Universidad de Chile

Facultad de Ciencias Jurídicas y Sociales

Departamento de Derecho Comercial

Restricciones a las Obras Derivadas Bajo la GNU General Public License

Memoria para optar al grado de Licenciado en Ciencias Jurídicas y Sociales

Autor: Pablo Gabriel Fernández Osorio

Profesor Guía: Santiago Schuster Vergara

Santiago

2023

Contenidos

0Resumen

El objetivo del trabajo es analizar las restricciones a obras derivadas por parte de la GNU *General Public License*. Se considera especialmente cómo las restricciones a las obras derivadas juegan un rol en las formas de explotación de programas computacionales que existen actualmente.

Para la consecución del objetivo, el procedimiento a usar es:

1. la exposición de aquello que constituye las bases de la Protección de programas computacionales bajo la Propiedad Intelectual y su licenciamiento.
2. Análisis pormenorizado de la licencia objeto del trabajo, en su faceta histórica, sus concepciones respecto del *software* y la aplicación de cláusulas a situaciones específicas.
3. Estudio del reconocimiento de la validez de la licencia en cuanto autorización de uso de obra ajena y los límites inherentes al Derecho de Autor que afectan a su vez a la GNU *General Public License*.

Los resultados del trabajo logran establecer con cierto nivel de precisión las restricciones a las obras derivadas y otorga elementos para el análisis de casos particulares que involucren a esta licencia. También se arriba a establecer que, respecto del reconocimiento de las cláusulas de licencia no existe respuesta definitiva si se observa la jurisprudencia de la Corte Suprema.

Introducción

Cuando un programado informático crea una obra derivada de un programa computacional, ¿Comprende las implicancias legales de aquello? El Internet posibilita poder descargar y a acceder a *software*[1], frecuentemente de formas en que la creación de obras derivadas resulte sencilla, pero sucede que muchas veces el aspecto legal de ello no es debidamente considerado. Por lo demás, incluso desde antes del avenimiento del Internet que se presentan maneras alternativas de concebir la forma en que la información se comparte, contrarias a la lógica tradicional del Derecho de Autor y que, por lo tanto, la tensionan.

El código abierto, el copyleft y el *software* libre, desafían con propósito, la lógica tradicional de la Propiedad Intelectual, y la GNU General Public License, como licencia que comprende los tres fenómenos recién mencionados, representa un caso que requiere mayor análisis. En relación con lo anterior, las mayores posibilidades fácticas de crear una obra derivada, en comparación al código cerrado, conllevan a la pregunta por las restricciones a este tipo de obras.

Este trabajo se centrará en los problemas jurídicos relacionados con el uso de la GPL en obras derivadas. La necesidad de este análisis se basa en el uso extendido de la GPL en sus distintas variantes, la cual en plataformas como Github.com representa un uso en 24.19% de los programas de código abierto[2], a lo que se agrega que hay dudas respecto del alcance de lo que realmente se concede gracias a los pocos casos que se han litigado[3]. Lo anterior genera que gran parte de la industria informática pueda tener un "flanco abierto" en materia de cumplimiento de la normativa de propiedad intelectual, en ese sentido, el estudio está relacionado con las posibilidades de cometer infracciones en materia de propiedad intelectual y analizar qué es lo que la licencia permite dentro del marco legal. Alternativamente, el análisis sirve para la ponderación respecto de la conveniencia de usar la GPL en un proyecto nuevo.

La discusión en el caso particular de la GPL es más complicada que otras licencias, puesto que proviene del proyecto GNU, un proyecto que va más allá de la programación y el *software* y

[1] Señálese que, en general, se usará el termino "*software*" y "programa computacional" de manera intercambiable

[2] Ben Balter, "Open Source License Usage on GitHub.Com", *The GitHub Blog* (blog), 10 de marzo de 2015, https://github.blog/2015-03-09-open-source-license-usage-on-github-com/.

[3] Anna Haapanen, "Free and Open Source Software and the Mystery of Software Patent Licenses Under the GPL", *International Free and Open Source Software Law Review* 7, nº 1 (29 de diciembre de 2015): 19–28, https://doi.org/10.5033/ifosslr.v7i1.107.

entra dentro del ámbito de lo que se podría llamar la informática política[4], entonces aquí se enfrenta la problemática de analizar la licencia de un proyecto que, si bien no se opone en principio a lo comercial, limita gran parte de los métodos mediante las cuales se comercializa el *software*[5]. Lo anterior, como es de esperar, ha de afectar la redacción de la licencia y los enfoques que esta utiliza, pues fue concebida con dicha filosofía en mente.

En este trabajo, se postula como hipótesis inicial que la GPL impone restricciones respecto de las obras derivadas, pero en caso alguno restringe su comercialidad, por ello, se sostiene que la comercialidad de un programa bajo esta licencia es posible bajo ciertos escenarios.

El trabajo se divide en tres capítulos.

El primer capítulo dejará sentada las bases generales para la comprensión de lo que es el *Software* como obra protegida bajo la Propiedad Intelectual y las formas de licenciamiento que son de relevancia para las secciones siguientes.

En el segundo capítulo se hará el análisis pormenorizado de la GPL, la cual considera indagar en: a) el rol de esta como licencia; b) las motivaciones detrás de su redacción y existencia; c) la conceptualización que hace de obra derivada y Distribución y; d) el tratamiento pormenorizado de distintas situaciones comunes por parte de esta.

El capítulo 3 plantea un estudio generalizado de la eficacia de las disposiciones de la GPL en el derecho chileno, a fin de establecer la obligatoriedad efectiva de esta, y una mención breve a los Límites al Derecho de Autor, a fin de asentar que corresponde a aquello que no puede ser protegido por la licencia.

[4] Sin perjuicio de que se desarrolla mejor el concepto más adelante, por ahora bastará con definir este concepto como una filosofía que concibe cómo deben darse la distribución y uso de los programas computacionales, así como las relaciones proveedor-usuario.

[5] Véase la página oficial de GNU, en particular la parte de su Filosofía para entender mejor la posición del proyecto en https://www.gnu.org/philosophy/philosophy.html

1. Generalidades de Propiedad Intelectual y *Software*

En esta sección se expondrá conceptos básicos sobre licenciamiento de programas computacionales, breves explicaciones sobre por qué se usa la Propiedad Intelectual y conceptos informáticos que serán relevantes posteriormente para el análisis de la GPL. Al final del capítulo, el lector debiese tener ya las herramientas para entender el objeto de regulación de una licencia de *software,* su estatuto legal y las características que puede tener cuando es de código abierto.

Tratamientos de los programas computacionales bajo la Propiedad Intelectual

Evolución de la Protección de los Programas computacionales a nivel Internacional

El tema de si el *software* debe ser protegido por la propiedad intelectual o industrial no es del todo pacífico, incluso hay países que admiten la patentabilidad del código si existe o involucra un carácter o aprendizaje técnico[6]. En el caso de la Propiedad Industrial y en particular de las patentes, ocurría que el sistema no se adaptaba dado que se enfoca en formas de hacer, lo que transpuesto a programas computacionales, equivaldría a monopolizar los procesos intelectuales, en oposición al principio fundamental de que las ideas son de libre circulación[7]. En cuanto a un derecho *sui generis* tampoco hubiese sido la mejor solución, como lo explica Pellegrini:

> "*El derecho sui generis por su naturaleza, hubiera representado la solución más adaptada. Sin embargo, habría acarreado la temible desventaja de tener que adaptarlo país por país, lo que en la era de la informática habría generado demoras insoportables antes de haber podido llegar a establecer una protección mundial uniforme. Es pues, casi por defecto, que se prefirió el derecho de autor, toda vez que éste ya goza de un reconocimiento mundial gracias a la Convención de Berna de 1886*"[8]

A partir de esta cita, podemos destacar la transversalidad de la protección de la Propiedad Intelectual en el escenario internacional, particularmente por el alto nivel de adopción que ha

[6] *WIPO Intellectual Property Handbook* (Geneva, Switzerland: World Intellectual Property Organization (WIPO), 2008), 436.
[7] François Pellegrini, "ORIGINALITY OF SOFTWARE WORKS", *International Review of Copyright*, nº 252 (abril de 2017): 44–105.
[8] Pellegrini.

tenido la Convención de Berna. En el caso de la protección de los programas computacionales, esta proviene de progresivos avances en el derecho internacional de propiedad intelectual, destacando que un comité de expertos de la Organización Mundial de Propiedad Intelectual (en adelante "OMPI") y la UNESCO en 1985 marcó un avance decisivo en la elección del Derecho de Autor como forma adecuada de protección de los programas computacionales, meses más tarde múltiples países legislan que los programas computacionales se consideran trabajos protegidos por la propiedad intelectual y desde entonces, se acepta esta protección en detrimento de un derecho *sui generis.*[9] Esta corriente de Protección de los programas computacionales mediante Propiedad Intelectual se consagra a nivel internacional mediante la adopción de dos instrumentos internacionales:

1. El Acuerdo Sobre Los Aspectos De Los Derechos De Propiedad Intelectual Relacionados Con El Comercio, en 1994
2. El Tratado de la OMPI sobre Derecho de Autor, en 1996

El primero consagra la protección de los programas computacionales como obras literarias bajo el convenio de Berna en su artículo 10, el segundo consagra en su artículo 4 la protección de los programas computacionales como obras literarias según el artículo 2 del convenio de Berna, con independencia de su modo o forma de expresión. Respecto del Tratado de la OMPI sobre Derecho de autor, es menester señalar que en su artículo 1 expresa que el Tratado se adopta en virtud del artículo 20 del convenio de Berna, que consagra la posibilidad de que los países miembros del Convenio adopten arreglos particulares que confieran derechos más amplios que los del convenio, aparte de estipular en su preámbulo, que los países miembros reconocen la necesidad de "*clarificar la interpretación de ciertas normas vigentes a fin de proporcionar soluciones adecuadas a los interrogantes planteados por nuevos acontecimientos económicos, sociales, culturales y tecnológicos*"[10]

El programa computacional como obra literaria

[9] *WIPO Intellectual Property Handbook*, 436.
[10] OMPI, "Tratado de la OMPI sobre Derecho de Autor", 20 de diciembre de 1996, sec. Preámbulo, TRT/WCT/001, https://www.wipo.int/wipolex/es/treaties/textdetails/12740.

Los programas computacionales son protegidos mediante propiedad intelectual como obra literaria, pues corresponden generalmente a un texto plano escrito en forma de código fuente o a texto escrito o compilado en código binario. Hay que tener cuidado de no confundir la protección del código con la protección del algoritmo, la propiedad intelectual sólo protege el texto, pero no el algoritmo, por ser encontrarse en el ámbito de las ideas, las cuales no son protegidas por el Derecho de Autor[11]. En cierto sentido, cuando se habla de conjunto de instrucciones, podría pensarse que es similar a una receta, puede escribirse una receta sobre un pastel de manzana, dicha receta puede estar protegida por el Derecho de Autor, pero no da derecho a hacerse dueño del pastel de manzana ni a impedir que se escriban otras recetas sobre pastel de manzana, de la misma forma que se puede ser autor de un código para una aplicación de calculadora, pero el algoritmo detrás de dicha aplicación, al ser una idea, es de libre circulación.

La forma de expresión predilecta del programa computacional es el texto, por tanto, se le otorga la misma protección que cualquier obra literaria. Aun así, el código y los programas computacionales presentan características especiales a considerar, las que podrían en algún momento haber sugerido la Propiedad Industrial como más idónea.

Podría inclinarse, de manera instintiva, hacia la propiedad industrial dado que la comercialización del *software* se suele relacionar con el ámbito empresarial y el lucro, mientras que la propiedad intelectual se puede relacionar con lo artístico. Sin embargo, ello es una conclusión apresurada que no resiste demasiado análisis, por las razones que se expondrán a continuación. La propiedad intelectual no tiene como requisito un propósito específico detrás de la obra sino que otorga protección a las formas de expresión originales, por ejemplo, un manual de contabilidad es un trabajo evidentemente práctico y protegido por la propiedad intelectual con la misma fuerza que cualquier clásico de la Literatura mientras cumpla los requisitos de ser una forma de expresión original; en contraste, el derecho Propiedad Industrial exige que los inventos cumplan varios criterios para su patentabilidad, en particular la invención debe ser aplicable industrialmente (útil), nuevo (novedoso), demostrar "actividad inventiva" (no

[11] Este principio del Derecho de Autor se consagra en el artículo 9 del Acuerdo Sobre Los Aspectos De Los Derechos De Propiedad Intelectual Relacionados Con El Comercio, así como el artículo 2 del Tratado de la OMPI sobre Derecho de Autor. En su traducción oficial, en sus respectivos artículos ambos consagran: "La protección del derecho de autor abarcará las expresiones pero no las ideas, procedimientos, métodos de operación o conceptos matemáticos en sí"

ser obvio) y la declaración de la invención debe cumplir ciertos estándares al solicitar el registro de la patente[12]. Los requisitos de la Propiedad Industrial pueden complicar la protección de los programas computacionales. Es difícil demostrar su aplicación industrial cuando estos no participan de procesos industriales. La novedad en un programa computacional no siempre es clara, en la actualidad existen múltiples programas que cumplen funciones similares con leves diferencias, por ejemplo, Googles Docs y Microsoft Word son aplicaciones muy similares, pero con enfoques diferentes. En contraste con la Propiedad Industrial, la propiedad intelectual no exige formalidades de registro para la protección de la obra, elemento que es determinante para optar por esta en la protección del *software*. La no exigencia permite mayor protección en el área informática, pues este campo está en permanente cambio a acelerado ritmo, por lo que el uso de registros legales para la protección de los programas computacionales se torna complicado y en cambio, hace atractivo la protección automática de la Propiedad Intelectual por el sólo hecho de creación de la obra.

Los programas de computador caen dentro de la misma clasificación que una obra literaria a pesar de no estar escrito en lenguaje natural. Ha de aclararse que, en el caso del código fuente, este usa una lengua construida con un propósito determinado, es por ello inteligible para un humano que posea el suficiente conocimiento; en el caso del *machine code* o código binario, si bien no es inteligible, puede ser "descompilado" a código fuente donde sea legible[13]. A mayor abundamiento, en el derecho de la Propiedad Intelectual se exige que las obras sean una expresión original, entendiendo por expresión "*la exteriorización de un proyecto, la que ha pasado a tener una forma determinada y de tal modo permite identificar su contenido*"[14] y por original, se entiende que "*las ideas en la obra no necesitan ser nuevas pero la forma, sea literaria o artística, en que son expresadas debe ser una creación original del autor*"[15]. Un programa computacional, puede cumplir los requisitos recién delineados para ser considerado una expresión original, por cuanto este debe exteriorizarse en alguna de las distintas configuraciones que existen para dicho propósito, las que permiten determinar su contenido, en

[12] *WIPO Intellectual Property Handbook*, 17. (Traducción Propia)
[13] *WIPO Intellectual Property Handbook*, 437. (Traducción Propia)
[14] Elisa. Walker Echenique, *Manual de propiedad intelectual*, 2a. ed. actualizada (Santiago, Chile: Thomson Reuters, 2020). 70.
[15] *WIPO Intellectual Property Handbook*, 42. (Traducción Propia)

tanto que la originalidad dependerá de la manera en la que el programador plasme los algoritmos a usar.

Todo lo expuesto en esta sección y la anterior convergen en el estatuto jurídico que la Propiedad Intelectual le asigna a los programas computacionales. Si bien los conceptos específicos pueden cambiar, en general las definiciones no varían demasiado. La definición del Diccionario panhispánico del español jurídico de "programa de ordenador" señala:

> "*Adm. Secuencia de instrucciones o indicaciones destinadas a ser utilizadas, directa o indirectamente, en un sistema informático para realizar una función o una tarea o para obtener un resultado determinado, cualquiera que fuere su forma de expresión y fijación.*
>
> *Real Decreto Legislativo 1/1996, de 12-IV, por el que se aprueba el Texto Refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia, art. 96.*"[16]

De esta definición hay que rescatar que se concibe que un programa conlleva una tarea o función específica en un sistema informático o computadora "cualquiera sea su forma de expresión y fijación", lo que lo hace independiente de cualquier tecnología habida o por haber. En general el principio de Neutralidad Tecnológica requiere que las normas generales no definan los medios tecnológicos para sus objetivos, con tal asegurar una permanencia estable del marco normativo en el tiempo[17], por ello, en general cualquier definición legal de programas computacional, para ser efectivo, debiese respetar este principio.

Generalidades de la protección de los Programas computacionales bajo la Ley de Propiedad Intelectual Chilena

[16] "Definición de programa de ordenador - Diccionario panhispánico del español jurídico - RAE", accedido 11 de mayo de 2023, https://dpej.rae.es/lema/programa-de-ordenador.

[17] Lorena Donoso Abarca y Carlos Reusser Monsálvez, *Derecho informático*, Serie de documentos materiales docentes (Santiago de Chile: Academia Judicial de Chile, 2021), 20–21, https://academiajudicial.cl/wp-content/uploads/2022/03/Derecho-informatico.pdf.

La ley 17.336 de Propiedad Intelectual protege desde el artículo 1 los programas computacionales, al decir que protege los derechos de obras literarias por el sólo hecho de creación de la obra, pero para que no queden dudas de su aplicación, incluye específicamente a los programas computacionales dentro del numeral 16 del artículo 3, respecto de las obras especialmente protegidas.

En el artículo 5 letra t) se define programa computacional como:

> *t) Programa computacional: conjunto de instrucciones para ser usadas directa o indirectamente en un computador a fin de efectuar u obtener un determinado proceso o resultado, contenidas en un cassete, diskette, cinta magnética u otro soporte material.*
>
> *Copia de programa computacional: soporte material que contiene instrucciones tomadas directa o indirectamente de un programa computacional y que incorpora la totalidad o parte sustancial de las instrucciones fijadas en él.*

Dicha definición va en línea con aquella usada al inicio de la sección anterior del Diccionario Panhispánico del Español Jurídico que, como se mencionó, se usa en la Ley de Propiedad Intelectual Española. Si bien la definición de la Ley Chilena muestra su edad al mencionar los cassetes y los diskettes, cumple de manera suficiente con el principio de neutralidad tecnológica al considerar otros soportes materiales. Por otro lado, la definición de copia de programa computacional no es del todo clara, por cuanto hace la diferencia entre contener instrucciones tomadas e incorporarlas, aparte, la definición habla de "soporte material" en vez de "medio" como lo hace la definición de reproducción; si bien hay autores, como Elisa Walker, que consideran esto como una inadecuación de la norma frente al impacto de los medios digitales[18], dicha inadecuación no es insalvable. Los medios digitales involucran un soporte material, dado que los programas computacionales se almacenan en un componente físico de una computadora o accesible para esta y copiarlos conlleva modificar parcialmente los componentes físicos de una computadora con tal de fijar dicha copia, en definitiva, los medios digitales pueden ser considerados un soporte material que poseen una facilidad extraordinaria de crear nuevas copias. Considerado lo anterior, es que no se nota una diferencia entre la definición de copia de

[18] Walker Echenique, *Manual de propiedad intelectual*. 154.

programa computacional y la definición de reproducción que tenga consecuencias prácticas en la aplicación de la ley.

