Roman Yankovskiy, PhD (Candidate of Law)
Associate Professor at HSE University; Partner of *Tomashevskaya and Partners* Law Firm, (e-mail: roman.yankovskiy@gmail.com).

Ivan Bardov
Co-Founder of *Legal Tip* Law Firm (e-mail: bardov.ivan@gmail.com).

Artem Nikiforov
Senior Associate at *GeekBrains* LLC, Postgraduate Student of the Law Faculty of Lomonosov Moscow State University (e-mail: ar.nikiforov@gmail.com).

For citation:

Yankovskiy, R. M., Bardov, I. A., & Nikiforov, A. A. (2022). *Tri vzglyada na kompyuternuyu programmu: iskhodnyy tekst, proizvodnoe i sluzhebnoe proizvedenie* [Three Legal Views of a Software: Source Code, Derivative Work, and Work for Hire]. *Vestnik ekonomicheskogo pravosudiya Rossiyskoy Federatsii* [Bulletin of Economic Justice of the Russian Federation], (10), 100-137. doi:10.37239/2500-2643-2022-17-10-100-137

Three Legal Views of a Software: Source Code, Derivative Work, and Work for Hire[1][2]

Abstract

As the digital landscape continues to evolve, intellectual property law plays an increasingly significant role in regulation of websites, databases, and computer software. This article critically examines key issues in the regulation of computer software in Russia, highlighting potential shortcomings in the legislation and its alignment with legal practice and technological advances. Specifically, the analysis focuses on three core areas: 1) the legal definition of computer software and its source code, 2) the legal protection of computer software, and 3) the regulation of software made for hire. We will focus in particular on source code as the primary component of a computer software.

While this article provides valuable insights into the aforementioned aspects of software regulation in Russia, it does not purport to address all related legal issues. In particular, it excludes discussions on unprotected elements of computer programs, software alienation, protection of software-generated audiovisual images, and indirect copying of source code. It also does not address the legal issues of "import substitution" of software in Russia, such as the legal status of Russian software and compulsory licensing of foreign software.

Keywords: source code, computer software, open-source software, free software, copyright, HTML-code, proprietary software

DOI 10.37239/2500-2643-2022-17-10-100-137

[1] The authors of this article would like to express their gratitude to Evgeny Dontsov and Alexey Zaitsev for their proofreading efforts, insightful comments, and unwavering patience.
[2] I would like to express my deepest gratitude to Allie Andersen for her invaluable assistance in translating this article into English. Her keen eye and expertise were truly remarkable. Any remaining errors, inaccuracies or omissions are solely my responsibility. – Roman Yankovskiy

Introduction

The legal regulation of software is an important area for growth in Russian intellectual property law. We have summarized the most important and least explored issues in one article. These are areas where the law lacks specificity, is inconsistent with established practice, or conflicts with the technical aspects of software development and operation. These areas include: 1) the legal classification and status of software and its source code; 2) legal protection of software; 3) derivative software and versioning; 4) protection of software created as a work for hire.

This article does not attempt to cover all legal issues related to software. In particular, it does not address issues relating to unprotected elements of computer programs, alienation of software, protection of software-generated audiovisual images, or undirect copying of source code. It also does not address the problems of "import substitution" of software in Russia (legal status of Russian software, compulsory licensing of foreign software, etc.)

1. Software and its types

1.1. The traditional view of a computer program in Russian law

The definition of a computer program is given in Article 1261 of the Civil Code of the Russian Federation[3] (hereinafter the Civil Code). A program is a set of data and commands, presented in an objective form, intended to operate an "electronic computing machine or other computer device" in order to achieve a certain result, including preparatory materials obtained in the development of the program and audiovisual content generated by it. The program may be developed in any language and in any form; the Civil Code explicitly provides such forms as source code and object code. In the following, for the sake of brevity, we will use the commonly used terms “computer program” and “software” to refer to any type of application or set of computer instructions.

The definition in the Civil Code reflects the common understanding of a computer program and how it works. To recall a typical high school computer science class, a program is usually thought of an instruction like the following simple "Hello, world!" program on QBasic:

```
10 cls
20 print "Hello, world!"
30 sleep
40 end[4]
```

These commands exist as source code, written in a high-level programming language (simply put, a language that can be read by humans). The example above shows a program written in an interpreted language, i.e. this program is intended to be processed (interpreted) by the computer line by line. So first it runs the cls (clean screen) command, then it prints "Hello,

[3] Part 4 of the Civil Code of Russian Federation enacted 18 December 2006 (Federal law №230-FZ).
[4] We are not going to mention the authors of simple programs such as this "Hello, World! These programs do not require a significant amount of creative input and can literally be written by anyone with a basic knowledge of the programming language.

world!", then it sleeps (goes to sleep mode), and then it ends. This is a general idea of how computer programs work.

It seems reasonable that computer programs are protected by law as literary works[5]. There is no definition of a literary works in Russian law, but we can take the United States Code as an example: these are "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects <…> in which they are embodied"[6]. This definition applies well to computer software, as it consists of words, numbers and other signs and exist regardless of the objects on which it is recorded. After all (and in terms of Russian law), software is also the result of creative effort and reflects the ideas of its creators, which makes it similar to literary works.

As a matter of fact, the development process, use, and context of computer software are each highly specific and distinct from other forms of intellectual property. In the following section, we will highlight the key features of software, including size, design, and other relevant aspects.

1.2. The development of software and its specificities

Computer programs are typically the result of a collaborative effort by many co-authors, often strangers to each other. For modern software it's rare to be written entirely from scratch; it's much faster and easier to use pre-developed components. There are plenty of off-the-shelf resources available, including libraries of networking, mathematical and other algorithms, pre-built graphical elements, and more. Developers are constantly working to improve these components, refining them and eliminating their weak points, which ultimately leads to better performance for the software that utilizes them. The terms of use for off-the-shelf resources are determined by their creators and outlined in a license agreement. Such agreements can vary greatly, from royalty-bearing to royalty-free, from standardized forms to individually tailored documents. As a result, the use of libraries and other components can have a range of legal implications.

Some programs, such as websites, are made up primarily of standard components. By using off-the-shelf components, it's possible to get a fully functional website up and running in a matter of days, complete with attractive graphics, animations, and support for all modern browsers and devices. In fact, the developers may not need to write any source code at all. This means that the quantity of off-the-shelf components used in a program will greatly surpass not only the amount of hand-written code, but also the overall quantity of meaningful content on the website. All these factors make most computer programs, from a legal point of view, highly composite works.

Most computer programs are very large in size. Even small applications, such as mobile games, can contain anywhere from 10,000 to 50,000 lines of code. Top-tier projects, on the other hand, are much larger. For example, Quake 3 (a 1999 video game) consists of 400,000 lines of code; the Google Chrome browser has up to 6.7 million lines, and the Android OS contains a whopping 12 million lines (as shown in *Figure 1*). To give you an idea of the scale, an A4 page of text typed in Microsoft Word using Times New Roman 12-point font and 1.15 line spacing contains 46 lines. So, if you were to print the source code of a simple mobile application with

---

[5] Article 1259.1 of the Civil Code.
[6] 17 USC § 101.

these settings, you would end up with more than 1,000 pages. Quake 3 would require more than 8,000 pages, and the Android operating system would take more than 250,000 pages.

