future of software patenting. IIC; *International Review of Industrial Property and Copyright Law, 50*(7), 823–859. https://doi.org/10.1007/s40319-019-00841-w

Lloyd, I. J. (2020). 15. Software patents. En *Information Technology Law* (pp. 246–270). Oxford University Press. https://doi.org/10.1093/he/9780198830559.003.0015

Mujtaba, G., & Zia-Ul-Haq, M. (2024). The disclosure requirements of software patents: Suggestions for developing countries. *Kutafin Law Review, 11*(1), 96–123. https://doi.org/10.17803/2713-0533.2024.1.27.096-123

Reuters. (2024). *Databricks sues patent holders over alleged extortion scheme. https://www.reuters.com/legal/litigation/databricks-sues-patent-holders-over-alleged-extortion-scheme-2024-09-09*

Tiller, R. (2010, 3 mayo). *Total victory for open source software in a patent lawsuit.* Opensource.com. https://opensource.com/law/10/5/total-victory-patent-lawsuit-against-open-source-software

Tran, B. (2026). *The role of open source foundations in patent litigation*. PatentPC.https://patentpc.com/blog/the-role-of-open-source-foundations-in-patent-litigation