Además de la legislación nacional, en materia de Derecho Internacional Chile es signatario del Convenio de Berna para la Protección de las Obras Literarias y Artísticas que no reconoce expresamente la protección específica de programas computacionales. Chile también es signatario del Acuerdo Sobre Los Aspectos De Los Derechos De Propiedad Intelectual Relacionados Con El Comercio y del Tratado de la OMPI sobre Derecho de Autor que consagran que los programas computacionales están protegidos como obras literarias, en concordancia con el Convenio de Berna.

Los programas computacionales a pesar de ser obras literarias tienen disposiciones especiales bajo la ley chilena, en especial en cuanto a la titularidad del derecho (artículo 8 inciso segundo) y las excepciones y limitaciones al Derecho de Autor (artículo 71 ñ). La letra a) del artículo 71 ñ de la Ley de Propiedad Intelectual es de interés para este trabajo, pues da derecho al tenedor de un programa computacional para crear una adaptación o copia del programa cuando sea esencial para su uso, para fines de archivo o respaldo y no sea usado para otros fines. Dependiendo del alcance de las modificaciones al que sea necesario someter un programa, esta adaptación puede ser, eventualmente, una obra derivada.

Conceptos respecto del licenciamiento de Programas Computacionales

Conceptos preliminares

Antes de entrar al tema a los conceptos de licenciamiento propiamente tal, hay que explicar brevemente los conceptos de **programa computacional**, **código binario** y **código fuente.**

En la tercera versión de la GPL se define **programa** como "*cualquier obra protegible por derecho de autor licenciada bajo esta licencia*"[19], **código fuente** como "*la forma preferida de una obra para realizarle modificaciones a esta*" y **código objeto** (sinónimo de lo que se ha denominado código binario hasta ahora) como "*cualquiera forma no fuente de una obra*", si

[19] "The GNU General Public License v3.0 - GNU Project - Free *Software* Foundation", accedido 9 de agosto de 2023, https://www.gnu.org/licenses/gpl-3.0.html. (Traducción propia)

bien estas definiciones son de interés para analizar la licencia, no permiten, razonablemente, comprender de qué se tratan estos conceptos. Dado lo anterior, habrá de darse una explicación más comprensible de estos conceptos.

Los **programas computacionales** suelen ser generalmente caracterizados por la Propiedad Intelectual como un conjunto de instrucciones que son ejecutados por una computadora, en particular una definición dada por la OMPI dice:

> **"***Los programas computacionales son conjuntos de instrucciones que controlan el funcionamiento de una computadora para que pueda realizar una tarea específica, como el almacenamiento y la consulta de información*"[20]

Los programas computacionales suelen ser el resultado del **código fuente**, correspondiente al escrito de un *software*, que contiene las instrucciones donde el autor plasma las operaciones a ser realizadas en un lenguaje de programación[21][22], siendo este una forma legible para un humano de escribir programas informáticos que luego pueden ser compilados por una computadora,[23] debiendo el programador seguir las reglas que el lenguaje de programación impone para su uso.

El **código binario** (también llamado código ejecutable u objeto), es simplemente el resultado de la compilación o interpretación del código fuente y es lo que le permite a la computadora ejecutar las instrucciones descritas en el código fuente. El código máquina ya actúa a nivel de *Hardware*, ejecutando las instrucciones en los distintos componentes de la computadora[24]. En la compilación del programa se puede crear un archivo ejecutable como resultado final (en Windows son los archivos con la extensión .exe), dicho archivo ejecutable es capaz de, valga la redundancia, ejecutar el programa sin necesidad de ser recompilado y sin necesidad del código fuente y así se puede traspasar a otra computadora y ejecutarse. Piénsese en el código fuente como un plano de construcción y el archivo ejecutable el resultado de seguir las instrucciones

[20] *Principios básicos del derecho de autor y los derechos conexos* (Genève, Switzerland: Organisation Mondiale de la Propriété Intellectuelle (OMPI), 2016), 8.
[21] Anna Bajer, "Understanding of source code in language: Contribution of philosophical hermeneutics to the critical code studies", *Digital Scholarship in the Humanities* 37, nº 2 (1 de junio de 2022): 308, https://doi.org/10.1093/llc/fqab089.
[22] Ha de aclararse que no existe un solo lenguaje de programación, existen una plétora de estos. Estos lenguajes pueden ser asimilarse los unos con otros en algunos aspectos y diferenciarse en otros, en general, un mismo algoritmo puede ser plasmado en una multitud de lenguajes de programación, dependiendo la elección de. cuál usar en criterios impuestos por el programador, que pueden variar de criterios meramente técnicos a preferencias personales.
[23] Noam Shemtov, "70On Reverse Engineering and Decompilation", en *Beyond the Code: Protection of Non-Textual Features of Software*, ed. Noam Shemtov (Oxford University Press, 2017), 73, https://doi.org/10.1093/oso/9780198716792.003.0003.
[24] Shemtov, 73.

para crear el programa, similar a un edificio. De esta forma, las empresas suelen compilar el programa en una computadora y dejar a disposición de los usuarios, gratis u onerosamente, el archivo ejecutable o un instalador que deja disponible el archivo ejecutable para descargar.

Normalmente en los programas Computacionales, el código fuente y el código binario son independientes y hay que considerar la circunstancia de que el acceso al segundo, que puede ejecutarse en una computadora, no necesariamente me da acceso al primero, salvo que se aplique de manera exitosa una ingeniería reversa. Ello ha dado lugar a dos grandes modalidades de código: abierto y cerrado.

El paso histórico del código abierto hacia el cerrado como regla general en el licenciamiento

Como sugiere el título, hubo un momento en la historia en que la regla general era el código abierto. Sin realizar un relato histórico detallado de la historia detrás del *software*, ha de destacarse al menos los hechos que permitan entender el contexto histórico en que surge el concepto de código abierto y código cerrado.

Previo a que el concepto de *software* como producto existiese, "*se distribuían [los programas] con el Hardware y su protección se confundía con la del conjunto del bien comercializado, el ordenador*"[25], es en este periodo de la computación en la que empresas como IBM y DEC entregaban el código fuente a través de sus proyectos[26], marcando un periodo de la computación donde el código abierto parecía ser la regla general, en este periodo de tiempo en la que IBM empaquetaba (*bundle*) el *software* y servicios en el costo de alquiler de *Hardware*, limitando la capacidad de la competencia de cobrar por programas computacionales[27].

[25] "Propiedad Intelectual: derechos de autor y copyright", en *Aspectos legales y de explotación del software libre*, de Malcolm Bain et al. (Universitat Oberta de Catalunya, 2009), 9, http://hdl.handle.net/10609/229.

[26] Jens Hardings y Alejandro Fuentes, "*Software* Libre: introducción histórica Revisión: 1.1", 04/072003, 3, https://www.hardings.cl/publications/hardings2003intro.pdf.

[27] Peter S. Menell, "Economic Analysis of Network Effects and Intellectual Property", 2019, 308, https://doi.org/10.15779/Z381V5BD7T.

De esta época del código abierto, existen relatos que describen la existencia de "*una pequeña comunidad de científicos y programadores que intercambiara código, a la vez que informes de errores e ideas*"[28].

La pregunta ahora es, ¿qué sucedió? para explicar el cambio de paradigma de la época, Peter S. Menell hace el siguiente relato:

> "*La estructura de la industria de la computación y el rol del Derecho de Autor cambió dramáticamente durante los 70's. Con avances tecnológicos que crearon un mercado de minicomputadores y la decisión en 1969 de IBM de desempaquetar software del alquiler de computadoras ante los cargos antimonopolio, proveedores de Hardware de computadoras y desarrolladores independientes de software llegaron a usar licencias de derechos de autor para proteger programas computacionales. La apertura de un mercado de software propietario competitivo terminó una era en la que el software era compartido libremente*"[29]

Este relato es bastante completo, destacando principalmente el avance de la computación y el aumento de la competitividad de este mercado naciente, sin embargo, falta destacar un tercer elemento relevante que llevó a este cambio: la copia. Ciertamente un elemento de las computadoras modernas que hacen su uso tan extendido es la de copiar información, a diario se copia información: de internet, la nube, otras computadoras, etc. Sin embargo, esto no siempre fue así, al inicio no había tecnología que posibilitase hacer una copia de un programa computacional o utilizarlo fuera del computador de manera generalizada por lo que no era preocupación su protección[30]. Posterior al caso antimonopolio contra IBM y su decisión de separar la venta de *Hardware* y *software*, nace la necesidad de un estatuto de protección legal a los programas computacional, por su extrema vulnerabilidad a la copia, proteger la inversión que requieren y para incentivar su divulgación a más amplios sectores de la humanidad[31], dicho estatuto terminaría siendo el Derecho de Autor, a la vez el concepto de código cerrado nace como una alternativa atractiva para comercializar programas computacionales.

[28] Vicente Matellán Olivera et al., eds., *Sobre software libre: compilación de ensayos sobre software libre* (Madrid: Universidad Rey Juan Carlos, Servicio de Publicaciones : Dykinson S.L., 2004), 1.
[29] Menell, "Economic Analysis of Network Effects and Intellectual Property", 261.
[30] "Propiedad Intelectual: derechos de autor y copyright", 9.
[31] "Propiedad Intelectual: derechos de autor y copyright", 9.

A partir de lo anterior, pareciera que el paso de una etapa a otra en el desarrollo de *software* no fue más que un proceso natural en la historia de la computación y más específicamente de los programas computacionales, ciertamente hay autores que lo interpretan así, Hardings y Fuentes señalan "*si bien en un inicio todo el software era libre, esto probablemente no era más que un hecho determinado por las circunstancias*"[32], no es sino hasta después del surgimiento de los programas de código cerrado que surge lo que los autores recién citados denominan *software* (intencionalmente) libre. En este cambio en el paradigma del desarrollo de *software*, el código fuente pasa de ser algo normalmente compartido, a ser reservado, ello determina que el acto de publicación de dicho código sea un acto deliberado con un propósito.

Terminada esta contextualización histórica, es tiempo de caracterizar en mayor detalle cada modalidad, código abierto y código cerrado. Posterior a ello, se realizará una descripción general del *software* libre y *Copyleft*, formas de licenciamiento alternativas que son de interés junto con el código abierto para lo que será el análisis de la GPL.

Código Cerrado

El **código cerrado** (llamado a veces *software* propietario o privativo) es la modalidad que usan comúnmente los programas de uso comercial, donde se distribuyen un binario o un conjunto de ellos, pero no se da acceso al código fuente, que permanece en poder del distribuidor, por ejemplo, Microsoft da acceso a la aplicación Word por un precio, pero no se puede acceder al código fuente que se usó para crear dicha aplicación.

Considerando la independencia entre el código binario y el código fuente, este último, que es el código que en principio permite crear una obra derivada, tiene, bajo la modalidad del código cerrado una protección fáctica[33] además de la jurídica, que es el secretismo que rodea al código fuente. Por ello, las licencias de código cerrado se suelen centrar en el uso del programa, prohibición de ingeniería reversa y en exenciones de responsabilidad.

[32] Hardings y Fuentes, "*Software* Libre: introducción histórica Revisión: 1.1", 5.
[33] Además de esta protección, los distribuidores de *software* pueden incluir tecnologías para otorgar una protección aún mayor del *software*, erosionando mayormente las posibilidades del usuario de modificar el programa, estas suelen ser conocidas como medidas de protección tecnológicas (TPM, por sus siglas en inglés).

En general se puede señalar que el código cerrado se caracteriza por la reserva del código fuente, siendo también posible que existan actos de ocultamiento del código fuente más allá de la reserva, como las cláusulas de prohibición de ingeniería reversa sobre un programa.

Código Abierto

La segunda modalidad es el **código abierto**, que se suele usar en aplicaciones de uso no comercial o cuya comercialidad no se encuentra en la distribución de la aplicación, pudiendo acceder al código fuente libremente, ya sea que esté disponible para todo el mundo o a petición[34].

Dentro del código abierto, se han creado diversas modalidades de licenciamiento, debido a que exista una miríada de nuevos problemas que surgen con el acceso al código fuente por parte del público. Dado que existe este acceso al código fuente, si se requiere crear una variación del programa, no es necesario aplicar ingeniería reversa o crear un programa desde cero, sino que se puede crear una copia del código fuente desde el repositorio y modificarla desde esa base, en términos informáticos, esto es lo que se denomina crear un *fork.* Los *forks* suelen introducir modificaciones al código fuente, por lo que usualmente terminan siendo versiones modificadas, y por tanto, bajo la propiedad intelectual, obras derivadas. Dentro de las problemáticas que deben considerar las licencias de código abierto, está la pregunta de si se permite introducir modificaciones al código fuente, si puede derivarse un uso comercial de él, en qué condiciones debe ser creado y compartido los *fork* con o sin modificaciones, si debe ser de código abierto o puede ser de código cerrado, etc.

El problema se agrava cuando el código se lanza al público sin licencia que la cubra, lo que refleja probablemente un problema con que Github, el repositorio dominante, no requiere licencias y falta de comprensión (de los desarrolladores de *software*), aunque se ha tratado de remediar esta falta dejando claro que sin licencia, el código no es *open source*[35], exactamente, Github dice que "*a menos que usted ponga una licencia en su código fuente, su Copyright*

[34] Un ejemplo es el uso de Zotero como gestor bibliográfico, cuyo código fuente en sus distintas variantes se encuentra disponible en internet en https://www.github.com/zotero

[35] Ian Walden, "Open Source as Philosophy, Methodology, and Commerce: Using Law with Attitude", en *Open Source Law, Policy and Practice*, ed. Amanda Brock, 2ª ed. (Oxford University PressOxford, 2022), 8, https://doi.org/10.1093/oso/9780198862345.003.0001. (Traducción Propia)

exclusivo aplica y nadie puede realmente utilizar su código"[36]. El problema que subsiste en los códigos fuentes no licenciados disponibles al público, es que, incluso si desde un punto de vista legal no hay autorización, en los hechos sigue fácil reproducir dicho programa o transformarlo, prácticamente sin consecuencia alguna y con independencia de la voluntad del autor.

A pesar de que el relato que se ha estado construyendo sirve para contextualizar los problemas del código abierto, aún no se responde qué es exactamente código abierto. Véase el caso que se acaba de plantear, si se publica un código sin licencia no hay autorización de uso, ¿es eso código abierto? Ciertamente en el sentido más literal de la palabra, lo es, por cuanto está disponible para ver y estudiarlo, aunque no pueda usarse, pero el problema en realidad no es tan simple. La realidad es que no existe un concepto inequívoco de código abierto, por lo que la respuesta final puede variar dependiendo de a quien se le pregunte.

En este trabajo ha de decantarse por la definición de código abierto que ha dado la *Open Source Initiative* (en adelante "OSI"), pues en la práctica, ha sido una suerte de órgano que ha creado el estándar con la definición de código abierto y el sello OSI[37]. Dicha definición contiene en total 10 criterios[38] para que una licencia sea considerada como de código abierto debe:

a) permitir redistribución libre del programa;

b) dar acceso al código fuente;

c) permitir la creación de obras derivadas;

d) asegurar la integridad del código del autor;

e) no discriminar contra persona o grupo alguno;

f) no discriminar contra ningún área de trabajo;

[36] Gina Häußge, "A dev's guide to open source software licensing", accedido 26 de junio de 2023, https://github.com/readme/guides/open-source-licensing.

[37] Ross Gardler y Stephen Walli, "Evolving Perspective on Community and Governance", en *Open Source Law, Policy and Practice*, ed. Amanda Brock, 2ª ed. (Oxford University Press, 2022), 48, https://doi.org/10.1093/oso/9780198862345.001.0001.

[38] Estos criterios serán mencionados pero no serán explicados aquí en su totalidad, ello es fútil para efectos de este trabajo ahora mismo. Estos criterios son explicados en mayor profundidad en su aplicación a la GPL.

g) realizar distribución de la licencia junto con el programa a todo aquel que lo reciba, sin necesidad de celebrar otra licencia;

h) no hacer depender a los derechos bajo la licencia al hecho de que el programa forme parte o este unido a algún producto u otro programa en específico;

i) no aplicar sus disposiciones a programas que se almacenen en el mismo medio y;

j) ser tecnológicamente neutro[39].

Bajo este criterio, el caso planteado anteriormente sobre los códigos sin licencia, no sería código abierto, porque los requisitos de la OSI requieren que haya una licencia que cumpla con estos parámetros mínimos de libertad para el recipiente de la licencia. En ese sentido, el código está meramente disponible, en el sentido de que es posible su escrutinio, no pudiendo ser considerado tampoco código cerrado por lo mismo. Bajo la premisa anterior, atenerse al criterio de la OSI es problemático, pero de no usarse, se dificulta la determinación de qué programas computacionales son de código (verdaderamente) abierto y cuales son abiertos por mera voluntad del autor, es dudoso que sea de código abierto un programa cuyo autor no se encuentra legalmente obligado a mantenerlo así en el futuro, pudiendo eliminar su publicación, discriminar quien puede acceder al código fuente, utilizar el programa, etc. Es por ello que ha de distinguirse entre programas de código abierto y aquellos cuyo código está meramente disponible, cumpliendo los primeros con la definición de la OSI, no así los segundos.

Modalidad de software libre o FOSS

Para entender este concepto, es necesario explicar a fondo un elemento del código abierto, que es su uso como una expresión político-filosófica, que se contrapone al uso del código abierto como un método para el desarrollo de *software.* En principio, el uso de código abierto podrá parecer contraintuitivo, ya que se tiende a pensar que el autor de un programa querrá explotarlo según las concepciones de la economía clásica, pero hay que entender que detrás de las

[39] "The Open Source Definition", Open Source Initiative, 7 de julio de 2006, https://opensource.org/osd/.

motivaciones del código abierto puede existir una racionalidad política y/o económica, Mirko Böhm concibe al código abierto en la sociedad actual de la siguiente forma:

> "*El acceso ubicuo a Internet, combinado con el código abierto, permite formas participativas de toma de decisiones en la sociedad antes inviables, enfoques alternativos para la transferencia de conocimientos y oportunidades para el suministro de infraestructuras públicas de tecnologías de la información y la comunicación (TIC), con menos ataduras a proveedores tecnológicos concretos.*"[40]

Bajo la premisa de la obra recién citada, detrás del código abierto existe una posible motivación política y de transformación social, en este trabajo dicha concepción es particularmente relevante pues la *Free Software Foundation* (en adelante "la FSF") quien es formalmente la redactora de la GPL, concibe de esta forma el código abierto desde su eslogan "*Free as in Freedom*" (cuya traducción es "Libre como en Libertad", en contraposición a "*free*" como el equivalente en inglés a libre de costo). La FSF posee una visión de **informática política** determinada, entendiendo por tal una posición respecto de cómo se debiese proceder respecto de la distribución y uso de los programas computacionales, así como una posición respecto de cómo dicha forma de distribución y uso debiese concebir las relaciones proveedor-usuario. La FSF es contraria no a la comercialización, pero debido a su concepción de política informática, sí se opone a la forma tradicional de comercialización de los programas computacionales, activamente promocionando una forma alternativa de suministro de *software*, contrariando la forma del suministro de TIC que usan actualmente las compañías de informáticas como FAANG[41], Adobe, Microsoft, etc.