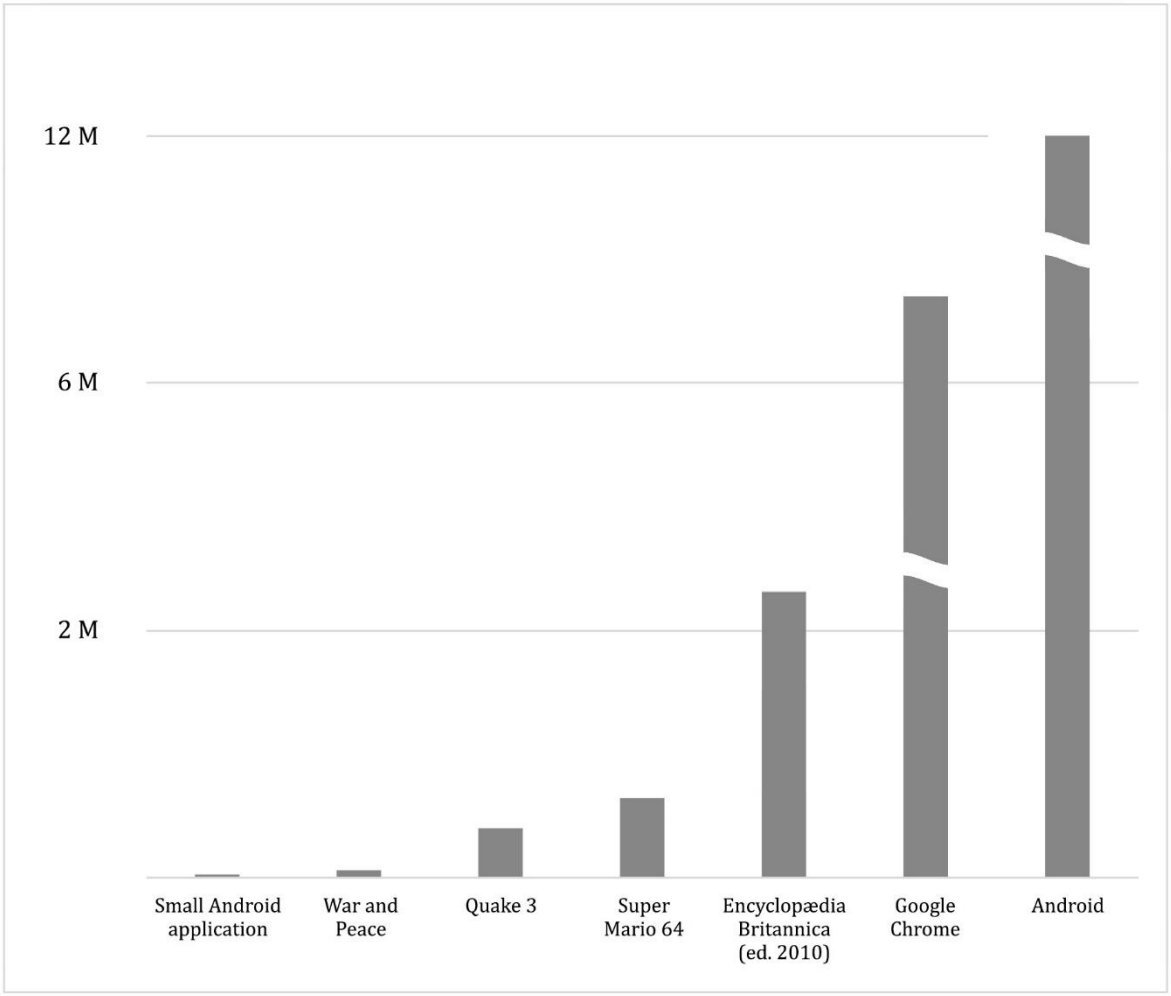

Fig. 1. Volume of some literary works and computer programs in millions of lines (a logarithmic scale).

At the same time, the source code for any program is constantly changing due to ongoing development and support. Developers are adding new features, fixing bugs, improving performance, and ultimately restructuring the source code. Because of this, even programs that have already had their basic functionality implemented can see dozens of changes and hundreds of lines of code being added on a daily basis[7]. To manage these changes, developers use special tools such as source code repositories, which usually include version control tools, project and task management applications, change request trackers, and more.

As a result, program development is a collaborative effort involving dozens of co-authors who use a multitude of off-the-shelf components. This results in the source code for many programs becoming very large and constantly being updated and modified.

[7] As an example, take a look at the extensive list of changes made to the Linux kernel source code on a daily basis.

1.3. Software compilation and reverse engineering

Most of the programs that are installed and run on a personal computer, a mobile phone, or any other similar device are written in a compiled programming language. This means that the source code of a program is first converted (compiled) into object code by a special program builder (compiler) and then executed directly at the operating system level[8].

Of course, there are exceptions, such as uncompiled programs written in interpreted programming (like the QBASIC program in Table 1)[9]. However, it is more likely that the source code of most programs installed on your computer is closed, and it can only be accessed by obtaining it from the developers.

The only way to analyze a program's installation files is to study them in their compiled form, that is, as object code. This code, unlike high-level programming languages, is not intended for human reading. Here is an example of the object code of a standard "Hello, World!" program, compiled for an x86 architecture processor, in hexadecimal representation[10]:

```
BB 11 01 B9 0D 00 B4 0E 8A 07 43 CD 10 E2 F9 CD 20 48 65 6C 6C 6F 2C
20 57 6F 72 6C 64 21
```

Converting the object code into a human-readable form requires a resource-intensive process of decompiling[11]. Even in cases where special decompilation software can be used, this process takes a long time, and the larger the program, the longer it takes. One example is the decompiling of the Super Mario game, released in 1996 for the Nintendo 64 console. In 2019, a group of enthusiasts published the game's source code, decompiled using modern tools. The result was about 660,000 lines (including project documentation)[12], and the work took several years.

A lot of software programs have built-in protection against decompiling. For instance, developers may obfuscate the program's code to throw off competitors or researchers. Optimizing a program for faster performance and smaller size also makes it more difficult to restore the

---

[8] Here and below, we will use the term "object code" for compiled source code, following the letter of the Civil Code, even though it is more correct to refer to it as machine code. Object code is not any machine code, but only that which is contained in intermediate object files, obtained after compilation of individual program components, but before their final assembly into an executable or library file.

[9] In some interpreted programming languages (e.g., the Solidity smart contract language), code is not compiled, but translated into so-called bytecode, which is very different from the original. Although technically they are two different entities, it seems correct to extend the legal regime of object code to bytecode.

[10] For the sake of clarity, "Hello, World!" is not a specific program, but any program that displays the given text. "Hello, World!" is a simple program, and can be written in any programming language, either interpreted or compiled. The compiled code of such a simple program as "Hello, World!" written in different languages, but for the same processor architecture, will be close.

[11] Strictly speaking, converting object code to source code requires not one but two conversion steps: disassembly (translation of a program from machine code to assembly language) and decompilation itself (translation of a program from assembly language to high-level language source code). However, decompilation is often used, including in this article, as a general term encompassing both actions (decompilation in the broad sense).

[12] Project repository: https://github.com/n64decomp/sm64.

original code in its intended form. For example, the original *Perl* code that outputs the text "Just another Perl hacker," looks like this:

```
print "Just another Perl hacker,";
```

An obfuscated code version that achieves the same result might look like this[13]:

```
@P=split//,".URRUU\c8R";@d=split//,"\nrekcah xinU / lreP rehtona
tsuJ";sub p{@p{"r$p","u$p"}=(P,P);pipe"r$p","u$p";++$p;($q*=2)+=$f=!fo
rk;map{$P=$P[$f^ord($p{$_})&6];$p{$_}=/ ^$P/ix?$P:close$_}keys%p}p;p;p
;p;p;map{$p{$_}=~/^[P.]/&&close$_}%p;wait un-
til$?;map{/^r/&&<$_>}%p;$_=$d[$q];sleep rand(2)if/\S/;print
```

In Russia, decompiling a program without the copyright holder's permission is legal under specific conditions[14], such as when it is the only way to make the program interoperable with other software. Relevant conditions on the right to decompile a program are found abroad, for example, in Article 6 of the EU Directive on the Legal Protection of Computer Programs[15] or in US case law[16].

It's crucial to note that the text of a decompiled program likely will not be identical to the original. In this regard, one should not be misled by the wording of the Civil Code, which equates decompiling with "transformation of object code into source code"[17]. While the program's text can be reconstructed, it will not match the original text, meaning it will not be "source" of it in the conventional sense (Fig. 2). Decompiling only enables the reconstruction of the program's high-level language representation, not necessarily its original form. The type of decompilation software used can also significantly impact the result.