Antes de entrar a ver la concepción del *software* libre, es menester señalar que los proyectos de código abierto **pueden mas no necesariamente tienen** motivaciones políticas y no comerciales. Existen proyectos de código abierto que se usan de base para un proyecto de código cerrado, como el proyecto Chromium, de código abierto, que sirve de base para Chrome, de código cerrado, ambos de Google. Existen proyectos que parten siendo cerrados, pero abren su código posteriormente, como Go o Tesseract OCR, el primero una lengua de programación creada por

[40]Mirko Böhm, "Economics of Open Source", en *Open Source Law, Policy and Practice*, ed. Amanda Brock (Oxford University Press, 2022), 300, https://doi.org/10.1093/oso/9780198862345.003.0015. (Traducción Propia)
[41] Facebook (ahora Meta), Apple, Amazon y Google

Google y la segunda un programa tipo OCR creada y posteriormente abandonada por Hewlett Packard. Existe una relación innegable entre la industria comercial del *software* y el código abierto, uno de los proyectos "insignia" de la comunidad FOSS es el uso del Linux Kernel como base para sistemas operativos alternativos a Windows y MacOS, a la vez que Linux es la base de muchos proyectos comerciales, como Red Hat Enterprise Linux, cosa que incluso se ve reflejada en los miembros de la Linux *Foundation,* que incluye, irónicamente, a Microsoft como miembro *platinum*[42].

Sólo ahora puede explicarse la diferencia entre código abierto y libre, haciendo uso de los demás factores anteriormente explicados. El *software* libre debe entenderse en el contexto del uso de código abierto como una filosofía de lo que hemos denominado informática política. FOSS es el acrónimo de *Free Open Source Software* (programas libres de código abierto), entonces ya explicamos la parte de código abierto como la posibilidad de ver o acceder al código abierto y usarlo, bajo ciertos parámetros predefinidos por la OSI, ahora es tiempo de indagar el sentido de "Libre". Ya se ha ido adelantado el sentido de Libre pero no se ha explicado en detalle, hemos de tomar a la FSF como referente, no sólo por el enfoque de este trabajo en la Licencia GPL, sino que es que una iniciativa que da protagonismo a la palabra Libre y ha hecho esfuerzos por explicarlo de la manera más sencilla posible. De la propia página de la FSF se lee:

> "*Software libre" significa software que respeta la libertad de los usuarios y la comunidad. A grandes rasgos, significa que* ***los usuarios tienen libertad para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software****. Por tanto, el "software libre" es una cuestión de libertad, no de precio.*[43]

A mayor abundamiento, la libertad en el sentido que lo entiende la FSF se divide en "las cuatros libertades esenciales" (*the four essential freedoms)*:

> "*La libertad de ejecutar el programa como desees, para cualquier propósito (libertad 0).*

[42] Véase https://www.linuxfoundation.org/about/members

[43] "What is Free *Software*? - GNU Project - Free *Software* Foundation", accedido 1 de mayo de 2023, https://www.gnu.org/philosophy/free-sw.html. (Traducción Propia)

La libertad de estudiar cómo funciona el programa y modificarlo para que realice las tareas informáticas que desees (libertad 1). El acceso al código fuente es una condición previa para ello.

La libertad de redistribuir copias para ayudar a otros (libertad 2).

La libertad de distribuir copias de tus versiones modificadas a otros (libertad 3). De este modo, darás a toda la comunidad la oportunidad de beneficiarse de tus cambios. El acceso al código fuente es una condición previa para ello."[44]

Estas libertades resaltan mejor como la FSF concibe la relación usuario-proveedor y el sentido de la libertad en su concepción de FOSS. También se deja entrever cómo la FSF concibe un modo de desarrollo comunitario en la libertad 3, en el cual la redistribución es esencialmente una oportunidad para beneficiar a la comunidad entera.

Estas 4 libertades se pueden ver como la concepción exclusiva de la FSF pero la realidad es que han sido influyentes en la concepción general de cómo debe ser una licencia de código abierto tipo FOSS, Böhm relata:

"*En la actualidad, los requisitos formales para que un código sea de código abierto se basan en las "cuatro libertades del software" establecidas por la Free Software Foundation (FSF) y, de forma más general, en la Open source Definition (OSD) de la Open Source Initiative (OSI).*"[45]

Copyleft

Este, más que un concepto, es una categoría de licencias que no se limita a lo informático, sino que se usa como un nuevo paradigma de distribución de derechos en general, sólo por dar un ejemplo, las licencias *Creative Commons* incluyen versiones tipo copyleft pensadas para el ámbito literario. Pero ¿qué es exactamente el *copyleft*? Una definición concreta que se ha dado es la siguiente:

[44] Ibid.
[45] Böhm, "Economics of Open Source", 303. *(Traducción Propia)*

> "*El copyleft es un método general para hacer que un programa (u otra obra) sea libre (en el sentido de libertad, no de gratis), y exigir que todas las versiones modificadas y ampliadas del programa sean también libres.*"[46]

Esta definición es la que ha dado el proyecto GNU en su página oficial, siendo la FSF la encargada del proyecto GNU, ha de entenderse que el sentido de libertad ha de ser el mismo que se ha descrito en la sección anterior. Sin embargo, esa es la definición de la FSF pero, ¿qué es el *Copyleft* jurídicamente? Véase una definición jurídica adecuada que ha sido escrita:

> "*En lenguaje jurídico, el Copyleft es una licencia que garantiza el derecho a cualquier usuario de utilizar, modificar y redistribuir un programa o sus derivados, siempre que se mantengan las mismas condiciones de utilización y difusión establecidas en la licencia otorgada por el titular de los derechos de una obra.*"[47]

Lo único a mejorar de esta definición es el hecho de que habla de "un programa o sus derivados" cuando la modalidad *copyleft* se usa en diversos ámbitos, por lo que debería decir "una obra y sus derivados". Es de interés destacar lo anterior, dado que los problemas jurídicos del *copyleft* no se limitan al tipo de obra que se aplica, sino que representan un desafío para la lógica entera del Derecho de Autor, entonces para el análisis de una licencia *copyleft* de un programa computacional (como el de este trabajo), no debe descartarse lo que se ha dicho sobre licencias *copyleft* de otros tipos de obras bajo el Derecho de Autor.

Sin perjuicio de que será tratado con mayor más adelante, hay que dejar enunciado que el principal problema que acecha a las licencias *copyleft* es su validez como ejercicio de los derechos económicos por parte del autor, en países como España, la ley de Propiedad Intelectual señala en su artículo 17 que la explotación de los derechos exclusivos de explotación pueden ser ejercidos **de cualquier forma**, mientras que en Chile la ley de Propiedad Intelectual señala en su artículo 20 que las autorizaciones otorgadas por el titular de los derechos se hacen de forma **contractual**, dicha variación en los términos legales sumada a la visión dogmática que

[46] "What is Copyleft? - GNU Project - Free *Software* Foundation", accedido 14 de junio de 2023, https://www.gnu.org/copyleft/. (Traducción Propia)

[47] Liliana Galindo Díaz, Andrea Martínez Devia, y Rodolfo Yáñez Otálora, "Los derechos de autor en el entorno digital y el Internet en Colombia: una mirada al estado del arte actual y sus principales problemas", *Revista de Derecho, Comunicaciones y Nuevas Tecnologías*, nº 10 (diciembre de 2013): 24.

tengan los tribunales puede hacer variar las posibilidades de que una licencia *copyleft* sea considerada válida.

2. La GNU General Public License

En esta sección se analizará la GPL en términos generales.[48].

Antes de comenzar la sección, se ha de explicar la metodología respecto de las distintas versiones que conforman a la GPL. Existen actualmente tres versiones de la GPL, además de contar la versión dos y tres de la GPL con distintas variantes con propósitos específicos[49]. Para efectos de este trabajo interesa una descripción general de la licencia, por lo que se centrará en las características de la GPL en su versión dos y tres y no de las variantes que existan. Nótese que la variante GPL v3 *Affero* es de interés, por lo que será revisada en el siguiente capítulo, en su propia sección. En General, cuando se hable de "la licencia" o de la "GPL" sin referirse a una versión específica, ha de entenderse que se está refiriendo a algo que aplica tanto a la v2 y la v3.

Entendiendo la causa detrás de la GPL

Antes de entrar al análisis de las cláusulas de la GPL, hay que profundizar más en el contexto histórico en que surgió esta licencia y a qué se refiere exactamente con sus libertades, a modo de tener una visión más clara de la lógica que se utiliza y su propósito.

Más allá de que el caso de la FSF esté influenciado particularmente por las experiencias de su fundador, Richard Stallman[50], lo cierto es que el movimiento de *software* libre en general surge como un movimiento de oposición "frente a la tendencia de apropiarse del conocimiento científico y tecnológico de las grandes empresas de informática"[51].

[48] Disponible en sus distintas variantes en: https://www.gnu.org/licenses/licenses.html. Las cláusulas de la GPL que sean mencionadas no serán transcritas en su totalidad, salvo que sea necesario para el mejor entendimiento de la licencia, ya que ello es fútil para los propósitos de este trabajo, pudiendo el lector buscar y descargar el texto completo de la licencia

[49] Por ejemplo, existe la variante *Lesser* que es más compatible con programas de código cerrado, o la variante *Affero* que está pensada para programas que vayan a ser usadas a través de una red. Las distintas variantes nacen como respuestas a necesidades específicas, por ejemplo, en el caso de la variante *Affero*, nace para superar la debilidad de la GPL original de limitarse a señalar la obligación de acceso al código fuente con la entrega de códigos binarios, lo que permite a programas usados en la nube a no compartir su código fuente.

[50] Véase Tennille Christensen, "The GNU General Public License: Constitutional Subversion", *Hastings Constitutional Quaterly* 33, nº 4 (2006): 399-404.

[51] Malcolm Bain et al., *Aspectos legales y de explotación del software libre* (Universitat Oberta de Catalunya, 2009), 4, http://hdl.handle.net/10609/229.

Este relato, que debe relacionarse con el paso histórico del código cerrado al abierto, es incorporado en la visión de la GNU:

"*El proyecto para desarrollar el sistema GNU se denomina «Proyecto GNU». El Proyecto GNU se concibió en 1983 como un modo de retomar el espíritu cooperativo que prevalecía en la comunidad informática en sus comienzos, posibilitar la cooperación eliminando los obstáculos impuestos por los dueños de software privativo.*"[52]

A partir de lo anterior, es que se ve como el proyecto GNU posee esta visión de espíritu cooperativo, que hace alusión al estado del arte en desarrollo de *software* previo al surgimiento del código cerrado. Obsérvese que este la GNU aparece poco después de los eventos transcurridos en el paso al uso de código cerrado y poco antes de que el Derecho de Autor surgiese como la alternativa definitiva para la protección de los programas computacionales

El eslogan que usa la GNU para promocionar este movimiento suele ser que "*la palabra «libre» en «software libre» se refiere a libertad, no a precio*"[53], pero ¿a qué se refiere exactamente la GNU con "libertades"? Para responder esta pregunta, se hará alusión a la evolución del discurso en relación con estas.

De los primeros discursos sobre *software* libre de Richard Stallman, el mayor exponente de la FSF y su ideología, se da el siguiente ejemplo:

> "*Imagina que compras una casa y el sótano está cerrado y sólo el constructor original tiene la llave. Si necesitaras hacer algún cambio, reparar algo, tendrías que acudir a él, y si estuviera demasiado ocupado haciendo otra cosa te diría que te perdieras y te quedarías bloqueado. Estás a merced de esa persona y te vuelves abatido y resignado. Eso es lo que ocurre cuando los planos de un programa informático son mantenidos en secreto por la organización que lo vende. Es la forma habitual de hacer las cosas*"[54]

[52] "Visión general del sistema GNU - Proyecto GNU - Free *Software* Foundation", accedido 18 de octubre de 2023, https://www.gnu.org/gnu/gnu-history.es.html.
[53] Ibid.
[54] "Richard Stallman at the First Hackers Conference in 1984 - GNU Project - Free *Software* Foundation", accedido 21 de noviembre de 2023, https://www.gnu.org/gnu/first-hackers-conference-1984.html. (Traducción Propia)

Este discurso, de 1984, da a entender una idea de propiedad, del *software* de código cerrado como una de pérdida del control de un bien material.

Aunque ya había sido mencionada por Stallman antes, la idea de solidaridad se refleja mejor en esta cita del Manifiesto GNU:

> "*Considero que la Regla de Oro requiere que si me gusta un programa debo compartirlo con otras personas a las que les guste. Los vendedores de software quieren dividir a los usuarios y conquistarlos, haciendo que cada usuario se comprometa a no compartirlo con los demás. Me niego a romper la solidaridad con otros usuarios de esta manera. No puedo, en conciencia, firmar un acuerdo de confidencialidad o un acuerdo de licencia de software.*"[55]

Es en este conjunto de dos ideas en que se entiende mejor las libertades de la FSF, las cuales pasan de ser un concepto abstracto a ser algo más aterrizado a problemáticas concretas que se relacionan con aquellas temáticas de las que se ocupa la informática política. Por un lado, el sentido de la propiedad es lo que motiva principalmente la libertad 0 de ejecutar libremente el programa y la libertad 1 de poder estudiar cómo funciona el programa, pues esto permite al usuario ser dueño íntegramente de lo que ocurre en la computadora, pues puede saber exactamente que está sucediendo cuando corre un programa y escoger cuándo y cómo ejecutar las instrucciones que contiene dicho *software*. Por otro lado, este sentido de solidaridad y comunidad, basada en el estado de la computación previa aparición del código cerrado motiva las libertades 2 y 3 sobre redistribución de copias y versiones modificadas, pues ello permite a quien posea el programa o una versión modificada de ella, a ayudar o apoyar a otras personas que necesiten el programa para sus propios propósitos.

Entonces, de esta sección se debe destacar que las motivaciones de la GPL como la licencia insignia del *software* libre, están relacionadas principalmente con cumplir estas visiones de permitir al usuario no perder su facultad de controlar su computadora, y de promocionar la solidaridad que existe en la comunidad informática.

[55] "The GNU Manifesto - GNU Project - Free Software Foundation", accedido 21 de noviembre de 2023, https://www.gnu.org/gnu/manifesto.html. (Traducción Propia)

¿qué significa que la GPL es una licencia?

Siendo la GPL originaria de Estados Unidos, es natural dirigir la pregunta hacia lo que el *common law* entiende por licencia, en términos generales se le ha definido de la siguiente forma:

> "*Como implica la palabra licencia, el propietario del bien o derecho, el* ***licenciante****, concede permiso a una persona u organización independiente, el* ***licenciatario****, para fabricar o hacer fabricar, vender o hacer vender y utilizar productos, procesos y servicios que incorporen el bien o derecho específico en cuestión, con fines comerciales, pero sin transferir el título de propiedad.*"[56]

En términos generales, como casi siempre comprenden una oferta, su aceptación y contraprestaciones[57], lo correcto es llamar a estas licencias, contratos[58]. No obstante lo anterior, la calidad de contrato de las licencias FOSS bajo el *common law* ha sido puesta entre dicho, habiendo duda, en la jurisprudencia estadounidense, si las cláusulas de estas deben ser consideradas contractuales o si son meras exenciones unilaterales (*waiver*)[59].

La pregunta que falta responder es si se le da el mismo tratamiento bajo la tradición del derecho continental, pues el enfoque de este trabajo de analizar la GPL bajo el derecho chileno. Volcándose directamente a licencia de *software* (y no licencia en general), una definición que es ilustrativa y que es de especial interés por haber sido citada por la Corte Suprema[60], es la que dan Bain, Gallego, Martinez y Rius:

> "*Por licencia de software entendemos el* ***contrato*** *que establece* ***los términos y condiciones*** *en virtud de los cuales el autor o titular de los derechos de explotación de un programa informático (al que llamaremos "el licenciante", "titular del software" o*

[56] Charles D. Desforges, *The Commercial Exploitation of Intellectual Property Rights by Licensing: A Thorogood Report* (London: Thorogood, 2001), 2. (Traducción Propia)
[57] El término original usado aquí es *Consideration*, en relación con la doctrina contractual del *common law* en el cual la obligatoriedad de los contratos depende en la existencia de un intercambio hecho entre las partes, donde ambas comprometan algún derecho legal.
[58] Desforges, *The Commercial Exploitation of Intellectual Property Rights by Licensing*, 2.
[59] Véase el análisis jurisprudencial de Estados Unidos que se hace en: Maggs, Peter B. "License Contracts, Free *Software* and Creative Commons in the United States". The American Journal of Comparative Law 62 (2014): 407–23.
[60] Véase la Sentencia Rol 44948-2016 (Corte Suprema 18 de enero de 2018)

"proveedor del software") ***permite la utilización de dicho programa*** *a otra persona (al que llamaremos "el licenciatario o usuario").*"[61]

En contraste con la anterior, aquí se habla únicamente de utilización, aunque, en un sentido amplio, utilización puede incluir la venta o fabricación usando el programa. Aquí se ve cómo, al igual que en la definición anterior, se caracteriza la licencia como contrato, aunque en el caso del derecho chileno esto merece mayor análisis, la que se realiza más adelante en el siguiente capítulo

Entonces, la GNU *General Public License*, como licencia de "Público General", es el documento que establece los términos y condiciones que permiten la utilización de los programas que estén bajo esta licencia. De acuerdo con las cláusulas de la GPL, en ambas versiones, la manera de dejar un programa bajo la licencia es indicarlo de acuerdo con los párrafos que se señalan al final del documento, donde se indica el texto y la información que se debe señalar en los archivos del programa, además de las obligaciones de acompañar copias de la licencia junto con el programa y dar suficiente aviso del *Copyright*[62].

Esta caracterización de la GPL como licencia de *software* permite comprender mejor su función legal, qué pretende regular y las disposiciones legales que le son aplicables. En efecto, la GPL como documento legal pretende ser la autorización legal para quienes reciban copias de un programa cubierto, imponiendo términos y condiciones para la utilización del programa, remitiéndose a las normas de Derecho de Autor aplicables en numerosas ocasiones.

Ahora se debe caracterizar qué tipo de licencia de *software* es la GPL.

En primer lugar, se analizará si cumple los requisitos para ser considerada de código abierto bajo los parámetros de la definición de la OSI, siendo la GPL aprobada como licencia de código abierto[63], ha de presumirse que sí, pero véase cómo se cumple específicamente los requisitos:

---

[61] "Las Licencias de *Software* (I)", en *Aspectos legales y de explotación del software libre*, de Malcolm Bain et al. (Universitat Oberta de Catalunya, 2009), 9, http://hdl.handle.net/10609/229.