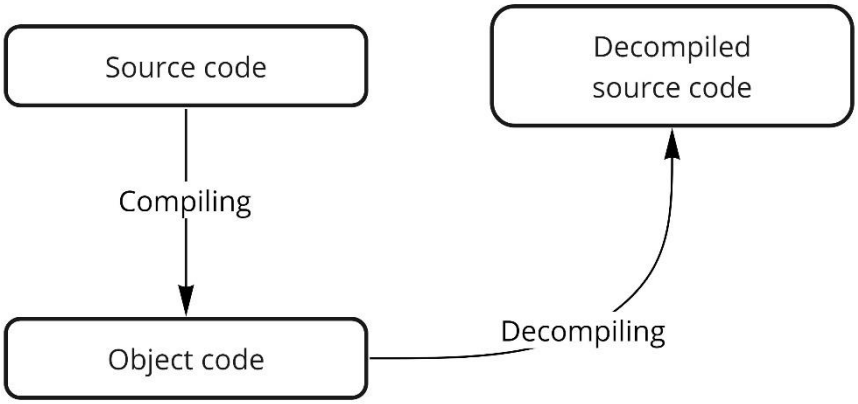

Fig. 2. Source code before and after decompiling

[13] The program was written by Mark Jason Dominus. URL: https://perl.plover.com/obfuscated/
[14] This conditions are outlined in Article 1280 (3) of the Civil Code. See also Russian Intellectual Property Court (IPC) Decision No C01-1116/2017 of 05.12.2018 in case No A45-13248/2017.
[15] Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the Legal Protection of Computer Programs.
[16] Sega Enterprises Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992).
[17] Article 1280.

Because decompiling can be challenging, it's often more straightforward to create a similar program via reverse engineering than to try to break down an existing one. This involves analyzing a program's object code and how it responds to determine how the program operates. Armed with this knowledge, a developer can create a new program that replicates the same functions without containing the original program's source code, thus avoiding copyright infringement.

Computer software has an important feature that is often overlooked. A computer program is expressed not only through its source code but also its object code (as well as audio-visual displays and preparatory materials, according to the definition in the Civil Code). While these entities do not necessarily create a complex legal object, they do influence each other: source code is compiled into object code, which then forms an executable file. When the executable file runs, it generates audio-visual displays that are also part of the program.

Typically, users do not have access to a program's source code. Unless it's open-source software, the source code for most programs is not available to third parties and retrieving it from the program's executable files is either impossible or extremely challenging. As a result, users do not have access to all forms and components of a program, including its source code or the preparatory materials used during the development process.

1.4. Declarative software and its status in Russia

Both compiled and interpreted programs share a common feature: they contain computational operations, which essentially means they execute an algorithm to solve a computational task. That a program has an algorithm as its key element is obvious to computer scientists and has been described in the literature[18].

Various standards also outline the connection between a program and computational tasks. For example, *ISO/IEC/IEEE* requires that the program be written in a programming language[19]. Furthermore, the programming language must meet specific criteria to be considered legitimate, with the primary attribute being the language's Turing-completeness, meaning it should be capable of implementing any computable mathematical function.

In contrast, let's look at the definition of a program in the Civil Code:

> "…a set of data and commands intended for the functioning...of computer devices for the purpose of obtaining a specific result."

This definition does not require a program to include computational operations or be written in a recognized programming language. The disadvantage of this broad definition is that software that is not strictly a program from a common point of view may still be classified as such under the Civil Code. Examples include database queries, regular expressions, configuration files or web pages. Let's have a look at this issue with the example of web pages.

Technically, a website may contain programs written in various languages[20] that execute both on the web server and client-side in the browser. However, the markup language HTML,

---

[18] See Nichols, K., *Inventing Software: The Rise of 'Computer-Related' Patents* 38-41 (1998).
[19] See ISO/IEC/IEEE 24765:2017 (E) Systems and software engineering — Vocabulary. S. 3.726. P. 85.
[20] So-called scripting languages are very common in web programming, and so the software components of web pages are often referred to as scripts.

which lacks computational operations, is sufficient to display basic website content like text and images. HTML is used on all websites, and for a relatively long time static websites written in pure HTML without using programming languages constituted the majority of internet pages.

Although HTML syntax somewhat resembles a programming language, it does not describe a control flow to perform an action; instead, it specifies the outcome. Such languages are called declarative; as a declarative markup language, HTML cannot perform computational operations because it only allows existing data (text, images) to be displayed in a particular way. HTML does not pass the Turing completeness test, meaning that it can only describe a limited set of actions that the visitor's browser will perform. Thus, HTML code is not a computer program in the fullest sense, and HTML is not a programming language. Here's an example of HTML code that outlines a simple website with a single line of text:

```
<!DOCTYPE HTML>
<head>
<title>Simple Website Title </title>
</head>
<body>
<p>This is a simple website. <font color="red" face="Times New Ro-
man">F</font>irst letter of this sentence is written in Times New Ro-
man and painted red.</p>
</body>
</html>
```

Russian legislation does not distinguish between programming languages and similar entities such as markup languages, query languages, pseudocode, etc. This implies that rights to the HTML code that constitutes a website can be protected in the same manner as a computer program.

According to the legislation, a computer program is a necessary element of a website, and HTML markup pages can fulfil this role. For example, the Federal Law “On Information…”[21] defines a website as "a set of computer programs and other information... accessible through the Internet... by domain names...". From this it can be concluded that a computer program is a fundamental component of a website. One can also refer to the position of the Supreme Arbitration Court of the Russian Federation in 2008, which stated that:

> "the content of a website consists of specially selected and arranged materials (texts, drawings, photographs, diagrams, audiovisual works, etc.) that can be used with the help of a computer program (computer code), which is an element of the website."[22]

Therefore, both Russian legislation and case law regard a computer program as an essential part of a website. Since many websites contain only HTML code and content (text and graphics),

---

[21] Item 13 of article 2 of the Federal Law No. 149-FZ of 27.07.2006 “On Information, Information Technology and Information Protection”.

[22] Decision of the Presidium of the Supreme Arbitration Court dated 22.04.2008 No. 2 55/08 in case No. A63-14046/2006-C1. Cf. Decision of the Ninth Arbitration Court of Appeal of 31.05.2010 in case № A40-64483/09-76-279. The court considered a similar issue - protection of rights to an Excel spreadsheet. However, the criterion of "autonomy" of the program invented in that case excessively extends the terms of Article 1261 of the Civil Code and must not be upheld.

HTML code can be recognized as a computer program, at least for such websites, in the absence of a better alternative. This allows HTML code to be considered a full-fledged object of exclusive rights.

The Judicial Collegium for Civil Cases of the Supreme Court of the Russian Federation recently considered the issue of copying the layout of a website[23]. The plaintiff cited the similarity between the HTML code of its website and the defendant's website as evidence of infringement. By remitting the case for reconsideration, the Supreme Court indirectly allowed HTML code to be recognized as an object of copyright (presumably a computer program).

The Collegium has also stated that for HTML code to be considered an object of copyright, the author must have made a sufficient creative contribution to its creation. But in fact, the presence of an author's creative contribution is a constitutive element of any object of copyright. If a work merely fulfils a technical function and the author has made no creative contribution, it cannot be legally protected. However, HTML code can indeed require no less creative effort than a program written in a Turing-complete programming language.

The question of granting legal protection to declarative programs (including HTML code) cannot be resolved in every situation on the basis of the results of an expert assessment of the creative contribution; it requires a fundamental legal decision. In our view, there is nothing to prevent HTML code from being legally treated as a computer program. This will help to protect websites that have independent creative value.

2. Legal protection of software rights in Russia

2.1. Challenges of the source code discovery

The unique characteristics of computer programs make it difficult to protect their source code from plagiarism. Compared to other copyrighted works, such as photographs or literary works, substantive infringement of computer software is difficult to identify and prove.

The main criterion for identifying identical computer programs (and thus a violation of the rights to the original program) is the established fact that their source codes are either wholly or partially identical[24]. Copying the functional capabilities of someone else's program is not a violation of their rights[25], nor is the similarity of the goals and tasks performed by programs[26]. Therefore, to determine a violation, plagiarism must be proven by obtaining the source code of both programs and comparing them. Such a comparison usually requires special forensic examination[27].

---

[23] Resolution of the Supreme Court of the Russian Federation of 27.04.2021, No. 5-KG21-14-K2, 3-466/2019.
[24] Decision of the Court of intellectual property (IP) Rights of 19.03.2020 in case No. A40-161684/2018.
[25] Decision of the Court of IP Rights of 01.03.2016 in case No. A40-149313/2013.
[26] Decision of the Court of IP Rights of 19.03.2020 in case No. A40-161684/2018. In general, the boundaries of the legal protection of computer programs in Russian law are not clearly defined, resulting in a number of creative components of the program is not protected, and the utilitarian components are formally subject to protection. See Akhobekova, R. A. *Perspektivy ispol'zovaniya sudebnogo opyta SShA v reshenii voprosa o predmete pravovoy okhrany programm dlya EVM* [Prospects for the use of U.S. judicial experience in resolving the issue of the subject of legal protection of computer programs], *Zakon* [The Statute], No. 5, at 172-187 (2021).
[27] Decision of the Court of IP Rights of 09.08.2018 in case No. A40-20593/2017; Section 1.6 of the Review of the Court of IP Rights Practice on issues arising in the application of the Civil Code on legal protection of computer programs and databases (approved by Decree of the Presidium of the Court of IP Rights of 18.11.2021 No. СП-21/26).

However, only the violator has access to the source code of both programs, and the rightful developer whose rights have been violated has only the source code of their own program. How can they obtain the source code of the infringing program for forensic examination?

In some cases, this task can be carried out relatively easy:

1. The infringer may have registered and deposited the source code of the counterfeit program with the Russian Patent Service (Rospatent). The plaintiff can request the deposited code from Rospatent and check whether it is identical to the source code of the original program[28]. Deposition is an optional procedure and therefore unlikely to be used by the infringer;
2. It is possible to study programs of small size (e.g., microcontroller firmware) byte by byte[29];
3. If a program or library, even a large one, is 100% borrowed, file checksums can be compared to quickly determine its identity[30];
4. In some cases, one can also examine some of the program's executable files and libraries[31].

In most cases, however, establishing plagiarism usually requires obtaining the source code of the disputed program, which the plaintiff typically does not have. This requires significant legal work, as the source code must be obtained through an evidence discovery process[32], and then a forensic examination must be conducted to confirm or refute the identity of the texts. This is the part where the first challenges arise.

1. The provision of evidence prior to the claim (preliminary relief) requires sufficient evidence of the infringement of exclusive rights, although not to the extent necessary to substantiate the claims in the dispute[33].
2. Russian procedural law provides for different procedures depending on whether the case falls under civil or commercial (so-called *arbitrazh*) jurisdiction. Most copyright issues are litigated in the commercial courts, where the pre-trial evidence discovery, as well as other preliminary injunctions, is significantly limited[34].

---

[28] Decision of the Court of IP Rights of 06.06.2018 in case No. A40-248072/2016.
[29] Decision of the Court of IP Rights of 27.11.2018 in case No. A53-40003/2017.
[30] Decision of the Court of IP Rights of 27.11.2018 in case No. A53-40003/2017.
[31] Such research is not considered to be an infringement of an exclusive right because, according to the case law, it is carried out by the plaintiffs in order to identify signs of infringement for the purpose of bringing an action in court. See Decision of the Court of IP Rights of 19.11.2013 in case No. A40-10750/2013. See also Welte v. D-Link № 2-6 O 224/06 LG Frankfurt am M., 06.09.2006.
[32] However, courts, referring to the possibility of self-protection of the right, recognize the legitimacy of obtaining access to the original text or database of another person without his permission in order to compare and further protect their rights. See Ruling of the Ninth Arbitration Court of 12.08.2013 in case No. A40-10750/2013.
[33] Cf. section 160 of the Resolution of the Plenum of the Supreme Court of the Russian Federation of 23.04.2019 No.10 "On application of Part Four of the Civil Code of the Russian Federation" and section 18 of the previous Decision of the Plenum of the Supreme Court of the Russian Federation of 19.06.2006 No. 15 "On issues encountered by the courts in civil cases involving the application of legislation on copyright and related rights". An example of the abuse of the preliminary relief can be found in the decision of the Arbitration Court of St. Petersburg and Leningrad region of 19.06.2015 in case No. A56-21040/2015 (subsequently revoked).
[34] In fact, it comes down to the blocking of the site that violates exclusive rights (Art. 144.1 of the Code of Civil Procedure) and does not provide for actions such as compulsory disclosure of the source code.

3. In order to ensure the enforceability of a court order for the recovery of the source code, it is necessary to prove that it exists in the defendant's possession [35].

Difficulty proving a violation in civil or arbitration proceedings often leads victims to turn to the possibilities of criminal or administrative proceedings; such as filing a criminal complaint under Article 146 of the Criminal Code of the Russian Federation ("Violation of copyright and related rights"). If the organization has implemented a strict non-disclosure policy (in accordance with the Federal Law "On commercial secrecy"[36]), this may also fall under the Article 183 of the Criminal Code of the Russian Federation ("Illegal obtaining and disclosure of information constituting commercial, tax or banking secrets"). Victims can also report a violation under Article 7.12 of the Code of Administrative Offenses of the Russian Federation ("Violation of copyright and related rights, invention and patent rights"). Evidence can be gathered much more quickly and efficiently for use in civil and commercial proceedings by using the tools available in criminal proceedings (search, seizure)[37] and administrative proceedings (inspection of premises and documents)[38].

Criminal and administrative prosecution are not always effective as a means to obtain evidence and by resorting to it the victim can damage their own reputation. It is widely believed that the use of criminal law instruments in such disputes is excessive. In this situation, the party with the best connections in the law enforcement agencies gains an unfair advantage, which forces the weaker party to resort to such practices as well. All this has a negative impact on competition in the IT market and increases the pressure for corruption[39]. The recent high-profile case involving the criminal prosecution of the beneficiaries of the Nginx at the request of Rambler is a good example[40]. Another example is the dispute between Mail.ru and former employee Yuri Gursky over the Prisma app[41].

Such conflicts are often exacerbated by the fact that criminal cases under Articles 146 and 183 of the Russian Criminal Code are considered public charges and therefore cannot be dropped at the request of an injured party. For this reason, Rambler was unable to drop the charges in the Nginx case, even after a direct request from the major shareholder[42].

In the United States, by contrast, the practice of issuing orders for the disclosure of source code in both civil and criminal proceedings is quite common[43]. Interestingly, requests for the disclosure of source code are made (and granted) not only in cases of infringement of exclusive

---

[35] Decision of the Eighth Arbitration Court of 21.05.2019 in case No. A70-36/2019.
[36] Federal Law No. 98-FZ of July 29, 2004 "On commercial secrecy".
[37] For the use of evidence obtained in criminal proceedings in software copyright arbitration, see Decision of the Thirteenth Arbitration Court of 28.06.2011 in case № A56-69593/2010.
[38] See Decision of the Court of IP Rights of 25.09.2018 in case No. A28-4981/2017.
[39] See The Official Position of the Program Committees of Highload++ and Other IT Conferences on the Claims Against Igor Sysoev and Maxim Konovalov, *Habr.com* (Dec. 13, 2019), https://habr.com/ru/company/oleg-bunin/blog/480136/.
[40] Materials of the criminal case № 11901450149005396 are available on the Internet and in the media. See Chronicle of the Rambler/Nginx confrontation // Habr.com. 2019. 16 Dec. URL: https://habr.com/ru/post/480510/.
[41] Yapparova, L., It Feels Like You're in an Action Movie from the 1990s: How Russian IT-Companies Are Competing with the Help of Law Enforcement Agencies, *Meduza* (Feb. 26, 2020), https://meduza.io/feature/2020/02/26/oschuschenie-chto-ty-okazalsya-v-boevike-iz-90-h.
[42] Lev Khasis: "We have nothing more to do with the Nginx case", *Vedomosti (Apr. 24*, 2020), URL: https://www.vedomosti.ru/technology/articles/2020/04/24/828895-ne-imeem-otnosheniya.
[43] Pallas Loren, L., & Johnson-Laird, A., Computer Software-Related Litigation: Discovery and the Overly-Protective Order, 6 *Fed. Cts. L. Rev.* 75 (2012).

rights, but also in determining the admissibility of evidence in criminal proceedings. For example, defendants often challenge the accuracy of forensic software. This is particularly common in cases involving DNA analysis programs, since the results of such analysis are based on the assessment of probabilities, and the algorithm for such assessment is described in the source code of the program. Courts are forced to call in experts to evaluate the text of the program for correctness of its work and absence of errors[44].

In Russia, there is currently no well-developed practice of judicial disclosure of source code at the request of a civil or commercial court. As a result, we still see criminal cases initiated with the aim of obtaining the source code of a program for use in court proceedings. As high-profile cases such as the "Nginx case" have shown, such criminal cases can easily be initiated without valid grounds, under the basis of formal claims.

Some experts propose excluding Article 146 from the Russian Criminal Code[45], but we believe that a more important and effective way is to develop an effective practice of compulsory disclosure of source code in court, as we see in American case law. Russian developers would gain additional legal protection if they could prosecute software plagiarists.