[62] El término original usado por la licencia es *Copyright notice*, entendiendo que la GPL es de origen estadounidense, esto refiere entonces al aviso de *Copyright* descrito en el capítulo 4, §401 y siguientes del Título 17 del *United States Code*, conocido como la ley de Derecho de Autor Estadounidense. Este aviso legal corresponde a la fórmula de © (nombre del autor) (año), sin embargo, la GPL establece formulas personalizadas que están descritas en la licencia para su uso.

[63] Véase https://opensource.org/licenses/?ls=GNU

1. Redistribución del *software*: La GPL establece que se puede redistribuir las copias que se reciban del programa tanto en forma de código fuente o binario, siempre que se cumplan las condiciones que se establecen para ello, principalmente mantener la misma licencia y dejar aviso suficiente de esta.
2. Acceso al código fuente: La licencia establece que se debe acompañar el código fuente junto con el código binario de un programa o dejarlo disponible a través de los medios permitidos, como una oferta escrita o dejándolo en algún lugar designado. Los medios específicos dependen de la versión de la licencia.
3. Obras Derivadas: Se permite la creación de obras derivadas, siendo obligatorio que está este también bajo los términos de la GPL, lo que significa que se les aplica el mismo deber de permitir acceso al código fuente.
4. Integridad del código fuente del autor: Este requisito es más complejo que los anteriores, la definición de código abierto de la OSI define este requisito de la siguiente forma:

> "La licencia puede restringir la distribución del código fuente modificado sólo si permite la distribución de "archivos de parche" [*patch files*] con el código fuente para modificar el programa al momento de la compilación. La licencia debe permitir explícitamente la distribución de *software* creado a partir de código fuente modificado. La licencia puede exigir que las obras derivadas lleven un nombre o un número de versión diferente del *software* original."[64]

La primera hipótesis refiere a que en caso de no permitir la distribución de código fuente modificado, debe al menos permitir la distribución de archivos parches, ello ha de relacionarse con lo exigido en el requisito anterior. Esto es una posibilidad para la licencia y no una obligación, la que no es aplicada por la GPL, por lo que los aspectos más técnicos no serán explicados, pues puede generar confusión. La segunda hipótesis corresponde a una obligación de permitir la distribución de obras derivadas en forma de código binario, esta es cumplida a cabalidad por la GPL. La tercera hipótesis de este requisito refiere a que la licencia puede exigir que se use un nombre o número de versión distinta a fin de diferenciar más claramente la obra derivada de la original, esta

[64] "The Open Source Definition". (Traducción propia)

posibilidad es usada por la GPL desde que exige que las obras derivadas lleven un aviso prominente de que han sido modificadas en relación con la obra original.

5. No discriminación contra personas o grupos: En general no hay ninguna cláusula de la GPL que establezca una discriminación contra alguna persona o grupo. En particular, es necesario mencionar que la GPL no impone ninguna limitación a la comercialización del *software* ni impide su uso por empresas.
6. No discriminación contra áreas de trabajo: Esta licencia es neutra en cuanto a las aplicaciones que se les puede dar al programa, es más, se establece una libertad para correr el programa y modificarlo para ajustarlo a los propósitos del usuario.
7. Distribución de la licencia: La GPL como licencia es autosuficiente, no requiere celebrar otra licencia o contrato entre las partes que participan en la cadena de la distribución de la obra para que se otorguen los derechos sobre el programa.
8. Licencia no es específica a un producto: Para que este requisito quede más claro, habrá que ver la caracterización de la OSI:

   > "Los derechos asociados al programa no deben depender de que el programa forme parte de una distribución de *software* concreta. Si el programa se extrae de esa distribución y se utiliza o distribuye dentro de los términos de la licencia del programa, todas las partes a las que se redistribuya el programa deben tener los mismos derechos que los que se conceden junto con la distribución original del *software*."

   En la misma línea que el requisito 7, la GPL es también neutra en cuanto a si el programa puede o debe ser inserto a tecnología específica para la aplicación de los derechos que se conceden bajo sus cláusulas.
9. La licencia no debe restringir otro *software* que se distribuido con el programa: este requisito necesita mayor explicación. Los términos exactos de la OSI son:

   > "La licencia no debe imponer restricciones en otro *software* que sea distribuido junto con el programa licenciado. Por ejemplo, la licencia no debe insistir en que

todos los demás programas distribuidos en el mismo medio deban ser *software* de código abierto."[65]

La GPL establece que los programas que sean independientes, como aquellos que sólo estén casualmente en el mismo medio de almacenamiento, no quedan bajo los términos de la licencia. No obstante lo anterior, la licencia sí aplica sus términos a programas que estén enlazados con la obra cubierta, por ahora basta decir que son programas que interactúan con la obra cubierta para el funcionamiento de uno u otro. Ello podría verse como la imposición de restricciones en otro *software*, pero forzosamente ha de concluirse que la OSI estima que la GPL cumple este requisito si la considera código abierto, ¿cómo se concilia esta aparente contradicción? Una posible respuesta es que, técnicamente, estos programas enlazados pasan a ser parte del *software* licenciado, cosa que definitivamente ocurre en algunos casos, aun así, a la fecha en que se escribe esto, no hay constancia de un documento de la OSI que explique esto en detalle, por lo que no hay una respuesta oficial, pero la que se ha dado aquí se estima suficiente para los propósitos de este trabajo.

10. Neutralidad Tecnológica: La GPL es tecnológicamente neutra, si bien existen términos que pueden relacionarse con una tecnología en particular, como Librerías del Sistema o Interfaz Estándar, la licencia se encarga de definirlos y usarlos en contextos lo suficientemente neutros para no ser relacionados con una tecnología en particular, por lo que cumple este requisito.

En base a lo recién expuesto, ha de concluirse que la GPL es, de acuerdo con la OSI, una licencia de código abierto.

Ahora se analizará si la GPL es o no una licencia tipo *copyleft*. Como se ya se ha ido adelantando, esta licencia impone la obligación de que cualquier distribución de la obra original o derivada, sea bajo los términos de la licencia, lo que genera que la GPL se "esparza" hacia obras nuevas que ocupen todo o parte de una obra bajo la licencia. Este efecto se le ha denominado el "efecto viral" de la GPL, fenómeno que fue descrito por primera vez en el año 2000 por la profesora Margaret Radin, quien caracterizaba a los contratos virales como un intento de hacer correr las

[65] "The Open Source Definition". (Traducción propia)

obligaciones o la letra chica, con el objeto digital[66]. Este efecto viral es esencial para los objetivos de la GPL y este efecto no es más que la consecuencia de su carácter de *copyleft*. Aunque el término viral y copyleft son términos distintos, hay autores que los confunden y los usan de manera intercambiable[67]. Para efectos de este trabajo, *copyleft* y efecto viral no son intercambiables, *copyleft* es un tipo de licencia mientras que efecto viral es el fenómeno producto de ciertas licencias copyleft de extenderse a toda obra en la que se incorporan, en ese sentido, copyleft y efecto viral tienen una relación causa efecto. Entonces, la conclusión es que la GPL es una licencia copyleft, cuya redacción provoca un efecto viral.

Por último, se hará el análisis de si la GPL es una licencia de *software* libre. Para ello, ha de volverse a las cuatro libertades establecidas por la FSF, similar al análisis del código abierto, habrá que hacer un estudio pormenorizado de cómo se cumplen estas libertades.

La Libertad 0 para ejecutar el programa sin restricción se respeta desde el momento en que la GPL excluye el acto de correr el programa de su ámbito, en particular, no impone restricción alguna para ejecutar el programa, salvo que el resultado de un programa, por sus contenidos, constituya una obra derivada bajo los términos de la licencia.

La Libertad 1 de estudiar cómo funciona el programa y modificarlo, está estrechamente relacionada con su carácter de código abierto. A mayor abundamiento, la GPL es una licencia de código abierto desde el momento en que al suministrar, ya sea mediante distribución o puesta a disposición, una obra o una obra derivada de esta en forma de código binario u objeto debe acompañarse con el código fuente ya sea en forma física, en forma de una oferta escrita de mandar el código fuente o dejar el código fuente en algún lugar que sea accesible. Las distintas versiones de la GPL presentan diferencias en los medios específicos en que se puede cumplir está obligación, notablemente, la versión 2 de la licencia sólo permite dejar el código fuente en el mismo lugar que el código objeto, debiendo ser un acceso equivalente[68], esta diferencia clave se explica por el contexto histórico en 1991 cuando fue escrita esta licencia, pues en esa época el acceso a internet era mucho menos prevalente, por lo que aceptar una distribución online

[66] Margaret Jane Radin, "Humans, Computers, and Binding Commitment", *Indiana Law Journal* 75, nº 4 (2000): 1132. (Traducción Propia)
[67] En ese sentido, véase Guadamuz, Andres. "Viral contracts or unenforceable documents? Contractual validity of copyleft licenses". European Intellectual Property Review 26, nº 8 (1 de enero de 2004): 331–39. 334.
[68] "GNU General Public License, Version 2" (Free *Software* Foundation, junio de 1991), sec. 3.

hubiese significado que muchos destinatarios de la licencia no hubiesen podido acceder al código fuente[69].

La Libertad 2 de distribuir copias se refleja en la GPL permite que se redistribuyan copias del programa tal y como se reciben, en forma código fuente o código binario, siempre que se redistribuya bajo la misma licencia, se acompañe con una copia de esta y, en el caso de que se redistribuya en forma de código binario, se otorgue acceso al código fuente en cualquiera de las formas permitidas por la GPL. En cuanto a la copia de la licencia, se puede concluir que una copia electrónica de la licencia es permisible pero un enlace a una copia en línea es insatisfactorio para los propósitos de la GPL[70]

La Libertad 3 de distribuir versiones modificadas se expresa en que la GPL establece que, a rasgos generales, se puede distribuir copias de una obra derivada siempre que se distribuya bajo la misma licencia, se haga el aviso correspondiente de que la obra ha sido modificada y se respeten los términos de la licencia en cuanto a acompañar copia del texto de la licencia y acceso al código fuente, si corresponde. Si bien se puede ver estos requisitos como una barrera a la distribución de obras derivadas, hay que entender estos son indispensables para los propósitos de la GPL de mantener al *software* libre.

A grandes rasgos, la mayor parte de las disposiciones de la GPL se centra en los aspectos relacionados con estas cuatro libertades, siempre teniendo presente que la *Free Software Foundation* es la propietaria formal del Proyecto GNU. Fuera del ámbito de estas cuatro libertades, la GPL tiende a ser más laxa en su regulación o derechamente a excluir ciertas situaciones de su aplicación, no teniendo intenciones de ser una licencia totalizante, sino que solamente "*asegurarse que el software permanezca libre para todos sus usuarios*"[71]

Entonces el lato análisis anterior permite afirmar con seguridad que la GPL es una licencia de código abierto, copyleft y libre. Esto, más allá de las etiquetas que se le coloquen a la licencia, debe guiar la aproximación hacia el estudio de la licencia, en vista de que estas categorías demuestran una aproximación filosófico-política hacia la distribución y desarrollo de *software*.

[69] Neil Brown, "GNU GPL 2.0 and 3.0: obligations to include license text, and provide source code", *International Free and Open Source Software Law Review* 2, nº 1 (17 de junio de 2010): 10, https://doi.org/10.5033/ifosslr.v2i1.31.
[70] Brown, 9.
[71] "GNU General Public License, Version 2", sec. preamble. (Traducción Propia)

Conceptos de Derecho de Autor en la GPL

En esta sección se verá en mayor detalle el tratamiento de las obras derivadas bajo la GPL, para será necesario realizar un examen conceptual de términos de Derecho de Autor previo que permitan arribar a un análisis sistemático de la licencia. Este análisis partirá por abordar el concepto de Obra Derivada y de Distribución bajo esta licencia, ello permite ver que tan cerca o lejos están las definiciones de la GPL de las que da la normativa de Propiedad Intelectual, la interacción que la licencia prevé con la legislación nacional aplicable, además de ser un elemento para considerar en determinar si determinadas situaciones caen o no bajo obligaciones impuestas por la licencia.

Obra Derivada

En cuanto al concepto de Obra Derivada, hay que partir señalando que la GPL entrega su propia definición de Obra derivada que no necesariamente se alinea con la definición de la Ley Nacional que le aplique.

El marco de referencia será la definición de Obra derivada del Convenio de Berna:

> "*Estarán protegidas como obras originales, sin perjuicio de los derechos del autor de la obra original, las* ***traducciones, adaptaciones, arreglos musicales y demás transformaciones*** *de una obra literaria o artística*"[72]

La GPL en ambas versiones, hace remisión a la ley de Derecho de autor aplicable, por lo que ha de tenerse en cuenta la definición de la Ley 17.336, artículo quinto:

> *i) Obra derivada: aquella que resulte de la adaptación, traducción u otra transformación de una obra originaria, siempre que constituya una creación autónoma*

Como se puede observar, ambas definiciones tienen como elemento común referirse a transformación.

La definición de la GPL varía según la versión, por lo que hay que hacer un breve análisis de ambas. La versión 2 lee:

[72] Convenio de Berna, artículo 2 número 3 (énfasis agregado)

> "*'obra basada en el Programa' significa ya sea el Programa o cualquier obra derivada bajo la ley de Derecho de Autor: eso es, una obra que contiene el Programa o una parte de ella ya sea textualmente o con modificaciones y/o traducida a otro lenguaje*"[73] [74]

Esta definición en principio parece simplemente remitirse a la ley de Derecho de autor que aplique, pero la segunda parte define que constituye una obra derivada, siendo dicha definición distinta de las del Convenio de Berna o la ley 17.336, lo que puede generar confusión. En el caso de la GPL v2, la definición de obra derivada no se basa en la transformación, sino que de cierta forma apunta a la identificación con el contenido entre la obra original y la derivada. Lo anterior puede deberse a que, siendo la licencia en cuestión estadounidense, esté influenciado por las definiciones de Derecho de Autor de dicho país. Para la ley de *Copyright* de Estados Unidos, obra derivada es aquella basada en una o más obras preexistentes, en cualquiera de las formas en la que un trabajo puede ser refundido, transformado o adaptado[75].

En la GPL v2 se atisba un problema con la definición de "obra basada en el programa", ya que dado que la definición se basa en la identificación del contenido de la obra derivada con la original, se podría argumentar que una reproducción exacta de esta es a la vez una obra original y una obra basada en el programa, lo que carece de sentido. En este sentido, se atisba una diferencia clave entre la definición de la ley estadounidense y la de la GPL, mientras la primera usa la palabra refundir, la segunda usa la expresión contener el programa o una parte de ella. La palabra refundir (*recast*) tiene implícito un sentido de reorganizar algo o presentar de forma distinta[76], por lo que denota un uso más preciso que la palabra contener y que conlleva la realización de un acto en relación con la obra original, con tal de transformarla en derivada. Dando un ejemplo, si se reproduce un programa A en una copia B, entonces B contiene A pero no lo refunde pues no hay un sentido de organizar o presentar de forma distinta A, en cambio,

[73] "GNU General Public License, Version 2", sec. 0. (Traducción Propia)

[74] Hay que explicar a qué se refiere la GPL v2 con "traducido a otro lenguaje". Lenguaje en este contexto refiere a un Lenguaje de Programación, que, como se describió en el primer capítulo, es un lenguaje humano creado para escribir programas computacionales. Los lenguajes de Programación son múltiples, aunque las instrucciones o algoritmos pueden repetirse, entonces es posible crear dos programas computacionales idénticos en su función, aunque visualmente sean distintos sus códigos, similar a como dos lenguas humanas se ven y oyen distinto aunque signifiquen lo mismo. Esta situación permite que dos programas escritos en lenguas de programación diferentes, sean equivalentes funcionales

[75] U.S. Copyright Office, "Circular 92 Copyright Law of the United States and Related Laws Contained in Tıtle 17 of the United States Code", diciembre de 2022, sec. 101, https://www.copyright.gov/title17/. (Traducción Propia)

[76] "recast verb - Definition, pictures, pronunciation and usage notes | Oxford Advanced Learner's Dictionary at OxfordLearnersDictionaries.com", accedido 1 de enero de 2024, https://www.oxfordlearnersdictionaries.com/definition/english/recast?q=recast.

si se crea el *software* C copiando y pegando el programa D, además de contener otras obras o código propio, C contiene a D, a la vez que lo refunde, pues D ha sido reorganizado dentro de la estructura de C.

La GPL v3 en cambio usa esta definición:

> "*'Modificar' una obra significa copiar o adaptar todo o parte de la obra de una manera que requiera permiso de derechos de autor, que no sea una copia exacta. La obra resultante se llama 'versión modificada' de la obra anterior o una obra 'basada en' el trabajo anterior*"[77]

En esta versión de la licencia, se aparta del concepto de obra derivada, optando por la palabra modificar, que no corresponde a un concepto de Derecho de Autor propiamente tal. Por otro lado, es notable que la GPL v3 excluye crear una copia exacta, lo que es de toda lógica pues ello constituye reproducción y no transformación, además de dar una definición más enfocada en las acciones que se realizan para crear una obra derivada con respecto a la obra original, más que la identificación con el contenido, como lo hace la GPL v2. En cuanto a la remisión a la normativa del Derecho de Autor, existe una ligera variación de cambiar la remisión de la definición legal hacia las modalidades que requieran permiso, lo que en la ley 17.336 corresponde a las modalidades del artículo 18, pero este sólo menciona:

> "*Art. 18. Sólo el titular del derecho de autor o quienes estuvieren expresamente autorizados por él, tendrán el derecho de utilizar la obra en alguna de las siguientes formas:*
>
> *c) Adaptarla a otro género, o utilizarla en cualquier otra forma que entrañe una variación, adaptación o transformación de la obra originaria, incluida la traducción*"[78]

En general no añade ningún requisito en particular en relación con la GPL v3.

En conclusión, se puede observar que la GPL entrega definiciones de obra derivada que no son del todo claras, aunque estas en principio no parecen representar un obstáculo que impidiese definitivamente la interpretación correcta de la licencia. En relación con lo anterior, en ambas

[77] "GNU General Public License, Version 3" (Free *Software* Foundation, 29 de junio de 2007), sec. 0. (Traducción Propia)
[78] Ley 17.336, artículo 18 letra c.

versiones se descansa sobre un cierto nivel de remisión hacia la ley aplicable de Propiedad Intelectual, lo que permite remitirse a la normativa a fin de salvaguardar las deficiencias de redacción o llenar ciertos vacíos de las disposiciones.

Por motivos de claridad, en adelante se entenderá por "Obra Derivada" la definición de la Ley 17.336.

Distribución

Siguiendo con el concepto de **Distribución** y a modo de referencia, hay que tener en cuenta la definición de Distribución que hace la ley 17.336 en su artículo quinto:

> *q) Distribución: la puesta a disposición del público del original o copias tangibles de la obra mediante su venta o de cualquier otra forma de transferencia de la propiedad o posesión del original o de la copia*

La GPL v2 no define "Distribución" directamente, pero señala que se puede distribuir copias por "cualquier medio"[79] y también señala:

> "*Si la distribución de código ejecutable u objeto se hace ofreciendo para copiar desde un lugar designado, entonces ofreciendo acceso equivalente para copiar el código fuente desde el mismo lugar cuenta como distribución del código fuente, aunque los terceros no están obligados a copiar el código fuente junto con el código objeto*"[80]

Esta es una modalidad comúnmente usada hoy para poner a disposición programas de código abierto, típicamente manteniendo el código fuente y el código objeto en algún repositorio de *software* disponible en Internet, por lo que esta cláusula es necesaria por motivos pragmáticos.