2.2. Registration (deposit) of source code of software

As we have already noted, computer programs in Russian copyright law are treated in the same way as literary works. According to the Civil Code, the creation, implementation, and protection of copyright does not require registration of the work or other formalities[46]. However, programs have their own peculiarities: it is more difficult to prove when they were created, and difficult to get the source code disclosed, even in court. These difficulties have been partly overcome by the establishment of a register of software copyrights and a procedure for registering source code in it.

The copyright holder of a computer program may register it in the Unified Register of Russian Programs for Electronic Computers and Databases (hereinafter - the Register) at the Federal Intellectual Property Service (Rospatent)[47]. Each new version of the software can also be registered, but this is not compulsory: the court may accept registration in the Register as proof of authorship, despite the fact that the actual version of the program differs from the registered one because of additions and changes made[48].

The Register also publishes materials "identifying the computer program," including a summary of the program, as well as additional materials (for example, a graphical representation of the interface). The register is not an open repository, meaning its open access part does not contain the source code. The requirements for the summary of the program are formal: it should

---

[44] Imwinkelried, E. J., Computer Source Code: A Source of the Growing Controversy Over the Reliability of Automated Forensic Techniques, 66 *DePaul L. Rev.* 97 (2017).
[45] Kiryanov, A., For the Benefit of the Treasury and Business, *Izvestia* (May 29, 2018), https://iz.ru/748895/artem-kirianov/na-polzu-kazne-i-biznesu.
[46] Article 1259.
[47] Article 1262 of the Civil Code.
[48] See Decision of the Court of IP Rights of 09.12.2016 in case No. A56-7695/2016.

provide "unambiguous identification of the registered computer program," [49] and include the program's name, field of application and functionalities, its size and programming language, as well as the platform (device and operating system) for which the program was written. The content of the summary should fit into 900 characters[50] - slightly more than the paragraph you just finished reading.

Until 2016, the program listings required for application were only accepted in paper form on "durable, white, smooth, non-glossy paper"[51] and limited to a maximum of 70 pages[52]. For many programs this was a small part of their volume, and this limit was later removed. The registered text is not indexed, nor is it checked for functionality or errors. Even the novelty of the program is not required: if a similar program is already registered in the Register, it will still be accepted by Rospatent[53].

In the event of a legal dispute, the information entered by Rospatent in the Register will be considered reliable *unless proven otherwise:* the person named in the Resigter is presumed to have been the author [54]. However, the applicant is responsible for the accuracy of the information provided for state registration. Therefore, registration of a computer program with Rospatent does not create or transfer any rights to it (it does not itself have any legal effect[55]), but only confirms the applicant's priority regarding the specified program as of the date of registration. A certificate for a computer program is not a document establishing rights, and the registration procedure itself, if carried out by an unauthorized person, does not affect the existence of the rights of the actual rights holder.

Thus, the mechanics of registering a program with Rospatent is no different from publishing a program in any other public registry, such as *GitHub*, where the time and date of the upload of the source code and subsequent changes are also recorded. The advantage of registering with Rospatent is that (1) it identifies the depositor; (2) this method is described in the law and is therefore better perceived by courts and other state authorities; (3) unlike *GitHub*, Rospatent will not remove programs deposited by Russian developers who have been sanctioned.

The presumption of accuracy of the information in the Register may be rebutted by proof to the contrary - if the actual right holder brings an action against the depositor[56]. The proper way to protect oneself in such a case is to demand recognition of the exclusive right by bringing an

---

[49] Section 26 of the Rules for filling out an application for state registration of a program for electronic computers or a database (approved by order of the Ministry of Economic Development of Russia dated April 05, 2016 No. 211).
[50] Ibid, section 30.
[51] Ibid, section 4.
[52] Clause 2.1 of the Rules for filling in the application (documents and materials) submitted for registration (Annex to the Order of the Ministry of Education and Science of Russia dated 29 October 2008, No. 324).
[53] Clause 2 of the Guidelines on Implementation of Administrative Procedures and Actions within the Framework of Providing Public Services for State Registration of a Program for Electronic Computers or a Database and Issuance of Certificates of State Registration of a Program for Electronic Computers or a Database, their Duplicates (approved by Order of Rospatent dated 25 July2018, No. 129).
[54] Item 6 of Art. 1262 of the Civil Code; section 109 of Resolution of the Plenum of the Supreme Court of the Russian Federation of 23.04.2019 No.10.
[55] Decision of the Court of IP Rights of 06.08.2019 in case No. A60-46975/2016.
[56] See Decision of the Court of IP Rights of 22.01.2020 in case No. A40-21788/2018, of 06.08.2019 in case No. A60-46975/2016, of 16.12.2015 in case No. A40-2686/2012; decision of the Arbitration Court of Moscow of 30.05.2014 in case No. A40-184777/13, etc.

action against the person named as the right holder in the Register[57]. This raises the question of the possibility of bringing such a lawsuit beyond the general statute of limitations. The general limitation period established by the Civil Code is three years from the date when the person discovered or should have discovered of the violation of their right (in this case, after the software is registered in the Register)[58]. Exceptions to this rule are established by law (including claims specifically listed in Article 208 of the Civil Code). The Civil Code provides for such exceptions in cases where the dispute concerns the establishment of ownership of exclusive rights. For instance, it is directly established that a patent for an invention, a utility model, or an industrial design may be contested during its entire term of validity[59].

The registration procedure for computer programs in Russia has limited advantages. Registration in the Register does not affect the actual ownership of the exclusive right[60], and information about the copyright holder in the Registry is considered reliable only until proven otherwise. The true copyright holder may file a lawsuit against the person indicated in the Register and confirm their rights regardless of who filed the registration documents first[61].

It is also unclear whether the registration of a computer program can be challenged in the same way as a patent for the entire life of the object of intellectual property. While the registration of a computer program, unlike a patent, is not a legally binding event, the registration of someone else's source code is a continuing infringement of intellectual property rights as long as the infringer benefits from it. Therefore, the true author of the program should be able to challenge the registration of the program beyond the formal statute of limitations[62].

Neither legislation nor judicial practice has yet resolved the issue of the statute of limitations for claims relating to changes in the Register. This is aggravated by the fact that the Register (or at least its publicly accessible part) only contains sufficient information on computer programs registered since 18 January 2013. Earlier records do not even disclose the nature and subject matter of the registered programs.

Overall, the registration procedure seems formal, cumbersome, and unreliable. It does not provide software rights holders with better protection than copyright holders of literary, musical, and other traditional works. This procedure is often used not for intellectual property protection, but for indirect purposes, such as identifying the subject matter of public procurement,

---

[57] Decision of the Court of IP Rights of 27.01.2015 in case No. A27-6440/2013.
[58] Art. 196.
[59] Art. 1398(2) of the Civil Code, See also Decision of the Supreme Court of the Russian Federation of 29.01.2015 in case No. 300-ES14-1301.
[60] Decision of the Federal Antimonopoly Service of Moscow of 23.08.2010 in case No. A40-119761/09-110-828.
[61] The procedure for granting legal protection to inventions, utility models and industrial designs is regulated in the Civil Code in such a way that the issue of the moment when an exclusive right arises and a closing legal fact is not directly resolved and remains debatable to this day, therefore terms like “right-generating”, “right-establishing”, "legal" can be used with known reservations. See Novoselova, L. A., *Ob osobennostyakh nekotorykh pravoporozhdayuschikh faktov v patentnom prave* [On the Peculiarities of Certain Legal Facts in Patent Law], *Zhurnal Suda po intellektualnym pravam* [Journal of the Court of Intellectual Property Rights], No. 12, at 19-23 (2016).
[62] In this regard, judicial practice in recent years has gone far ahead compared to the early 2010s. See Saveliev, A. I., *Aktual'nye voprosy sudebnoy praktiki v sfere oborota programmnogo obespecheniya v Rossii* [Topical Issues of Judicial Practice in the Field of Software Turnover in Russia], *Vestnik VAS RF* [Bulletin of the Supreme Arbitration Court of the Russian Federation], No. 4, at 9 (2013).

confirming the results of research grants, confirming a company's status as a software developer for tax purposes, etc.

2.3. Open-Source Projects

We have written before about the difficulties of getting source code for commercial software, even through a court order. Aside from the problems of legal protection, the inaccessibility of the source code leads to other difficulties. Without access to the source code, users cannot verify its security, the absence of vulnerabilities, and undocumented features (such as the destination of the user data transferred). In addition, virtually all license agreements use the standard *As Is* disclaimer, which states that the developer is not responsible for damage caused by their software. As a result, not only does the user of the program have no control over its operation, but also the developer cannot be held liable for any malfunction of the program.