Para efectos del análisis legal, cuando se aplica la legislación chilena, se confunden en los términos de la GPL v2 un acto de disposición a puesta del público con distribución, a modo de comparación, véase la definición de la Ley 17.336, artículo quinto:

[79] "GNU General Public License, Version 2", sec. 1.
[80] "GNU General Public License, Version 2", sec. 3.

> *v) Comunicación pública: todo acto, ejecutado por cualquier medio o procedimiento que sirva para* ***difundir los signos, las palabras, los sonidos o las imágenes****, actualmente conocido o que se conozca en el futuro, por el cual una pluralidad de personas, reunidas o no en un mismo lugar, pueda tener acceso a la obra* ***sin distribución previa de ejemplares*** *a cada una de ellas, incluyendo la puesta a disposición de la obra al público, de forma tal que* ***los miembros del público puedan acceder a ella desde el lugar y en el momento que cada uno de ellos elija****.*

Como se puede observar, esto es exactamente lo que ocurre cuando se coloca el código fuente junto a su código objeto en Internet: Se difunde los signos y palabras, al cual una pluralidad de personas puede acceder sin distribución previa de ejemplares a través de Internet, un medio que permite acceder a la página desde el lugar y momento que el público elija. Sin embargo, hay que hacerse cargo de una pregunta: ¿qué pasa cuando el usuario descarga el programa? Es ahí donde recién existe un acto de Reproducción, pues recién es ahí cuando el usuario de la página adquiere un ejemplar del programa computacional.

Similar a como ocurre con la definición de Obra Derivada, la GPL v2 se ve influenciada por el derecho estadounidense de Derecho de Autor al definir Distribución. Para la ley de *Copyright* de Estados Unidos, Distribución constituye un derecho exclusivo en los siguientes términos:

> "*Con sujeción a lo dispuesto en las secciones 107 a 122, el titular de derechos de autor en virtud del presente título tiene el derecho exclusivo de realizar y autorizar cualquiera de las siguientes acciones:*
>
> *(...)*
>
> *(3) distribuir copias o fonogramas de la obra protegida al público mediante venta u otra transferencia de propiedad, o mediante alquiler, arrendamiento o préstamo*"[81]

En general, no se encuentra en la ley de *Copyright* estadounidense una mención explícita al derecho de Disposición a puesta del público (según la jerga empleada en la versión oficial en inglés del Tratado de la OMPI sobre Derecho de Autor, "*Making available to the public*").

[81] U.S. Copyright Office, "Circular 92 Copyright Law of the United States and Related Laws Contained in Tıtle 17 of the United States Code", sec. 106. (Traducción Propia)

Guiada entonces por la legislación estadounidense, la GPL v2 no hace distinción entre Distribución y Disposición a puesta del público. En principio ello no representaría un problema grave, en tanto la *U.S Copyright Office* ha entendido que la legislación estadounidense integra el derecho exclusivo de Disposición a puesta del público mediante los derechos de Distribución, Ejecución Pública (*Public Performance*), Comunicación Pública (*Public Display*) y, en los casos en que la disposición a puesta del público implica la creación de una copia, el derecho de Reproducción[82].

Véase de nuevo el caso planteado anteriormente: se publica un programa en una página web, ya sea en forma de código fuente u objeto, ¿cómo se debe considerar incorporado el derecho de Disposición a Puesta del Público? Cuando se trata de disponer el programa en cuestión en la forma de ofrecer la descarga de este, la *U.S Copyright Office* sostiene la posición de que el derecho de Distribución cubre este supuesto, aunque esto no está exento de controversia[83].

A partir de lo anterior, se ve que la distinción entre Distribución y Puesta a disposición al público que hace el derecho chileno no existe como tal en Estados Unidos de América, no habiendo en su ley de Derecho de Autor un equivalente para la Puesta a Disposición. En este orden de ideas, se ve que, en este sistema, el concepto de Distribución es más amplio y cubriría los supuestos que la GPL v2 denomina como tal a lo largo del texto de la licencia[84].

Por las razones ya expuestas, se ve que existe un problema conceptual, donde las definiciones de la GPL v2 pueden diferir de las legales, dependiendo de qué ley de Derecho de Autor se aplique. Estas desalineaciones entre las definiciones de la GPL v2 y las legales no impiden la aplicación de la licencia, pero sí complican su análisis.

La versión tres de la GPL define un término similar a Distribución, aunque particularmente usa la palabra transmitir (*to convey*) y propagar (*to propagate*) en vez de distribuir (*to distribute*), definiéndolo de la siguiente forma:

> "*'Propagar' una obra significa hacer cualquier cosa que, sin permiso, te haría directa o secundariamente responsable por infracción bajo la ley de Derecho de Autor*

[82] U.S. Copyright Office, "The Making Available Right in the United States: A Report of the Register of Copyrights", 23 de febrero de 2016, 15, https://www.copyright.gov/docs/making_available/making-available-right.pdf.
[83] U.S. Copyright Office, 23–24.
[84] En particular, véase las secciones 1 a 7 de la GPL v2

aplicable, excepto ejecutarla en una computadora o modificar una copia privada. La propagación incluye reproducir, distribución (con o sin modificación), poner a disposición del público, y en algunos países otras actividades también.

Transmitir una obra significa cualquier clase de propagación que permite a terceros a hacer o recibir copias. Mera interacción con un usuario a través de una red de computadores, sin transferencia de una copia, no es transmitir"[85]

Esta definición de propagar es bastante amplia, por la forma en la que está redactada, aunque hace referencia a la ley de Derecho de Autor aplicable. En el caso en que se aplique la ley 17.336 se debe hacer referencia al catálogo de formas de uso de la obra del artículo 18, aunque bajo las disposiciones de la GPL v3, ha de hacerse con exclusión de los artículos que refieran a modificar la obra cuando es una copia privada.

En la definición de transmitir, esta hace referencia a una clase específica de propagación, que incluye la distribución y la puesta a disposición al público. Una de las dificultades de la GPL v3, es que la definición de Transmitir no especifica que sea al público, por lo que se puede plantear si ciertas formas de distribución, aunque no constituyen infracción a las leyes de Derecho de Autor, constituyen infracción a licencia[86], por ejemplo, esto hace que el concepto de transmitir de la GPL sea más amplio que el concepto de distribución de la ley 17.336 que se mostró al inicio de esta sección. La licencia responde a esta problemática, aunque de forma deficiente, en el segundo párrafo de la sección segunda, al permitir, sin condiciones, hacer, ejecutar o propagar obras cubiertas que no se transmitan, el problema es que la definición de transmitir es tan amplia, que la excepción descrita no aplicaría en prácticamente ningún caso. En este mismo párrafo, se aplica otra excepción que permite la transmisión de obras cubiertas para que terceros realicen modificaciones exclusivamente para quien las encargue y siempre bajo dirección y control de este, esta excepción parece modelada a lo que normalmente sería una relación laboral entre un desarrollador de *software* y su empleador.

El uso de la palabra "transmitir" no es accidental o provocada por un desconocimiento de los términos de Derecho de Autor, por el contrario, es a propósito. La conclusión anterior se colige

[85] "GNU General Public License, Version 3", sec. 0.
[86] Walden, "Open Source as Philosophy, Methodology, and Commerce", 24.

a partir de las Preguntas Frecuentes Sobre la Licencia GNU que la FSF tiene en su página web (en adelante las "FAQ"), donde se explica que el uso de Distribuir en la GPL v2 provenía de la legislación estadounidense, pero se cambió a un término que no se empleaba en ninguna ley de Derecho de autor del mundo con el fin de hacer sus propósitos lo más claros posibles donde sea que se aplique la licencia (#WhyPropagateAndConvey)[87][88]. La decisión de la GNU se entiende cuando se analiza que la definición de distribuir de la GPL v2 puede confundirse con la definición legal, cosa que de hecho ya ha ocurrido en este trabajo, cuando se señaló que esta definición de la licencia confunde distribuir y poner a disposición una obra bajos los conceptos usados por la ley 17.336 pero no bajo los usados por la ley estadounidense, en cambio, bajo la GPL v3 al usar una definición diferente a la legal, quien haga un análisis legal de la licencia tiene la carga de calzar los conceptos de la licencia con los de la legislación aplicable.

En síntesis, la GPL utiliza diversos términos para referirse a lo que esencialmente es Distribución y Puesta a disposición al público, confundiendo a veces esta última con Distribución. Si bien la GPL v2 no posee definición de distribución, bajo la ley chilena se confunden puesta a disposición al público con Distribución, mientras que la GPL v3 tiene una definición pero que usa términos que no se encuentran en las normas de Propiedad Intelectual, sin perjuicio de que tanto la GPL v2 y v3 hagan remisión a estas en ocasiones.

Debido a que las desalineaciones entre las definiciones de la GPL y las definiciones legales pueden complicar innecesariamente el análisis, es que, de ahora en adelante, en este trabajo el término "distribución" se refiere a la definición que le da la ley 17.336, primero para poder ir introduciendo conceptos formales de Propiedad Intelectual al análisis de la licencia y segundo, para tener una definición que sea más clara que las que provee la GPL.

Restricción a las obras derivadas

[87] "Preguntas frecuentes acerca de las licencias de GNU - Proyecto GNU - Free *Software* Foundation", accedido 20 de septiembre de 2023, https://www.gnu.org/licenses/gpl-faq.es.html.

[88] Nótese que cuando se cita las FAQ luego de la cita en sí hay un paréntesis que contiene un # y una frase en inglés. Esto denota las distintas secciones de la página web para mejor navegación, en estilo HTML. Estas secciones se encuentran en la página original y se dejarán para facilitar la búsqueda de las citas.

Ahora que ya se ha visto a grandes rasgos la GPL y las obras derivadas, es tiempo de analizar con mayor detalle las situaciones particulares en que se puede crear una obra derivada, en particular interesa tratar los siguientes casos:

A. Modificaciones Privadas: transformación sin Distribución o puesta a disposición
B. Distribución y puesta a disposición de modificaciones propias: Transformación con Distribución o puesta a Disposición
C. El caso particular de la nube y el SaaS: Interacción de un programa a través de una red sin distribución de binarios
D. Interacción entre el código ejecutable y otros programas: Transformación usando otros programas con Distribución y/o Puesta a Disposición
E. EI efecto virus y el problema de la Incompatibilidad: Transformación con programas bajo otra licencia con Distribución y/o Puesta a Disposición

Estas distintas modalidades han sido seleccionadas por el nivel de análisis que permiten realizar sobre las cláusulas de la GPL en distintas situaciones, teniendo como hilo conductor que cada uno agrega cierto nivel de complejidad, teniendo por objetivo tratar de sistematizar mejor las cláusulas de la GPL y darle a la vez un enfoque lo más didáctico posible. En las siguientes secciones se verán en detalle, tanto su regulación en la licencia como los aspectos más técnicos que involucran.

Modificaciones Privadas

Con modificaciones privadas se refiere a transformar la obra sin realizar acto alguno que bajo la ley 17.336 constituyese Distribución o puesta a disposición de la obra.

Las modificaciones privadas no tienen una regulación extensa en la licencia. La GPL es clara en que regula la modificación y por lo tanto requiere aceptarse la licencia para modificar, las obligaciones bajo la GPL respecto de las modificaciones sólo nacen con la distribución o la puesta a disposición del público de dicha modificación, ya sea mediante código fuente o código objeto.

Esta interpretación se apoya en las FAQ[89]:

> "*¿Exige la GPL que el código fuente de las versiones modificadas se ponga a disposición del público? (#GPLRequireSourcePostedPublic)*
>
> ***La GPL no le obliga a publicar el programa modificado, ni ninguna parte de este. Usted es libre de hacer versiones modificadas y usarlas en privado, sin tener nunca que hacerlas públicas.*** *Esto es aplicable también a organizaciones (empresas incluidas); una organización puede hacer una versión modificada y usarla internamente sin hacerla pública fuera de la organización.*
>
> *Pero si publica de alguna manera la versión modificada, la GPL le exige que ponga a disposición de los usuarios el código fuente modificado, bajo la GPL.*
>
> *Así pues, la GPL le autoriza a publicar el programa modificado de determinadas maneras y no de otras; pero la decisión de publicarlo o no depende de usted.*"[90]

Entonces tenemos que por el sólo hecho de mantener de realizar una modificación no surge una obligación de publicarlo, ni de poner a disposición el código fuente. Como ya se mencionó, la GPL sólo impone obligaciones al momento de realizarse la distribución o puesta a disposición del público, aun así es menester complementar la cita anterior con otra FAQ: en particular, si únicamente hago copias de un programa bajo la GPL y las ejecuto, sin difundirlas ni distribuirlas a terceros, la licencia no impone ninguna condición al respecto. (#NoDistributionRequirements)[91]

Si bien esta FAQ habla de "programas bajo la GPL" y no de obra modificada, las obras modificadas de todas formas, aun cuando no se distribuyan o se pongan a disposición del público, están bajo la GPL, dados los términos extensos que ocupa la licencia a la hora de definir lo que cubre y lo que no. Esto se hace más explícito en la GPL v3 cuando se menciona que sólo

[89] Las preguntas frecuentes de la FSF sobre las licencias GNU son una forma de interpretar las cláusulas más obscuras de las licencias en cuestión, a la vez que arrojan luz sobre la intención de ciertas redacciones o incluso cambios entre versiones. Aunque la FSF no sea la única que utilice estas licencias, es razonable asumir que quienes ocupan esta licencia en sus programas, de cierta forma adhieren a la visión de la FSF.

[90] "Preguntas frecuentes acerca de las licencias de GNU - Proyecto GNU - Free *Software* Foundation". (énfasis agregado)

[91] Ibid.

esta licencia otorga permiso para modificar cualquier obra cubierta, indicando, además, que realizar una modificación conlleva la aceptación de esta.[92]

El hecho de que la licencia hable de modificar o propagar es indicación de que ambas son acciones independientes en la aceptación de la licencia

En conclusión, cuando se modifica una obra cubierta, sin distribuir o poner a disposición se acepta la licencia, pero no nace ninguna obligación a partir de ella. Respecto de las posibilidades de explotación de un programa bajo la GPL, se puede afirmar que si un *software* cubierto o sus obras derivadas son explotadas en privado, como en el contexto de la realización de una función al interior de una organización o empresa, existen nulas restricciones por parte de la licencia.

Distribución y puesta a disposición de modificaciones propias

Los programas computacionales pueden modificarse de diversas maneras, pudiendo usarse partes de otros códigos fuentes o haciendo modificaciones originales. La fusión o combinación de programas es una interrogante tratada más adelante, siendo esta sección la encargada de analizar las transformaciones que surgen de ocupar el código original e insertar cambios surgidos a partir de la originalidad de quien las realiza, siendo estas las que aquí se denomina modificaciones propias.

El trato por parte de la GPL en general es bastante sencillo: cualquier modificación de una obra cubierta, queda regida por los términos de la licencia, lo que, por extensión, significa que cualquier obra derivada se rige por la GPL.

La pregunta se reduce a qué constituye modificación o cuándo una modificación constituye obra derivada, y aunque esto ya se mencionó anteriormente, es menester recordar que la GPL usa el estándar de lo que constituye obra derivada bajo la legislación aplicable (GPL v2) y modificación como aquello que requiere permiso de Derecho de Autor (GPL v3).

Por ello, en el caso hipotético de tener una obra derivada con modificaciones propias, el paso del uso privado a la Distribución o puesta a disposición del público, significa aceptar la carga

[92] "GNU General Public License, Version 3", sec. 9. (énfasis agregado)

que ello conlleva bajo la GPL, incluyendo pero no limitado a dejar a disposición al código, por lo que ya a esta altura se tiene un límite al tipo de comercialidad disponible para el programa.

El caso particular de la nube y el SaaS

Antes de poder analizar cómo la nube afecta la efectividad de la GPL, es necesario aterrizar un concepto abstracto como lo es la "nube". La computación en la nube se considera un modelo para asegurar un acceso generalizado, conveniente y a petición del usuario a un grupo de recurso computacionales[93], aunque más allá de cualquier definición concreta que se pueda dar, hay que decir en concreto que el usuario interactúa con estos recursos computacionales a través de la red, incluido cualquier programa computacional instalado en ellos. Es dentro de la computación en la nube que encontramos el *software as a service* (en adelante "SaaS"), un método de suministro de *software* que permite acceder a la información con cualquier dispositivo con conexión a internet y un navegador web[94].

En el contexto de la nube y el SaaS que se encuentra uno de los vacíos en las disposiciones de la GPL: si se suministra *software* otorgando acceso a través de la red, sin distribuir o poner a disposición código alguno, no se está cumpliendo con la definición de distribuir de la GPL v2 o de transmitir de la GPL v3. Por ello, la *Affero General Public License* (en adelante "AGPL") busca solucionar este problema "*que afecta a los desarrolladores de programas libres que se usan con frecuencia en servidores*"[95] agregando una nueva cláusula, en los siguientes términos:

> "*Si modifica el Programa, su versión modificada debe prominentemente ofrecer a todos los usuarios interactuando con este remotamente a través de un red computacional (si su versión admite tal interacción) una oportunidad de recibir el correspondiente código fuente de su versión proporcionando acceso al correspondiente código fuente desde un*

[93] P M Mell y T Grance, "The NIST Definition of Cloud Computing", 0 ed. (Gaithersburg, MD: National Institute of Standards and Technology, 2011), 2, https://doi.org/10.6028/NIST.SP.800-145. (Traducción propia)
[94] György Kratochwill, Jr., "What is the Difference between a *Software* License Agreement and a '*Software* as a Service' (SaaS) Agreement?", *les Nouvelles - Journal of the Licensing Executives Society* LVI, nº 3 (septiembre de 2021): 259.
[95] "Por qué es necesaria la GPL Affero - Proyecto GNU - Free *Software* Foundation", accedido 1 de noviembre de 2023, https://www.gnu.org/licenses/why-affero-gpl.es.html.

servidor de red sin coste, a través de algún medio estándar o habitual para facilitar la copia de software."[96]

De la redacción destaca que se refiere a versiones modificadas, ello tiene sentido porque si se ofrece en modalidad SaaS una obra bajo la GPL sin modificar, el *software* permanece libre, pues el código fuente está ya disponible en el programa original, en cambio si se ofrece de la misma manera una versión modificada de una obra bajo la GPL, haciendo uso de este vacío legal de la GPL original, una obra que era libre pasaría a estar oculta en su forma fuente, rindiéndola para efectos prácticos, no libre. A pesar de la existencia de la AGPL para responder a la problemática, no todo programa libre que sea potencialmente explotable vía la nube va a estar necesariamente bajo esta licencia, ello crea un conjunto de programas libres cuyas obras derivadas podrían ser explotadas bajo la nube sin consecuencia alguna.