Conversely, having access to the source code of a program allows users to not only find errors, but also to correct them independently, as well as better understand the mechanism of its operation. For example, the developers of a "Personal Finance" application may disclose its code to confirm the security of its operation and the reliability of storing user data on bank accounts and savings. This distribution option, where the source code is available to users, is quite common and is called *source available*.

The fact that the program's code is available does not mean that the program is free[63] or that the code can be used without restriction[64]. Some developers explicitly prohibit the use of their published code elsewhere. Nevertheless, the majority of programs whose source code is published and available to users are considered free or open-source software.

What is Free and Open-Source Software (FOSS)?[65] The exchange of source code obviously emerged with the programs themselves, as it allows developers to productively study, improve and adapt these codes. Over time, the ideology of such collaboration has been described by several developer communities, which have formulated rules and principles for the exchange of source code. The two best-known communities are the *Free Software Foundation (FSF)* and the *Open Source Initiative (OSI)*. Free and open-source software can be considered as programs that meet the criteria of the FSF and/or the OSI.

The *FSF* was the first to be founded in 1985 by programmer and activist Richard Stallman, who created the ideology of "free software" and later, with the help of lawyers, translated it into

---

[63] Of course, if the entire text of the program is open, then, having sufficient knowledge, you can compile the program yourself and use it for free. However, not everyone is ready to do this, especially in the case of complex programs or if the developer requires additional guarantees for the functionality and support of the program. For example, there are many commercial distributions (assemblies) of GNU/Linux, even though the Linux kernel itself and system components are distributed under open GPL licenses. In this sense, the word free should not be misleading: according to Stallman, it should be understood of as “free speech”, and not “free beer”. See What is Free Software? URL: https://www.gnu.org/philosophy/free-sw.en.html.

[64] Obviously, the opposite is also true: not all free software is open-source.

[65] More about open source See McGowan, D., Legal Implications of Open-Source Software, 2001 *U. Ill. L. Rev.* 241. The normative definition of free software is also contained in GOST R 54593-2011 "Information Technology. Free Software. General provisions" (approved by the Order of Rosstandart of December 6, 2011 № 718-st).

specific license agreements. According to this ideology, anyone who distributes software, with or without modifications, has no right to restrict the freedom to redistribute or modify it.

The FSF has formulated four main criteria (freedoms)[66] that must be respected in the creation and distribution of free software:

0. The freedom to run the program as you wish, for any purpose.
1. The freedom to study how the program works, and change it so it does your computing as you wish.
2. The freedom to redistribute copies so you can help others.
3. The freedom to distribute copies of your modified versions to others for the benefit of the whole community.

The freedoms one and three are impossible if the program's source code is not accessible, i.e. closed. Therefore, the implementation of freedoms requires that the code of the program be available. At the same time, the very first freedom on the list - no restrictions on the purpose of using free software - allows using its code without restrictions, including creating paid (commercial) programs, military programs, for use in sanctioned countries, etc.

The Free Software ideology attracted many enthusiasts in the 1980s and 1990s. The release of the *Linux* operating system kernel on the free software terms played a major role in popularizing the idea. However, many were unhappy that the FSF had essentially monopolized the code-sharing movement, imbuing it with a specific radical philosophy that repelled the general public and corporate developers[67]. In 1998, activists Eric Raymond and Bruce Perens launched the OSI movement against "*free software*" and popularized the terms "*open software*" and "*open source*".

Like the FSF, OSI activists have published their criteria for open-source software[68]. In many ways, they duplicate the FSF's principles: licenses for open-source software must also allow free redistribution and modification. But in some cases the principles diverge: for example, the OSI criteria allow the integrity of the source code to be protected during distribution, while the FSF's freedoms do not[69]. Thus, although most free software meets the criteria for open-source and vice versa, these entities are not completely congruent, and there are exceptions. Therefore, a universal term FOSS (Free and Open-Source Software) was needed (fig. 3). Other software (non-free and closed, so to speak) is called proprietary (from the word "property").

---

[66] What is Free Software? URL: https://www.gnu.org/philosophy/free-sw.en.html.
[67] Perens explained this contradiction as follows: «Richard thinks all software should be free, and I think free and proprietary software can coexist.» (Revolution OS. URL: https://youtu.be/Eluzi70O-P4?t=2974).
[68] The Open Source Definition. URL: https://opensource.org/docs/osd.
[69] Cf. second and fourth freedoms of the *FSF* and *OSI* Criterion 4.

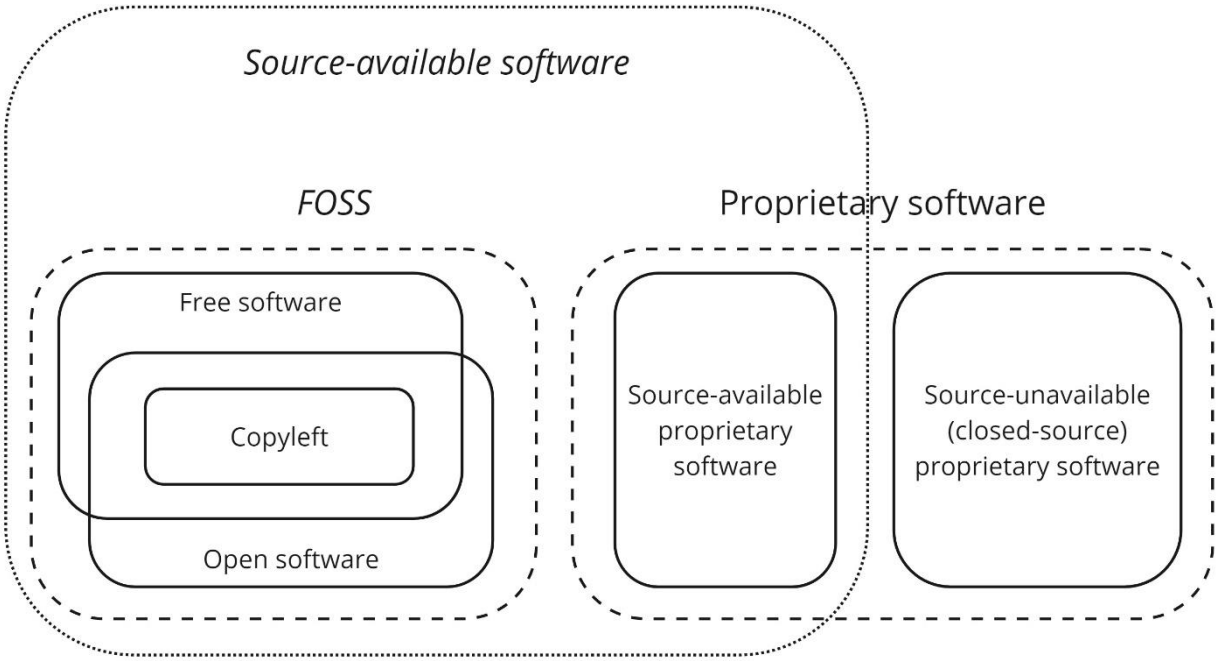

Fig. 3. The relationship between *source-available*, *FOSS* and proprietary software

The main difference between the *FSF* and *OSI* approaches lies in the ideological realm. From the beginning, the *OSI* founders promoted *open source* not as a fair or ethical approach to software development, but as a pragmatic one: open programs are more reliable than closed ones because they involve more people in their development and are not dependent on a specific author or company. In other words, the *FSF* ideology is more radical, emphasizing freedom and fairness in using software, which it partly opposes to the classical concept of exclusive rights[70]. *OSI*, on the other hand, proclaims a non-political approach to development, with the main value being the reliability and efficiency of open programs[71].

But there is no denying that the spread of the ideas behind *FOSS*, however radical they may be, has led to a rapid growth in the popularity of open-source software projects. Free and open-source software is now used in virtually everything from router firmware and the popular *Android* mobile operating system to web servers and supercomputers. The idea of openness is reflected not only in computer programs, but also in the related open standards that underpin the Internet.

Free and open-source programs are distributed under various conditions depending on what their authors have stipulated in the licensing agreement with future users. Standard license agreements are usually used, with several dozen of the most common options[72], each with several versions that vary from one another. Licenses can be more or less radical, particularly regarding conditions for using derivative works.