En base a lo anterior, es relevante distinguir en el contexto de *software* ofrecido como SaaS, si el programa en la nube está bajo la GPL o la AGPL, pues sólo la segunda considera un caso especial para esta modalidad. Entonces, en cuanto a la explotación de un programa en la modalidad SaaS, si el programa está bajo la GPL, no surge obligación alguna, asemejándose al uso privado, pero si el programa está bajo la AGPL, entonces surge la obligación de dejar disponible el código fuente del programa a los usuarios. Ha de destacarse que, similar a la GPL, la AGPL impone la obligación de dejar el código fuente respecto de los usuarios, no del público general.

Interacción entre programas bajo la GPL y otros programas

Aquí es de interés referirse principalmente a tres formas de interacción entre programas:

a) Fusión de códigos fuentes

b) Enlaces dinámicos

c) Enlaces estáticos

Si bien en el caso de los libros no tiene mucho sentido cortar parte de un libro y pegarlo en otro (salvo que sea una cita textual o una compilación), en los programas puede tener pleno sentido

[96] "Affero GNU General Public License, Version 3" (Free *Software* Foundation, 19 de noviembre de 2017), sec. 13.

como forma de modificación de un programa computacional. La funcionalidad de un programa y sus respectivos algoritmos, no son excluyentes de otros. Entonces podemos tener un código fuente del programa A y del Programa B, para crear un nuevo código fuente del programa C, esto es lo que se denomina aquí “Fusión de Códigos”.

En cuanto a los enlaces dinámicos y estáticos, puede suceder que un programa requiera a otro para funcionar, esto es bastante común en la informática, de hecho, existen muchos programas cuyo objeto es realizar tareas que se requieren frecuentemente para diversas aplicaciones y que se reutilizan entre programas, ello es lo que usualmente se denomina bibliotecas (en inglés “*Libraries*”), las cuales suelen enlazarse a otros programas dinámica o estáticamente. En un enlace dinámico, si el programa A necesita un programa B, en este esquema los programas serían independientes y deben ser instalados por separado e interactúan cuando se ejecute el programa A, aplicando esto a la lógica de la literatura, el símil sería que un libro haga referencia a otro y el lector deba consultar el otro libro para que lo que está leyendo tenga sentido. En un enlace estático, estos programas se encuentran unidos en un solo archivo ejecutable a través de un proceso de enlazamiento, creando un archivo ejecutable que contiene ambos, de nuevo, en la lógica de los textos, un libro hace referencia a otro, pero ambos libros se encuentran físicamente unidos por lo que el lector puede entender el sentido del libro por sí solo. La diferencia esencial entre el caso de la literatura y el *software*, es que en el caso de los libros una falta de uno de los libros podría provocar una falta de sentido del mensaje, pero en los programas esto puede causar una falla al momento de ejecutarlo, por tanto, es una situación con mayores implicancias para su uso.

Habiéndose explicado estos conceptos, es tiempo de ver cómo se tratan estos casos bajo la GPL.

Primero véase el caso de la Fusión de código. Este caso es el más simple de los tres, como ocurre que se agregan distintos fragmentos de diferentes programas, se crea efectivamente uno nuevo, que es obra derivada de todos los programas que se hayan usado, por tanto, la GPL aplica en pleno a la obra.

En el caso de los enlaces dinámicos y estáticos, hay que referirse a las disposiciones de la GPL que tratan este problema en específico, ambas licencias son claras en que, si un programa es independiente y separado del programa bajo la GPL, entonces no está sujeto a la licencia,

aunque, por el contrario, si el programa está combinado con otro bajo la GPL, el programa completo queda bajo la licencia[97]. En principio parecería que esta disposición solucionaría el conflicto, pues en el enlace dinámico son obras separadas y en el estático son obras conjuntas, por tanto la GPL aplicaría solo en el segundo caso, sin embargo, el problema no es tan simple, dado que la FSF es de la visión que tanto en el enlace dinámico como estático aplica los términos de la GPL[98], mientras que la GPL v3 coloca a los subprogramas dinámicamente enlazados como parte de la definición de código fuente correspondiente[99]. Al respecto existe todavía una discusión sin una respuesta definitiva.

En el enlace estático existe un antes y un después, está programa A y B y luego del enlazamiento está el programa AB, sin embargo, dicho antes y después no existe en el enlace dinámico, este proceso es tan simple como un programa que hace un "llamado" a otro, el problema con ello es los programas no están insertos en el aire, sino que interactúan con innumerable cantidad de *software*, siendo el sistema operativo de una computadora el ejemplo más básico, si el concepto de enlace dinámico es demasiado amplio, podría incluir *software* del sistema operativo o programas que no tienen una relación más que accidental con el programa. Es en relación con este problema, que la GPL v3 cuando define Código Fuente excluye explícitamente bibliotecas del sistema, herramientas de uso general o programas libres[100] que se usan inalterados (*unmodified)* en realizar las operaciones pero que no sean parte de la obra[101], de la lectura de estas excepciones, se puede ver que la primera y la segunda sección no hacen mención a que deban ser programas libres, por lo que ha de asumirse que la excepción aplica cualquier programa, aunque sea de código cerrado. En esta misma línea, la GPL señala situaciones en las cuales no se extienden los términos de la licencia a otros programas, como la mera agregación de un programa cubierto y uno no cubierto en un medio de almacenamiento[102]

Por otro lado, tanto la GPL v2 como la GPL v3 incluyen al final de la licencia, una advertencia que señala que la GPL no permite incorporar la obra en programas de código cerrado y señalando que, si se considera más útil permitir enlazar programas de código cerrado, ha de

[97] Véase la GPL v2 sección 2 y la GPL v3 sección 5.
[98] "Preguntas frecuentes acerca de las licencias de GNU - Proyecto GNU - Free *Software* Foundation".
[99] "GNU General Public License, Version 3", sec. 1.
[100] La GPL v3 habla de "free programs", lo que se puede traducir como programas libres o programas gratuitos, dado que la palabra "free" se usa en otras partes de la GPL para referirse a Libre, se asumirá que en esta sección ese también es el caso.
[101] "GNU General Public License, Version 3", sec. 1.
[102] "GNU General Public License, Version 2", sec. 0.

usarse la GNU *Lesser General Public License* (en adelante "LGPL"). La LGPL es una licencia, parte de la familia GNU, cuya característica especial, es que permite enlazar el programa cubierto por esta licencia a un programa de código cerrado sin transmitir la obligación de dejar disponible el código fuente del programa computacional al que se enlaza. Antes de usar la LGPL, la GPL pide leer la página sobre porque no usar la LGLP, donde la FSF destaca que existen circunstancias estratégicas que hacen mejor usar la LGPL, siendo el caso más común "*cuando las características de la biblioteca libre están ya disponibles para software privativo a través de otras bibliotecas*"[103], lo que demuestra un interés de la GNU de promocionar bibliotecas libres aun con el costo de limitar la expansión de la licencia.

Ignorando la licencia que pueda cubrir a un programa que se enlaza a una obra cubierta, ha de decirse que el uso de enlaces estáticos resulta en que el programa externo enlazado quede bajo la GPL, mientras que en el caso del enlace dinámico no hay respuesta clara. Por ello, en la creación de obras derivadas, el uso de enlaces estáticos o dinámicos ha de ser manejado con cuidado para manejar adecuadamente las implicancias legales de su uso.

El efecto virus y el problema de la Incompatibilidad

Hasta ahora se ha visto el trato que la GPL le da a las obras derivadas en distintas modalidades y situaciones, en particular cuando se ha visto el uso de otros programas, ha sido ignorando qué sucede con las licencias que pueden afectar a otros programas.

La GPL es particularmente problemática como licencia pues, como se ha demostrado hasta ahora, cualquier modificación debe quedar bajo la GPL, ya sea una modificación propia o la incorporación de otro programa esté o no bajo otra licencia.

La unión de código bajo la GPL y código no licenciado o licenciado bajo la misma versión de la GPL es una situación simple, aplica las cláusulas de la GPL respectiva sin ningún problema, pero cuando se fusiona código bajo la GPL y código bajo otra licencia, se genera una situación que requiere de mayor análisis.

[103] "Por qué en su próxima biblioteca no debería utilizar la Licencia Pública General Reducida de GNU - Proyecto GNU - Free *Software* Foundation", accedido 8 de octubre de 2023, https://www.gnu.org/licenses/why-not-lgpl.es.html.

La fusión de dos programas bajo licencias distintas genera la necesidad de analizar la compatibilidad legal de las licencias[104], para ello se necesita además un criterio que permita identificar cuándo dos o más programas son compatibles, para ello, Bain, Gallego, Martínez y Rius entienden que un programa es compatible con otro bajo la siguiente condición:

> "*si se pueden mezclar o eventualmente interrelacionar sus códigos para crear una obra derivada compuesta de elementos de cada uno de ellos y distribuir el resultado de la integración sin infringir las licencias de uno y de otro, de manera que se pueda cumplir con las condiciones de ambas licencias a la hora de redistribuir el resultado.*"[105]

Si por el contrario, si no es posible redistribuir el resultado sin infringir al menos una de las licencias, se tiene dos licencias incompatibles.

La propia FSF es consciente del problema y lo trata en las FAQs, señalando algo similar a la cita recién hecha, pero haciendo una distinción relevante:

> "*En algunas licencias la compatibilidad puede depender de cómo se haga la combinación. Por ejemplo, podrían permitir vincular dos módulos, pero no fundir los códigos en un solo módulo.*"[106]

En general, "*las licencias con copyleft son incompatibles entre ellas, salvo pacto explícito*"[107], lo que incluye una incompatibilidad mutua entre la GPLv3, GPLv2 y la *Affero* GPL.

Para efectos de analizar la compatibilidad de una licencia con la GPLv3, es menester señalar que esta licencia establece en su sección 7 restricciones adicionales que pueden ser impuestas por el autor, 6 en particular son permitidas por la licencia, siendo cualquier otra removible por el licenciatario. Bajo la interpretación de Bain, Gallego, Martínez y Rius esto sería un nuevo mecanismo para ampliar el abanico de licencias compatible con la GPLv3[108].

---

[104] No confundir con la compatibilidad técnica de los programas.
[105] "Las Licencias Libres en la práctica", en *Aspectos legales y de explotación del software libre*, de Malcolm Bain et al. (Universitat Oberta de Catalunya, 2009), 16, http://hdl.handle.net/10609/229.
[106] "Preguntas frecuentes acerca de las licencias de GNU - Proyecto GNU - Free *Software* Foundation".
[107] "Las Licencias Libres en la práctica", 17.
[108] "Licencias de Software Libre", en *Aspectos legales y de explotación del software libre*, de Malcolm Bain et al. (Universitat Oberta de Catalunya, 2009), 36, http://hdl.handle.net/10609/229.

Dado lo anterior, es que el análisis de la compatibilidad de una licencia con la GPL es casuístico y tiene que ser analizado en base a los términos específicos de cada licencia. Sin perjuicio de que la FSF tiene en su página web su propio análisis de la compatibilidad de la GPL con otras licencias, hay múltiples autores que han hecho un análisis extensivo al respecto[109]. Por ello, el problema de la compatibilidad entre licencias es un problema al que no se le puede dar una respuesta definitiva, salvo que la GPL y su efecto viral determinan que, en ningún caso la incompatibilidad entre esta y otra licencia representa una exención o vía de escape a las cláusulas de la GPL.

[109] Véase en particular: Nayareth Guerra Gacitúa, "Licencias incompatibles de *software* libre", *Revista Chilena de Derecho y Tecnología* 0, nº 2 (3 de agosto de 2013), https://doi.org/10.5354/0719-2584.2013.22402. Al igual que: P McCoy Smith, "*Copyright*, Contract, and Licensing in Open Source", en *Open Source Law, Policy and Practice*, ed. Amanda Brock, 2ª ed. (Oxford University Press, 2022), 71–112, https://doi.org/10.1093/oso/9780198862345.003.0003. Página 99, tabla 3.2.

3. Acerca del reconocimiento de las estipulaciones de la GPL en el derecho chileno

Este tema es crucial a efectos de este trabajo ya que, finalmente, las estipulaciones de la GPL son sólo relevantes en la medida en la que haya un tribunal esté dispuesto a hacer valer sus disposiciones. Dada la escasez de doctrina y jurisprudencia chilena al respecto, este se considera un tema no resuelto y que tampoco no se espera resolver aquí, por lo que se hará una exposición breve al respecto.

Las licencias *Copyleft* en general representan un desafío al implicar un cambio de paradigma respecto del *Copyright*, por lo que es de esperar que exista al menos una cierta inadecuación de la jurisprudencia y las leyes respecto a este fenómeno.

La autorización de uso de obra ajena en la ley 17.336 y la GPL

Bajo la ley 17.336, la autorización de obra ajena debe otorgarse de manera expresa según consta en el artículo 19 y complementada por el artículo 20. En las licencias *copyleft* no hay problemas con el artículo 19 desde que la autorización se manifiesta en la forma de acompañar un texto señalando los términos o señalar qué licencia rige a la obra, pudiendo el recipiente acceder a los términos de la licencia, lo que constituye autorización expresa. El artículo 20 requiere mayor análisis, ya que este artículo determina la forma específica que debe tener esta autorización. El artículo 20 lee:

> *Artículo 20.- Se entiende, por autorización el permiso otorgado por el titular del derecho de autor, en cualquier forma contractual, para utilizar la obra de alguno de los modos y por alguno de los medios que esta ley establece.*
>
> *La autorización deberá precisar los derechos concedidos a la persona autorizada, señalando el plazo de duración, la remuneración y su forma de pago, el número mínimo o máximo de espectáculos o ejemplares autorizados o si son ilimitados, el territorio de aplicación y todas las demás cláusulas limitativas que el titular del derecho de autor imponga. La remuneración que se acuerde no podrá ser inferior, en caso alguno, al porcentaje que señale el Reglamento.*

A la persona autorizada no le serán reconocidos derechos mayores que aquellos que figuren en la autorización, salvo los inherentes a la misma según su naturaleza.

En base al tenor literal del artículo, se tiene que una autorización debe ser dada de forma contractual, además, el segundo inciso establece las menciones mínimas obligatorias de la autorización, deduciendo que estas son obligatorias y no meramente ejemplares, por el uso de la palabra "deberá".

Haciendo el listado de requisitos tenemos que la autorización:

1. Debe ser darse de forma contractual.
2. Debe contener:
    a. Plazo
    b. Remuneración y forma de pago
    c. Número de ejemplares o si son ilimitados
    d. Territorio de aplicación
    e. Otras cláusulas limitativas

La GPL como contrato

Para analizar si la GPL es un contrato, hay que inevitablemente recurrir a la noción de contrato del derecho civil. El código civil define contrato en el artículo 1438 como "*un acto por el cual una parte se obliga para con otra a dar, hacer o no hacer alguna cosa*", en cuanto a sus requisitos, de acuerdo con Abeliuk, un contrato debe reunir los siguientes:

"*1º. El acuerdo de voluntades de dos o más personas, y*

2º. Que este acuerdo tenga la intención de crear obligaciones."[110]

Respecto del segundo requisito no hay duda alguna en cuanto la GPL impone múltiples obligaciones, pero respecto del primer punto se requiere mayor desarrollo. La pregunta principal es si recibir una copia de la licencia es suficiente para aceptarla. Ha de mencionarse que la GPL no requiere aceptación inmediata previa a tener una copia del programa o incluso usarlo, la GPL explícitamente excluye tener una copia y usarla de los términos de la licencia. La aceptación

[110] René Abeliuk Manasevich, *Las obligaciones. T. 1*, 5. ed. actualizada (Santiago, Chile: Ed. Jurídica de Chile, 2010), 59.

sólo se da una vez que se redistribuye una copia, se modifica o se distribuye la obra derivada de un programa cubierto por la licencia, por lo que la licencia no debe ser aceptada por gran parte del público general y aquellos que sí deban hacerlo, presumiblemente tienen cierta experiencia que les permite entenderla mejor. En cuanto a los requisitos para el consentimiento, la voluntad de las partes puede manifestarse de cualquier forma, incluido a través de hechos, signos o actitudes[111], por lo que el método de la GPL de aceptar la licencia de modificarse, distribuir una modificación o redistribuir el programa cubierto es una forma válida de obtención del consentimiento del recipiente de la licencia.

Sin perjuicio de la conclusión anterior, existe un problema respecto de la formación del consentimiento. En la doctrina, por falta de una regulación del código civil, se han establecido esencialmente dos doctrinas, la de la declaración y la del conocimiento, Ducci la describe en los siguientes términos:

> "*De acuerdo con el art. 101 del Código de Comercio el consentimiento se forma en el momento en que el aceptante manifiesta su aceptación. Dicho artículo dice que el contrato queda en el acto perfeccionado. Esta es la que se llama teoría de la declaración. Otra teoría sostiene que entre ausentes el consentimiento sólo se forma cuando el proponente conoce la aceptación. Tal teoría se llama del conocimiento.*"[112]

La aplicación de disposiciones del código de comercio es relativamente pacífica, en cuanto autores y jurisprudencia están de acuerdo en se aplican por analogía ante el silencio del código civil[113].

En cuanto a la elección entre una teoría u otra no es indiferente en cuanto a determinar si la GPL es un contrato válido o no. En la distribución moderna de *software* donde múltiples personas pueden descargar un mismo programa licenciada, la aceptación puede darse por múltiples medios electrónicos, que varían desde simplemente presionar un botón aceptando los términos y condiciones a medios más complejos como la Firma Electrónica Avanzada, en cambio, que el licenciatario conozca la aceptación es más complejo por una miríada de razones, como el alto volumen que puede suponer conocer de todas las licencias que supone un programa

[111] Arturo Allessandri Rodríguez, *De los contratos* (Chile: Jurídica de las Americas, 2009), 72.
[112] Carlos Ducci Claro, *Derecho civil: parte general*, 4a ed (Santiago de Chile: Jurídica de Chile, 2007), 252.
[113] Allessandri Rodríguez, *De los contratos*, 88.

computacional disponible en Internet o la falta de suficiente comunicación con el recipiente de la licencia para conocer de su aceptación. Entonces bajo la teoría de la aceptación, la GPL como contrato no tendría muchos problemas en su formación, pero de aceptarse la teoría del conocimiento, la GPL no se formaría como contrato sino en contadas ocasiones.

Sin perjuicio de que el problema de la formación del consentimiento requiere un análisis más pormenorizado que el que se ofrece aquí, en este trabajo se estima que la aceptación de las disposiciones del código de comercio para regular la formación del consentimiento, forzosamente conllevan a concluir que ha de aplicarse la teoría de la aceptación, por tanto, no se ve mayor obstáculo para la validez de la GPL en cuanto contrato.

Por último, un problema que resulta apropiado mencionar en esta sección, es la que da Guadamuz, respecto de si las cláusulas de la GPL son válidas en cuanto traspasan obligaciones a terceros que no forman parte del contrato. El problema es más aparente que real, siendo el propio Guadamuz quien entrega la respuesta, explicando que la validez contractual del copyleft funciona en base de uno a uno donde sólo las dos partes involucradas pueden demandar el cumplimiento[114]. Las obligaciones bajo las licencias copyleft no se plantean en la forma de obligar a terceros a respetar las cláusulas de la licencia, sino que se plantean en la forma de obligar a quien recibe una obra cubierta a aplicarle a terceros la misma licencia. La consecuencia de lo anterior es que, si de una obra original se crea una obra derivada X y de esta última una obra derivada Y, sólo el autor de la obra derivada X podría demandar el incumplimiento, no habiendo cabida para que haya una demanda por parte del autor original. Para evitar una situación así, la FSF en sus propios proyectos usa la figura de asignaciones de *Copyright*, para lo cual, cualquier contribuyente al proyecto, debe asignar sus derechos de autor a la FSF para que esta pueda encargarse de hacer cumplir la GPL[115].

A modo de conclusión, la GPL se estima es válido como contrato para el derecho chileno, por cuanto: a) cumple con los requisitos que se exigen para ser considerado un contrato; b) bajo la

[114] Andres Guadamuz, "Viral contracts or unenforceable documents? Contractual validity of copyleft licenses", *European Intellectual Property Review* 26, nº 8 (1 de enero de 2004): 337.

[115] "Por qué la FSF solicita a sus colaboradores la asignación del copyright - Proyecto GNU - Free Software Foundation", accedido 27 de noviembre de 2023, https://www.gnu.org/licenses/why-assign.es.html.

teoría dominante en la jurisprudencia, es capaz de formarse el consentimiento para su validez y; c) sus cláusulas no imponen obligaciones a terceros que podrían afectar su validez.

Cumplimiento de las menciones esenciales

La GPL no cumple con todas las menciones requeridas del artículo 20, en particular, ninguna versión de la GPL menciona expresamente plazos, números de ejemplares o sin ilimitados o territorio de aplicación. Se puede argumentar que la GPL las menciona de manera implícita, pero queda la duda de si las menciones esenciales del artículo 20 deben ser expresas o si pueden ser tácitas, Sierra y Bernet, a partir de la historia fidedigna de la Ley interpretan lo siguiente:

> "*según la historia de la Ley, se disponía que la autorización debía otorgarse: i) contractualmente —lo que impide considerar como autorización el acto unilateral de una de las partes—, y ii) que tal autorización debía tener un contenido mínimo esencial.*"[116]

Es a partir de ello que los autores concluyen que en la autorización es "*necesario que se detalle un conjunto de elementos que se consideran como esenciales*"[117]. Entonces, en base al tenor literal y a la historia fidedigna de la ley, que la GPL no sería una autorización válida acorde al artículo 20 de la ley 17.336.

No obstante, la explicación anterior se considera insatisfactoria, por cuanto no consta en la historia de la ley cuales fueron las razones particulares del legislador para el establecimiento de los requisitos específicos del artículo 20 inciso segundo, siendo insuficiente la referencia que realizan los autores recién citados a la obra de Henry Jessen como influyente en el proyecto, sin que constaten las razones del legislador de redactar la norma en un sentido más restringido que lo hace Jessen en su propia obra.

Más tangible es el objetivo que la Ley 17.336 declara en su artículo primero, de proteger los derechos de los autores de obras de la inteligencia. Si el objetivo es proteger al autor, no se

[116] Manuel Antonio Bernet y Alfredo Sierra, "La autorización de utilización de una obra ajena en la Ley de Propiedad Intelectual chilena", *Revista Chilena de Derecho y Tecnología* 8, nº 1 (30 de junio de 2019): 124, https://doi.org/10.5354/0719-2584.2019.52647.
[117] Bernet y Sierra, 125.

atisba razones para que la ley niegue valor a una licencia otorgada libremente por el autor. La GPL se usa como autorización por mera voluntad del autor, que se distribuye con el programa sin mediar negociación de ningún tipo, en un contexto en que el autor no se encuentra en una asimetría desfavorable en relación con quienes reciben el programa. Sumado a ello, un problema práctico que surge de no reconocer validez a la GPL por no cumplir algún requisito del artículo 20, es que no habiendo autorización de uso de obra ajena válida bajo la ley chilena, cualquier tercero que quiera usar la obra, no se encontrará autorizado por la sola referencia a la GPL dentro de ella y en cambio aplicará el derecho exclusivo del autor, derrotando uno de los propósitos de usar una licencia general. Este problema se agrava en el contexto del tráfico del internet, donde puede existir problemas con obtener del autor una licencia adecuada a los requerimientos del derecho chileno, ya que no necesariamente existe un contacto directo con el autor y pueden barreras adicionales, como las lingüísticas.

Otro esquema propuesto ha sido el de Schuster, para quien el sistema de Derecho de Autor se construye sobre las siguientes normas:

> "− *Una norma general prescriptiva que prohíbe todo acto que implique el uso de una obra ajena, sin haber obtenido una autorización del titular o de la ley. La acción prohibida es toda explotación de una obra intelectual, cualquiera sea la forma o modo de explotación y cualquiera sea la finalidad del agente al explotar la obra, incluidas finalidades lucrativas y no lucrativas.*
>
> − *Una norma general habilitante, que otorga un poder al autor o al titular del derecho para autorizar el uso de una obra intelectual que se llamará autorización o permisión convencional*
>
> − *Una norma general permisiva que se llamará autorización o permisión legal, de contenido disyuntivo, dado que se requiere identificar el contenido preciso de las acciones de uso de obras intelectuales ajenas, en un material legal disperso*"[118]

[118] Santiago. Schuster Vergara, "Las excepciones al derecho de autor como normas permisivas: una revisión crítica del derecho de usuario en el sistema normativo de derecho de autor" (Tesis para optar al grado de Doctor en Derecho, Santiago de Chile, Repositorio Universidad de Chile, 2019), 152.

Bajo esta estructura, la norma general prescriptiva sería la del artículo 19[119], mientras que serían los artículos 17 y 20 de la ley 17.336 los "*declaran el poder jurídico del autor y prescriben sobre la forma en que se entenderá concedida la autorización*"[120]. Esta última cita pareciese ir en la misma línea que Bernet y Sierra, sin embargo, bajo la concepción de Schuster el artículo 17 de la ley tomaría la forma de cláusula general, que constituye en el titular de los derechos de autor un poder de autorización"[121], en tanto, el artículo 20 complementa lo dispuesto por el artículo 17[122], lo que conlleva a que el rol del artículo 20 esté subordinado al poder de autorización que la norma habilitante concedida al titular de los derechos de autor. Esta subordinación implica que hay que interpretar sus contenidos en relación a la norma general habilitante. Schuster deja entrever dicha forma de interpretación al analizar la necesidad de señalar al remuneración bajo el artículo 20:

> "*El artículo 20 de la LPI establece los elementos que debe contener la autorización y prevé entre éstos la remuneración que deberá acordarse, lo que indica que no puede presumirse la gratuidad al autorizar. Pero la contraprestación pecuniaria no es un elemento esencial del permiso de uso, ya que el poder de autorización del autor, cuyo fundamento se encuentra en el articulo 17 LPI, no se encuentra sujeto a restricciones. El autor puede conceder el permiso de uso sin otro requisito que la descripción de la explotación permitida, que de no existir implicaría que la autorización estaría vacía de contenido*"[123]

El esquema recién expuesto implica que la GPL es una autorización válida en tanto se fundamenta en la norma general habilitante y, por supuesto, no está vacía de contenido. La Falta de mención de ciertos elementos especificados en el artículo 20 implica un eventual vacío que debe ser resuelto por el juez, pero que no le niega validez a la GPL en cuanto condiciones de uso que el titular ha impuesto para conceder la autorización de uso de la obra, en virtud del poder que le confiere el artículo 17.

[119] Schuster Vergara, 73.
[120] Schuster Vergara, 168.
[121] Schuster Vergara, 169–70.
[122] Schuster Vergara, 171.
[123] Schuster Vergara, 67.

Un elemento impredecible: El rol de los tribunales

En la sección anterior se estableció que la GPL: 1) es un contrato válido, 2) es una autorización expresa, 3) no cumple con las menciones esenciales exigidas por la ley 17.336 y, 4) debe ser considerado una autorización válida bajo la ley chilena. A pesar de ello, el razonamiento de los tribunales a la hora de analizar un caso que involucre la GPL o, en realidad, cualquier licencia *copyleft* puede variar de manera insospechada.

En cuanto a ser un contrato, en un tribunal español se negó validez a una licencia *Creative Commons* porque el tribunal estimó que la licencia "*no pasa de constituir un mero folleto informativo acerca del contenido de la misma y asimismo carente de toda firma, no pudiendo, por ende, serle atribuido ningún valor*"[124], decisión que en el contexto de la legislación española de Derecho de autor resulta extraña, por cuanto la legislación española define que la autorización de uso de obra ajena puede darse por cualquier medio. En el contexto de la legislación chilena que exige que la autorización debe ser contractual, bien podría algún tribunal negarle el estatus de contrato válido a la GPL.

Por último, en cuanto al cumplimiento de las menciones esenciales requeridos por el artículo 20, la jurisprudencia de la Corte Suprema ha sido ambivalente. En el caso Gómez Fuentes Octavio Alberto Con Empresa Nacional de Minería, la Corte Suprema rechazó el recurso de casación interpuesto por el demandante, con un voto en contra del ministro Blanco. El voto de mayoría en este caso admite que la autorización puede expresarse a través de actos unilaterales del titular sin cumplir con el contenido esencial del artículo 20 mientras que el voto de minoría estimaba que el contenido de dicho artículo es imperativo para las partes[125]. En un fallo más reciente la Corte Suprema, sin voto de minoría, expresó que la autorización de uso de obra ajena "se cumple exclusivamente bajo los parámetros impuestos en el inciso segundo del artículo 20 en análisis, sin que sea posible admitir la existencia de una autorización tácita"[126]. En este último caso, no se observa una decisión que esté dirigida a definir el rol del inciso segundo del artículo 20 en cuanto menciones esenciales, sino que sólo confirma que no admite la autorización tácita,

[124] SAP PO 678/2005 (Audiencia Provincial de Pontevedra 29 de noviembre de 2005).
[125] Bernet y Sierra, "La autorización de utilización de una obra ajena en la Ley de Propiedad Intelectual chilena", 123.
[126] Rol 131673-2020 (Corte Suprema 3 de agosto de 2022).

cuestión pacífica en cuanto ello se encuentra establecido en el artículo 19 de la misma ley, por ello, no es conclusiva a establecer un lineamiento jurisprudencial al respecto.

En síntesis, de la jurisprudencia de la Corte Suprema, no es posible extraer un criterio constante usado por el máximo tribunal para caracterizar el rol del artículo 20 en las controversias de Propiedad Intelectual, por lo que, sin perjuicio de lo dicho en la sección anterior, esta parte de la investigación ha resultado inconclusa.

Los límites del Derecho de autor

Sin perjuicio de que haya extensa bibliografía al respecto, es menester mencionar brevemente los límites del Derecho de Autor en cuanto a qué es protegible por este, partiendo por la dicotomía expresión/idea, para luego introducir una doctrina más atingente a los programas de computación, la *merger docrtne*.

Es un principio general del Derecho de Autor que su protección "abarcará las expresiones pero no las ideas, procedimientos, métodos de operación o conceptos matemáticos en sí"[127] tal como se declara en el Tratado de la OMPI sobre Derecho de Autor. En este sentido es que se define que la dicotomía expresión/idea se refiere a la distinción entre la expresión de una idea y, valga la redundancia, la idea misma. Aplicado lo anterior a programas computacionales, hay que recordar que lo que típicamente se denomina algoritmo, es un procedimiento lógico y/o matemático para obtener un resultado, por ello cuando se protege el *software*, se protege la forma de expresión de dicho algoritmo en la forma de código (ejecutable o fuente), pero no el algoritmo en sí mismo.

Dentro de la dicotomía expresión/idea, surge lo que se denomina la *merger doctrine*, que en palabras simples, es una doctrina de la jurisprudencia estadounidense respecto del Derecho de Autor, en el cual los tribunales estiman que una idea se ha fusionado (merge) con la expresión, lo que, basados en el principio de que no se protegen ideas, la haría perder su protección bajo el Derecho de Autor[128]. Basándose en que los programas computacionales utilizan lenguajes

[127] OMPI, "Tratado de la OMPI sobre Derecho de Autor", art. 2.
[128] Pamela Samuelson, "Reconceptualizing *Copyright*'s Merger Doctrine", *Journal of the Copyright Society of the U.S.A.* 63 (4 de diciembre de 2016): 1, https://ssrn.com/abstract=2763903.

usualmente más restringidos en la posibilidad de expresión, las lenguas de programación, y que típicamente el programa buscará plasmar un algoritmo, la *merger doctrine* se torna más necesaria cuando se analiza la protección del *software* bajo el Derecho de Autor.

En la práctica esto tiene consecuencias relevantes, en base a la jurisprudencia estadounidense que aplica este principio de separar idea y expresión, Peter Menell señala que "*mientras que programar una computadora puede incuestionablemente ser considerado "creativo" en un sentido general, las doctrinas limitantes aseguran que los aspectos funcionales no estén protegidos por la ley de Derecho de Autor*"[129], entre estas doctrinas a las que refiere el autor, se encuentra la dicotomía expresión/idea y la *merger doctrine*, mientras que los aspectos funcionales, bajo el relato de este autor, refieren principalmente a medidas de interoperabilidad de programas computacionales, sin perjuicio de extender a otros aspectos. El uso de la *merger doctrine* como defensa en casos sobre programas computacionales han tenido un éxito considerable, frecuentemente referido a algunos aspectos del programa, no la obra completa[130].

La pregunta es, ¿podría aplicarse esta doctrina en el derecho chileno? Para responder esta pregunta se debe partir respondiendo si esta doctrina es incompatible con la tradición continental del Derecho de Autor, o si es sólo aplicable a la tradición anglosajona del *Copyright*. Ciertamente, hay figuras legales de la tradición anglosajona que no pueden ser trasladados a tradición continental, como la doctrina del Fair Use, pero ello no es el caso de la *merger doctrine*. Esta doctrina se construye a partir de un elemento en común que tienen ambas tradiciones, que es el principio de no extender la protección a las ideas, por ello, debe ser aplicable. Dentro de la tradición continental, se puede ver un atisbo de apoyo a esta doctrina, en la teoría del plagio de Elba Lopéz, para quien "es un error considerar la separación absoluta entre la idea y su forma de ser expresada porque aquella, al ser materializada, queda indisolublemente unida a esta última"[131]. Entendiendo la dicotomía expresión/idea como un rango entre mayor o menor separación de las ideas con la expresión, más que dos elementos inseparables, permite crear un marco de análisis para la aplicación de la *merger doctrine*. Así, los programas computacionales no serían más que obras que tienen una alta potencialidad de

[129] Menell, "Economic Analysis of Network Effects and Intellectual Property", 317.
[130] Samuelson, "Reconceptualizing Copyright's Merger Doctrine", 11–12.
[131] Elba López Fernández, "El plagio: la dicotomía entre la idea y la expresión de la idea", *RICSH Revista Iberoamericana de las Ciencias Sociales y Humanísticas* 9, nº 17 (8 de enero de 2020): 330–50, https://doi.org/10.23913/ricsh.v9i17.201.

estar en el mayor extremo de este rango, donde la expresión y la idea son inseparables, de tal forma que la idea sólo puede expresarse en una forma o en una cantidad ínfima de formas[132].

Por ello, ha de estimarse que la *merger doctrine* puede aplicarse en el derecho chileno, en aquellos casos en que la expresión y la idea son inseparables, de tal forma que una misma idea sólo puede ser expresada en acotada varias de formas.

Aterrizando lo dicho anteriormente al análisis de la GPL, ha de destacarse que esta se encuentra afecta a estos límites. Ello determina que de los programas bajo la licencia, son de libre circulación los algoritmos que estos contienen, en tanto no se generen situaciones que pueda considerarse plagio o uso no autorizado de la obra. En determinadas ocasiones, si una expresión de un programa cubierto fuese inseparable de la idea, se estaría ante una parte no protegida de la obra, sin embargo, el análisis de dicha situación es casuístico, sin poder dar de antemano criterios suficientes para determinar cuándo se está ante una situación de este tipo, sin perjuicio de las guías generales que se pueden reconocer. A modo de primer acercamiento, las funciones que cumple la *merger doctrine* sirven para determinar los casos en que se aplica, de aquellas que Samuelson reconoce en la jurisprudencia estadounidense, se estima particularmente interesante para el análisis de la GPL, aquella referida a "*evitar variaciones innecesarias y permitir eficiencia y estandarización*"[133], bajo esta función, las cortes deben ponderar si la decisión de un demandante de usar una expresión particular de una idea fue lógica, razonable, practica, habitual o tradicional, útil o efectiva[134], en relación con la eficiencia, reconoce la autora señala la autora:

> "*Consideraciones en cuanto a la eficiencia han tenido alguna influencia en la gama de expresiones alternativas viables. Esto ha sido ampliamente considerado en casos de Derecho de Autor de software, pero la eficiencia como límite a las opciones de diseño para el autor ha sido reconocido hace bastante tiempo. La eficiencia es, por supuesto, un tipo de funcionalidad que la ley de derechos de autor ha excluido convencionalmente*

[132] Si bien puede existir una visión de la *merger doctrine* como aquella que aplica cuando existe sólo una forma de expresar la idea, Samuelson desmiente esto indicando que: "Sin embargo, se trata de una opinión claramente minoritaria en la jurisprudencia en los casos de *merger*; la opinión que prevalece actualmente, aunque no sea universalmente aceptada, es que el *merger* puede y debe constatarse cuando existe un número limitado de formas alternativas de expresar determinadas ideas, hechos o funciones." Samuelson, "Reconceptualizing Copyright's Merger Doctrine", 9.

[133] Samuelson, 44. (Traducción Propia)

[134] Samuelson, 44–45.

> *de su ámbito de aplicación. Si un primer programador adopta la forma más eficiente de estructurar el software para realizar ciertas tareas, los programadores posteriores no deberían verse obligados a utilizar diseños inferiores. La merger doctrine sirve para permitir la reutilización de diseños eficientes en obras protegidas por derechos de autor*."[135]

La eficiencia puede ser de suma importancia en la creación de un programa computacional, en tanto el *software*, por su esencia, está pensado para ser usado en una computadora, la que cuenta con recursos limitados (procesador, memoria, almacenamiento, etc.), factor que puede ser limitante a las opciones que el programador tenga para crear el programa computacional.

Por último, Samuelson reconoce que la *merger doctrine* ha sido usada de manera exitosa en aquellos casos en que surge un estándar de la industria, incluyendo aquellos en que un trabajo original se transforma posteriormente en un estándar de hecho, aunque en estos últimos la jurisprudencia ha sido más ambivalente[136]

En base a lo anterior, se tienen ciertas herramientas para el análisis casuístico a fin de determinar la posible aplicación de la *merger doctrine* en casos de *software*. Por extensión, las obras bajo la GPL pueden contener partes que no estén sujetas a protección de Derecho de Autor en virtud de la *merger doctrine*, si hay consideraciones de eficiencia o estandarización de hecho que así lo justifiquen.