---

[70] See *McGowan*, p. 303.
[71] History of the OSI. URL: https://opensource.org/history
[72] See Comparison of free and open-source software licenses. URL: https://en.wikipedia.org/wiki/Comparison_of_free_and_open-source_software_licenses

There are *FOSS* licenses that do not impose strict restrictions on derivative works. For example, the author of a derivative program may only be required to identify the original developer, mention the absence of liability, and include a copy of the original license. Thus, by creating a derivative work based on such a program, a developer can close their source code, which is convenient for commercial organizations that do not want to share their intellectual property with the public. Licenses of this type are called permissive; some typical examples are *BSD, MIT*, and *Apache License 2.0.*

The most radical licenses regarding derivative works are called *"copyleft"*. If a developer modifies the source code of a program distributed under a copyleft license, the derivative work must be distributed in a similar way, and its source code must be fully disclosed. In this way, the terms of use of copyleft software are "inherited" by derivative works. A programmer who uses source code distributed under a copyleft license (even a small amount of it) becomes a party to the license agreement and is obligated to fully disclose the source code of the derivative program. This makes the entire product free and open. This is why copyleft licenses are sometimes called viral: they "infect" all derivative works.

Certain copyleft licenses do not “infect” the program if it uses a free component as a separate part, for example, in a dynamic library[73]. This component is loaded on demand by the main program but not integrated into its code. This allows the user to modify, study, and redistribute the library separately from the main program, and when using a free library, the text of the main program may not need to be disclosed (figure 4)[74].

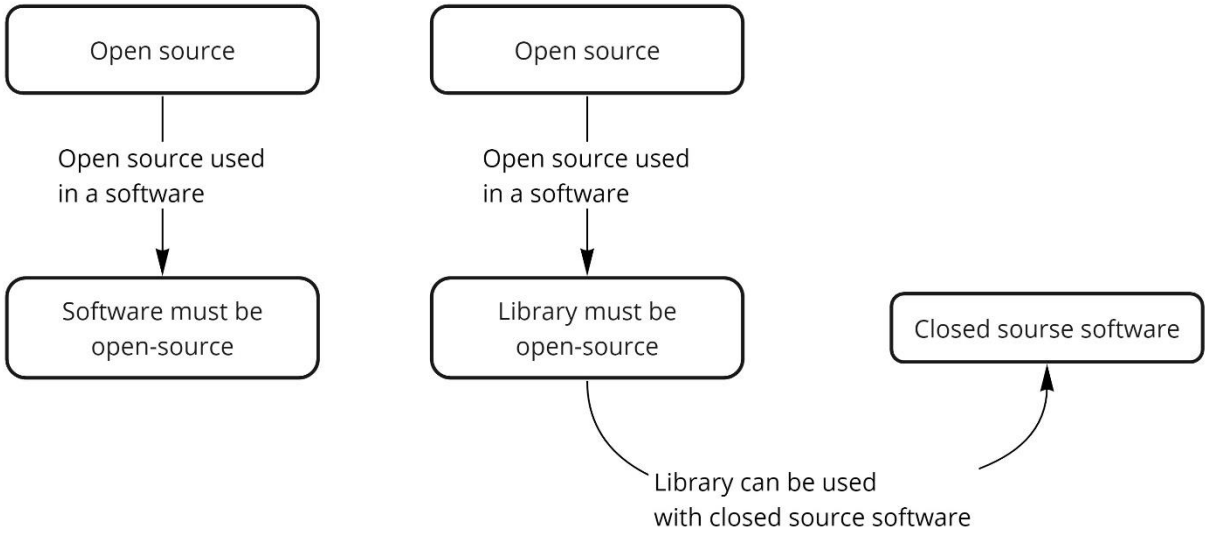

Fig. 4. Inheritance of 'viral' license terms in derivative works

When conducting due diligence on a company's intangible assets (e.g. for M&A purposes), it is important to consider whether open-source code is used in its software and under what license terms. Many popular programs and libraries are distributed under copyleft licenses,

[73] According to this criterion, the authors of the term "copyleft" divide copyleft licenses into strong and weak ones.
[74] It depends mainly on the license terms of a particular library. For example, the GPL prohibits using a free library as part of a closed-source program, while its restricted version, the LGPL, allows it.

including the *Linux* operating system kernel, the *GCC* compiler, and the *WordPress* website builder, all of which use the copyleft GPL license[75]. Even the use of a small fragment of open-source code distributed under a copyleft license may require that all the derivative software's source code be published and distributed under the same copyleft license. This effectively makes the entire derivative product available to everyone and has a major impact on the value of the developing company and its assets.

However, violators of copyleft licenses are rarely held accountable. Since this software is free, the plaintiff in such a dispute is limited in what they can claim in damages, and in many jurisdictions, this limits the financial aspect of potential litigation. However, this does not make such programs public domain, and individual activists together with non-governmental organizations, seek to prosecute the most flagrant violators of copyleft licenses. One of the most famous cases of this kind was the *Free Software Foundation's* lawsuit against the American network equipment manufacturer *Cisco*, which resulted in *Cisco* being forced to disclose the source code of the software used in its devices[76].

In Europe, Germany is the center of most legal disputes related to open-source software. A key figure in shaping the legal practice in this area is Harald Welte, a developer and legal activist who founded the *gpl-violations.org* project. He was able to hold *Sitecom* accountable for violating the use of the *netfilter* (*iptables*) project's source code in its product[77]. Similarly, Welte held *D-Link*, a major telecommunications equipment manufacturer, responsible for using *Linux* source code in its equipment without following the required procedures. As a result, the company not only had to remedy the infringements, but also compensate the plaintiff for the cost of acquiring the device, reverse-engineering the firmware, and hiring a lawyer[78]. Recent cases include *Ximpleware's* lawsuit against Versata and its customers over the use of open-source software, as well as the case between *Linux* developer Christoph Hellwig and VMware (which was lost by the plaintiff) [79].

Many jurisdictions have debated whether a breach of an open license is a breach of the developer's exclusive right or a breach of contractual obligations. In some countries, particularly common law jurisdictions where a license and a contract have different legal natures, this has different consequences[80]. In the US, this issue was resolved in the 2017 case of *Artifex v Hancom*[81], in which the court clarified that an open license could qualify as a contract, and therefore developers can bring corresponding claims against infringers. In France, a relevant claim was

---

[75] GNU General Public License (GPL) version 3.0: https://www.gnu.org/licenses/gpl-3.0.ru.html.
[76] Free Software Fdn., Inc. v. Cisco Sys., Inc., № 1:08-CV-10764 (S.D.N.Y. Dec. 11, 2008). See also: Zenin, I. A., & Meshkova, K. M., *Svobodnaya litsenziya v seti Internet* [Free License on the Internet], *Informatsionnoe Pravo* [Information Law], No. 4, at 8-13 (2011); Sobol, I. A., *Svobodnye litsenzii v avtorskom prave Rossii: monografiya* [Free Licenses in Russian Copyright Law: Monograph] (2014).
[77] Welte v. Sitecom Deutschland GmbH, No. 21 0 6123/04 LG München 1. 19.05.2004.
[78] Welte v. D-Link No. 2-6 O 224/06 LG Frankfurt am M. 06.09.2006.
[79] Peterson, S. K., GPL Enforcement Action in Hellwig v. VMware Dismissed, with an Appeal Expected, *Opensource.com* (Aug. 11, 2016), https://opensource.com/law/16/8/gpl-enforcement-action-hellwig-v-vmware.
[80] Maggs, P. B., The Uncertain Legal Status of Free and Open Source Software in the United States, in F*ree and Open Source Software (FOSS) and other Alternative License Models: A Comparative Analysis* 479 (A. Metzger ed., 2016).
[81] Artifex Software, Inc. v. Hancom, Inc., № 16 Civ. 6982, 2017 WL 1477373 (N.D. Cal. Apr. 25, 2017).

settled in a high-profile dispute between the developer *Entr'Ouvert* and the mobile phone operator *Orange.* The case went to the Paris Court of Appeal[82], which ruled in 2021 that in such cases the developer is only entitled to a contractual claim.

For Russian law, this problem does not seem to be so acute. Open licences are defined in the Civil Code[83]. It explicitly states that an affected developer can claim all the remedies available to copyright holders[84].