Alternativamente a la *merger doctrine*, en el derecho continental es posible usar la metodología de análisis de François Pellegrini:

> "*Por lo que a la manera cómo resolver los litigios en materia de falsificación se refiere, de lo que se trata en suma no es tanto de juzgar las similitudes de forma observables, sino de los medios que condujeron a autores diferentes a producir esas obras y de la parte que la copia conforme haya podido desempeñar en la similitud*"[137]

[135] Samuelson, 45.
[136] Samuelson, 45–46.
[137] Pellegrini, "ORIGINALITY OF SOFTWARE WORKS", 68.

Usando este criterio, la falsificación en programas computacionales debe centrarse sólo en las similitudes entre códigos y las razones propias de estas[138]. Así, los aspectos que hasta ahora se han denominado funcionales, no pueden ser considerados falsificación, en tanto las razones que dictan su inclusión no se basan en la expresión de la personalidad del autor, sino que en la eficiencia, el ánimo de seguir un estándar de la industria, la lógica, etc. De cierta forma, este análisis llega a las mismas conclusiones que la *merger doctrine* aunque parta de una premisa diferente. Así el autor recién citado concluye que, en los programas computacionales, en concreto no han de considerarse originales aquellos elementos que refieren a convenciones de interacción entre el módulo de un programa y otro (denominados "API")[139], aunque aquí se estima que los aspectos funcionales no se deben limitar de antemano sólo a las APIs de un programa.

Este segundo análisis, que sigue una ruta diferente a la de la *merger doctrine*, llega a conclusiones similares a las de Peter Menell de que, en caso alguno, los elementos funcionales de un programa computacional, como las interfaces de interacción, están protegidos por el Derecho de Autor.

Con dos análisis diferentes, uno proveniente del *Common Law* y otro del Derecho Continental, se ha llegado a la conclusión de que los aspectos funcionales de un programa computacional no reciben protección del Derecho de Autor, por lo que parece algo transversal a los Sistemas de Derecho de Autor. Por extensión, es posible afirmar que las secciones de una obra bajo la GPL dictaminadas por aspectos funcionales en general no reciben protección del Derecho de Autor, no pudiendo quedar estas bajos las restricciones que la licencia impone, en tanto corresponden a ideas de libre circulación.

[138] Pellegrini, 87–88.
[139] Pellegrini, 82.

Conclusiones

El desafío propuesto por la GPL en cuanto a su análisis legal resulta complejo, aun así, como licencia esta subsiste en virtud del Derecho de Autor, aunque lo sea como el caballo de Troya que usa las herramientas que este ordenamiento le entrega con tal de desafiarla. En este contexto, surge la pregunta por las obras derivadas en cuanto elemento que la licencia dedica gran extensión a regular.

En el capítulo 1, se hizo referencia a la relación entre *software* y Propiedad Intelectual. Primero se expuso el tratamiento de los programas computacionales bajo la Propiedad Intelectual, en particular se vio como se constituye este régimen jurídico como la protección idónea para los programas computacionales, el trato del *software* como obra literaria y las disposiciones generales de la ley 17.336 al respecto. Después en la sección conceptos de licenciamiento de Programas Computacionales, se trató conceptos generales de computación, el paso histórico del código abierto a cerrado y las distintas modalidades de licenciamiento propiamente tal que son de interés para el análisis de la GPL. A grandes rasgos, el primer capítulo del presente trabajo crea un marco de análisis que entremezcla elementos históricos y jurídicos, así como un acercamiento preliminar entre conceptos informáticos al área jurídica, que otorga una sólida base de la cual leer la GPL.

El capítulo 2 se dedica el análisis a la GPL propiamente tal. El primer análisis de las implicancias de que esta sea una licencia permitió delimitar las funciones que cumple en la relación legal entre el autor y el recipiente del programa. De las características de la GPL como licencia de código abierto, copyleft y libre, destaca que el carácter de abierto permite la creación de obras derivadas y no discriminar contra áreas de trabajo ni personas, mientras que sea copyleft determina que está afecta al llamado efecto virus y como licencia libre persigue las llamadas 4 libertades establecidas por la FSF, las cuales favorecen algunas hipótesis de uso del programa pero limita otras.

Respecto de los conceptos de Obra Derivada y Distribución bajo la GPL, se observa que la licencia en su versión 2 usa términos de Propiedad Intelectual de tal forma que pueden existir disparidades entre los conceptos usados por esta licencia y la ley nacional a aplicar, mientras

que la versión 3 evita a propósito el uso de dichos términos, a fin de permitir mejor la aplicación de la licencia, lo que determina que este es un elemento que, de no ser tratado con el debido cuidado, puede dificultar el entendimiento de la licencia. Las recomendaciones en este punto son, en el caso de la GPL v2, considerar la influencia de la ley de *Copyright* estadounidense y, en el caso de la GPL v3, aterrizar los conceptos nuevos a los términos usados por la ley nacional de Derecho de Autor a aplicar, previo al inicio del análisis.

Finalmente en el capítulo 2, se ve las restricciones a las obras derivadas, donde a través de las distintas situaciones planteadas se comprueba la hipótesis planteada en la introducción, desde que se ve que la explotación de la obra en el ámbito privado es la menos restringida, mientras que la Distribución y Puesta a Disposición de una obra introduce una serie de complicaciones, principalmente los deberes de dejar disponible el código fuente, la pregunta por los enlaces dinámicos y estáticos y la incompatibilidad entre licencias, constando el SaaS como una notable excepción desde que no constituye Distribución o Puesta a Disposición. Los elementos diferenciadores principales que se encuentran entre estas situaciones es: a) el hecho de si la copia del programa computacional en cuestión no ha sido publicado (en el sentido que le otorga la GPL) y; b) si la obra es publicada, la interacción entre la GPL y otra licencia que pueda presentar problemas de compatibildad.

En el capítulo 3, con tal de que el análisis se vea sustentado en su relevancia, se analiza su validez de las estipulaciones de la GPL en el derecho chileno, la cual se dividió en dos aristas: la validez de la GPL en cuanto autorización de obra ajena y los límites del Derecho de Autor, en cuanto estos últimos significan la frontera de lo que la GPL puede regular.

El análisis de la GPL como contrato determino que este es válido como tal bajo las definiciones doctrinales y la teoría (mayoritaria) de la aceptación como formación del consentimiento. Aun así, se encuentra posteriormente que la GPL como autorización de uso de obra ajena, no cumple con las menciones esenciales requeridas por el artículo 20 de la ley 17.336 si se toma la posición de que estas deben ser expresas, por el contrario, si se considera que la GPL es autorización vigente en virtud de la norma habilitante del artículo 17, no requiere cumplir con todas estas menciones, sino sólo las necesarias con tal de que no esté vacío de contenido. No obstante, lo anterior, analizada la jurisprudencia de la Corte Suprema, se encuentra que su posición respecto de si las menciones esenciales del artículo 20 deben ser expresas o tácitas, es ambivalente y en

última instancia no se puede determinar de antemano un criterio único mediante el cual resolvería un caso que involucrara a la GPL o una licencia similar.

En el análisis de los límites al Derecho de Autor, se hace referencia a la dicotomía expresión/idea y su aplicación a los programas informáticos, así como se expone que, en principio, resulta aplicable a la tradición continental, la *merger doctrine* como límite a la Propiedad Intelectual en aquellos casos en que se fusiona la idea y la expresión, entregando algunas herramientas para su eventual aplicación a un caso real, posteriormente se realiza un análisis desde la metodología entrega por Pellegrini, la cual se ve entrega resultados similares a los de la *merger dcotrine*. Esta sección, mediante los límites al Derecho de Autor, señala aquello que, por extensión, no puede ser regulado o apropiado por la GPL, que termina siendo los denominados aspectos funcionales de un programa computacional, tales como las interfaces de interacción entre programas.

Este trabajo en cuestión ofrece entonces una herramienta para el análisis de casos de Propiedad Intelectual que involucren a la GPL. Sin embargo, el análisis se ha visto limitado por la extensión y propósito específico que se ha propuesto, futuras líneas de investigación podrían darse en aquellos espacios donde el trabajo no ha podido ser conclusivo, incluyendo pero no limitándose a la aplicación de la *merger doctrine* en el derecho continental, el problema de los enlaces dinámicos o si las menciones esenciales del artículo 20 de la ley 17.336 deben ser explicitas o tácitas. En definitiva, el análisis del copyleft y el *software* libre en el derecho chileno tiene gran potencial de seguir siendo investigado.

Bibliografía

Abeliuk Manasevich, René. *Las obligaciones. T. 1*. 5. ed. actualizada. Santiago, Chile: Ed. Jurídica de Chile, 2010.

"Affero GNU General Public License, Version 3". Free Software Foundation, 19 de noviembre de 2017.

Allessandri Rodríguez, Arturo. *De los contratos*. Chile: Jurídica de las Americas, 2009.

Bain, Malcolm, Manuel Gallego Rodríguez, Manuel Martínez Ribas, y Judit Rius Sanjuán. *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya, 2009. http://hdl.handle.net/10609/229.

Bajer, Anna. "Understanding of source code in language: Contribution of philosophical hermeneutics to the critical code studies". *Digital Scholarship in the Humanities* 37, nº 2 (1 de junio de 2022): 307–20. https://doi.org/10.1093/llc/fqab089.

Balter, Ben. "Open Source License Usage on GitHub.Com". *The GitHub Blog* (blog), 10 de marzo de 2015. https://github.blog/2015-03-09-open-source-license-usage-on-github-com/.

Bernet, Manuel Antonio, y Alfredo Sierra. "La autorización de utilización de una obra ajena en la Ley de Propiedad Intelectual chilena". *Revista Chilena de Derecho y Tecnología* 8, nº 1 (30 de junio de 2019): 107. https://doi.org/10.5354/0719-2584.2019.52647.

Böhm, Mirko. "Economics of Open Source". En *Open Source Law, Policy and Practice*, editado por Amanda Brock, 0. Oxford University Press, 2022. https://doi.org/10.1093/oso/9780198862345.003.0015.

Brown, Neil. "GNU GPL 2.0 and 3.0: obligations to include license text, and provide source code". *International Free and Open Source Software Law Review* 2, nº 1 (17 de junio de 2010): 7–12. https://doi.org/10.5033/ifosslr.v2i1.31.

"Definición de programa de ordenador - Diccionario panhispánico del español jurídico - RAE". Accedido 11 de mayo de 2023. https://dpej.rae.es/lema/programa-de-ordenador.

Desforges, Charles D. *The Commercial Exploitation of Intellectual Property Rights by Licensing: A Thorogood Report*. London: Thorogood, 2001.

Donoso Abarca, Lorena, y Carlos Reusser Monsálvez. *Derecho informático*. Serie de documentos materiales docentes. Santiago de Chile: Academia Judicial de Chile, 2021. https://academiajudicial.cl/wp-content/uploads/2022/03/Derecho-informatico.pdf.

Ducci Claro, Carlos. *Derecho civil: parte general*. 4a ed. Santiago de Chile: Jurídica de Chile, 2007.

Gardler, Ross, y Stephen Walli. "Evolving Perspective on Community and Governance". En *Open Source Law, Policy and Practice*, editado por Amanda Brock, 2ª ed. Oxford University Press, 2022. https://doi.org/10.1093/oso/9780198862345.001.0001.

Gina Häußge. "A dev's guide to open source software licensing". Accedido 26 de junio de 2023. https://github.com/readme/guides/open-source-licensing.

"GNU General Public License, Version 2". Free Software Foundation, junio de 1991.

"GNU General Public License, Version 3". Free Software Foundation, 29 de junio de 2007.

Guadamuz, Andres. "Viral contracts or unenforceable documents? Contractual validity of copyleft licenses". *European Intellectual Property Review* 26, nº 8 (1 de enero de 2004): 331–39.

György Kratochwill, Jr. "What is the Difference between a Software License Agreement and a 'Software as a Service' (SaaS) Agreement?" *les Nouvelles - Journal of the Licensing Executives Society* LVI, nº 3 (septiembre de 2021): 259–61.

Haapanen, Anna. "Free and Open Source Software and the Mystery of Software Patent Licenses Under the GPL". *International Free and Open Source Software Law Review* 7, nº 1 (29 de diciembre de 2015): 19–28. https://doi.org/10.5033/ifosslr.v7i1.107.

Hardings, Jens, y Alejandro Fuentes. "Software Libre: introducción histórica Revisión: 1.1", 04/072003. https://www.hardings.cl/publications/hardings2003intro.pdf.

"Las Licencias de Software (I)". En *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya, 2009. http://hdl.handle.net/10609/229.

"Las Licencias Libres en la práctica". En *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya, 2009. http://hdl.handle.net/10609/229.

"Licencias de Software Libre". En *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya, 2009. http://hdl.handle.net/10609/229.

Liliana Galindo Díaz, Andrea Martínez Devia, y Rodolfo Yáñez Otálora. "Los derechos de autor en el entorno digital y el Internet en Colombia: una mirada al estado del arte actual y sus principales problemas". *Revista de Derecho, Comunicaciones y Nuevas Tecnologías*, nº 10 (diciembre de 2013): 2–36.

López Fernández, Elba. "El plagio: la dicotomía entre la idea y la expresión de la idea". *RICSH Revista Iberoamericana de las Ciencias Sociales y Humanísticas* 9, nº 17 (8 de enero de 2020): 330–50. https://doi.org/10.23913/ricsh.v9i17.201.

Margaret Jane Radin. "Humans, Computers, and Binding Commitment". *Indiana Law Journal* 75, nº 4 (2000): 1125–62.

Matellán Olivera, Vicente, Jesús M. González Barahona, Pedro de las Heras Quirós, y Gregorio Robles Martínez, eds. *Sobre software libre: compilación de ensayos sobre software libre*. Madrid: Universidad Rey Juan Carlos, Servicio de Publicaciones : Dykinson S.L., 2004.

Mell, P M, y T Grance. "The NIST Definition of Cloud Computing". 0 ed. Gaithersburg, MD: National Institute of Standards and Technology, 2011. https://doi.org/10.6028/NIST.SP.800-145.

Menell, Peter S. "Economic Analysis of Network Effects and Intellectual Property", 2019. https://doi.org/10.15779/Z381V5BD7T.

OMPI. "Tratado de la OMPI sobre Derecho de Autor", 20 de diciembre de 1996. TRT/WCT/001. https://www.wipo.int/wipolex/es/treaties/textdetails/12740.

Open Source Initiative. "The Open Source Definition", 7 de julio de 2006. https://opensource.org/osd/.

Pellegrini, François. “ORIGINALITY OF SOFTWARE WORKS”. *International Review of Copyright,* nº 252 (abril de 2017): 44–105.

“Por qué en su próxima biblioteca no debería utilizar la Licencia Pública General Reducida de GNU - Proyecto GNU - Free Software Foundation”. Accedido 8 de octubre de 2023. https://www.gnu.org/licenses/why-not-lgpl.es.html.

“Por qué es necesaria la GPL Affero - Proyecto GNU - Free Software Foundation”. Accedido 1 de noviembre de 2023. https://www.gnu.org/licenses/why-affero-gpl.es.html.

“Por qué la FSF solicita a sus colaboradores la asignación del copyright - Proyecto GNU - Free Software Foundation”. Accedido 27 de noviembre de 2023. https://www.gnu.org/licenses/why-assign.es.html.

“Preguntas frecuentes acerca de las licencias de GNU - Proyecto GNU - Free Software Foundation”. Accedido 20 de septiembre de 2023. https://www.gnu.org/licenses/gpl-faq.es.html.

Principios básicos del derecho de autor y los derechos conexos. Genève, Switzerland: Organisation Mondiale de la Propriété Intellectuelle (OMPI), 2016.

“Propiedad Intelectual: derechos de autor y copyright”. En *Aspectos legales y de explotación del software libre*. Universitat Oberta de Catalunya, 2009. http://hdl.handle.net/10609/229.

“recast verb - Definition, pictures, pronunciation and usage notes | Oxford Advanced Learner’s Dictionary at OxfordLearnersDictionaries.com”. Accedido 1 de enero de 2024. https://www.oxfordlearnersdictionaries.com/definition/english/recast?q=recast.

“Richard Stallman at the First Hackers Conference in 1984 - GNU Project - Free Software Foundation”. Accedido 21 de noviembre de 2023. https://www.gnu.org/gnu/first-hackers-conference-1984.html.

Rol 131673-2020 (Corte Suprema 3 de agosto de 2022).

Samuelson, Pamela. “Reconceptualizing Copyright’s Merger Doctrine”. *Journal of the Copyright Society of the U.S.A.* 63 (4 de diciembre de 2016). https://ssrn.com/abstract=2763903.

SAP PO 678/2005 (Audiencia Provincial de Pontevedra 29 de noviembre de 2005).

Schuster Vergara, Santiago. “Las excepciones al derecho de autor como normas permisivas: una revisión crítica del derecho de usuario en el sistema normativo de derecho de autor”. Tesis para optar al grado de Doctor en Derecho, Repositorio Universidad de Chile, 2019.

Shemtov, Noam. “70On Reverse Engineering and Decompilation”. En *Beyond the Code: Protection of Non-Textual Features of Software*, editado por Noam Shemtov, 0. Oxford University Press, 2017. https://doi.org/10.1093/oso/9780198716792.003.0003.

“The GNU General Public License v3.0 - GNU Project - Free Software Foundation”. Accedido 9 de agosto de 2023. https://www.gnu.org/licenses/gpl-3.0.html.

“The GNU Manifesto - GNU Project - Free Software Foundation”. Accedido 21 de noviembre de 2023. https://www.gnu.org/gnu/manifesto.html.

U.S. Copyright Office. “Circular 92 Copyright Law of the United States and Related Laws Contained in Tıtle 17 of the United States Code”, diciembre de 2022. https://www.copyright.gov/title17/.

———. “The Making Available Right in the United States: A Report of the Register of Copyrights”, 23 de febrero de 2016. https://www.copyright.gov/docs/making_available/making-available-right.pdf.

“Visión general del sistema GNU - Proyecto GNU - Free Software Foundation”. Accedido 18 de octubre de 2023. https://www.gnu.org/gnu/gnu-history.es.html.

Walden, Ian. “Open Source as Philosophy, Methodology, and Commerce: Using Law with Attitude”. En *Open Source Law, Policy and Practice*, editado por Amanda Brock, 2ª ed., 1-C1.P124. Oxford University PressOxford, 2022. https://doi.org/10.1093/oso/9780198862345.003.0001.

Walker Echenique, Elisa. *Manual de propiedad intelectual*. 2a. ed. actualizada. Santiago, Chile: Thomson Reuters, 2020.

“What is Copyleft? - GNU Project - Free Software Foundation”. Accedido 14 de junio de 2023. https://www.gnu.org/copyleft/.

"What is Free Software? - GNU Project - Free Software Foundation". Accedido 1 de mayo de 2023. https://www.gnu.org/philosophy/free-sw.html.

WIPO Intellectual Property Handbook. Geneva, Switzerland: World Intellectual Property Organization (WIPO), 2008.