For example, a lawyer conducting due diligence on a company's intangible assets (e.g. in an M&A transaction) needs to know whether open-source code was used in the development, under what license it was provided, and how those terms are interpreted from the perspective of applicable law. The terms of use may be "viral", i.e. they may extend to the entire source code of derivative works, even if the developer of the open-source code has not been materially harmed. As a result, the entire resulting program becomes open source, drastically reducing its price and the value of the assets associated with it.

Compatibility of licenses is also of great importance, especially when components used in development are distributed under different licenses. Free and open-source software are distributed under different conditions, which can lead to incompatibilities between components of a program, especially in the case of strong copyleft licenses such as *GPL*. "Dual licensing" is also possible, where different licenses are granted to different users[85]. For example, a commercial license may be aimed at businesses, while a free license is available to private individuals, educational institutions, etc. When the ownership of a program changes, the applicable license may also change.

Note that incorporation of open-source code into a proprietary program can make its legal protection much more difficult. Until recently, Russian law allowed the author of a derivative or composite work to exercise their rights only if the rights of the authors of the original works were respected. Thus, even if the authors of the open component made no claims about its use in a proprietary program, the author of such a program could not exercise their exclusive rights. This year, thanks to the efforts of developer and enthusiast Anton Mamichev, the Constitutional Court ruled that such an interpretation of the law was unconstitutional[86].

2.4. Derivative programs and Versioning

According to Russian law, the exclusive right to a work includes the right to use it[87], and use includes the creation of a derivative work. The author of a derivative work acquires a separate

---

[82] See Cour d'appel de Paris, Pôle 5, Chambre 2, 19 mars 2021, № 19/17493.

[83] Article 1286.1. See also Appeal ruling of the Moscow City Court of 16.07.2015 in case No. 33-25081/2015 (although the dispute was over *Creative Commons* licensing, not Copyleft).

[84] Such claims are listed in Article 1252.

[85] Valimaki, M., Dual Licensing in Open Source Software Industry, *Systemes d'Information et Management*, No. 1, at 63-75 (2003). There are other forms of multiple licensing as well - for example, the CKEditor 4 project is being distributed under three free licenses: *MPL*, *GPL* and *LGPL*. (https://ckeditor.com/legal/ckeditor-oss-license/).

[86] See Decision of the Constitutional Court of the Russian Federation of 16.06.2022 № 25-P "On the case of verification of constitutionality of the item 3 of Article 1260 of the Civil Code of the Russian Federation in connection with the complaint of the citizen A.E. Mamichev".

[87] Item 9 of Article 1270 of the Civil Code. Some argue that the right to create a derivative works does not fall within the scope of exclusive right, as this would conflict with the constitutional freedom of creativity. The owner of

copyright[88]. This concept of a derivative work was developed before the advent of computer programs. It works well for adaptations - translations, screen adaptations, stage adaptations, etc. But it is more complicated with programs, which change much more frequently. Each such modification formally creates a new copyrighted object, albeit legally related to the original, and, if the modification is not authorized, this infringes the exclusive right to the original program.

The only condition for the creation of a new object of copyright is the creative nature of the remake, in which case its creator[89] is considered to be a co-author. For this reason, compiling a program (converting the source code into object code) is not a creative act of authorship. Although the compiler may leave significant traces of its work in the object code, its developers do not acquire any rights in the compiled program. Likewise, under the direct provision of the Civil Code[90], the author of a programming language used in the creation of a program does not acquire copyright in the program[91].

To sum up, creative adaptation of a program creates a derivative work, which is protected as an independent object of copyright[92]. Nevertheless, there is no definition of the minimum degree of modification required to grant legal protection to the derivative work and the threshold at which the work stops being derivative and becomes separate and original. The Civil Code defines "modification of a computer program or database" as "any changes made to it"[93], but this does not mean that even minor changes to the source code create a derivative work[94]. And vice versa, a program that is substantially (even up to 88%[95]) identical to the original may still be considered a derivative work if the modification process involves a creative contribution.

The Civil Code contains a special provision for computer programs, that excludes adaptation from the scope of modification[96]. Adaptation refers to changes made solely to enable the program to operate on specific technical means of the user or under the control of specific user programs. It involves recompilation of the software, including for porting purposes (to guarantee operation on specific software or hardware). The Civil Code does not consider it an infringement of exclusive rights even if the transfer of the program to a remote platform (e.g. transfer of a game

---

the original work can only allow or prohibit the *further use of the derivative work*, but not its creation. See *Vitko V.S.* On the Attributes of the Concept of a Derivative Work [*O priznakakh ponyatiya 'proizvodnoe proizvedenie'*]. Intellectual Property. Copyright and Related Rights [*Intellektualnaya sobstvennost'. Avtorskoe pravo i smezhnye prava*]. 2018. No. 9. P. 37–54.

[88] Article 1260 of the Civil Code.

[89] See *Eliseev V.I.* The Right to Process Works under Russian Law [*Pravo na pererabotku proizvedeniy po rossiiskomu zakonodatelstvu*]. The Herald of Moscow University. Law Series [*Vestnik Moskovskogo universiteta. Seriya 'Pravo'*]. 2017. №1. P. 93–104.

[90] Item 5 of Art. 1259.

[91] See *Eliseev V.I.* Civil Law Regime for Derivative Intellectual Property Rights: A PhD Thesis in Law [*Grazhdansko-pravovoi rezhim proizvodnykh ob'ektov intellektualnykh prav: dis. … kand. yurid. nauk*]. Moscow, 2017. 245 p. See also: *Akhobekova R.A.* Interpretation of Reprocessing (Modification) of Computer Programs in Court Practice [*Tolkovanie pererabotki (modifikatsii) program dlya EVM v sudebnoi praktike*]. Intellectual Property. Copyright and Related Rights [*Intellektualnaya sobstvennost'. Avtorskoe pravo i smezhnye prava*]. 2020. No. 5. P. 27–38.

[92] Article 1260.4 of the Civil Code.

[93] Subitem 9 of Article 1270.

[94] See *Akhmedov G.A.* Challenges in Regulating Software Modification [*Problemy regulirovaniya modifikatsii programmnogo obespecheniya*]. Journal of the Court of Intellectual Property Rights [*Zhurnal Suda po intellektualnym pravam*]. 2020. No. 2. P. 20–26.

[95] See Decision of the Court of IP Rights of 21.11.2016 in case No. A56-21040/2015.

[96] Item 9 of article 1270. In fact, Russian law does not distinguish between software modification and reverse engineering, and in practice lawyers usually conclude that any modification results in the creation of a derivative work.

from *PC* to *Android*) requires significant changes to the source code. In practice, it is the purpose of the changes made to the program, rather than their scope, that distinguishes between modification and adaptation[97].

Existing rules on derivative works make it difficult to determine rights to versions of a work. Software development uses versioning, where each subsequent version of the source program is a modified (and usually supplemented) previous version. Programs are usually numbered using digital indices, where the first digit usually indicates the core content of the program, the second digit indicates major changes, and the third digit indicates minor current modifications. According to this system, version 2.3.0 is a modified version of the first program (1.0 → 2.0), which has already been substantially updated three times (2.0.0 → 2.3.0). However, according to a literal interpretation of the law, each version of the program is a derivative of the original[98].

Therefore, each updated version of the Program is a separate work derived from the original version. This means that if the User was granted the right to use Program version 1.0, after updating (automatically or by User action) to version 1.1, the User starts using a new Program to which he has no rights. Only the old version of the program 1.0 is covered by the license agreement signed by the user. For each new version of the program, a new license agreement must be concluded[99].

There have been several proposals to address this issue. For example, in 2012, as part of the reform of the Civil Code, it was proposed to include in the definition of a computer program "materials created during its subsequent improvement". In this case, subsequent versions of the program would not constitute separate works but would be included in the original object of intellectual property rights[100]. However, due to its size, the bill to amend the Civil Code has been divided into parts and adopted one by one, and, as a result, the lawmakers have not yet reached the required article.

The Skolkovo Centre for Competence in Regulatory Support for the Digital Economy proposed[101] a slightly different way to solve this problem in 2019. They suggested introducing a definition of "program version" into the Civil Code:

> "The modified version of a computer program or database, identified by a unique symbol used by the author or other rightsholder, is considered a version of the program or database. Copyright applies to all versions of the respective computer program or database.
>
> When the right to use a computer program or database is granted by a license agreement, the licensee has the right to use the version of the computer program or database specified in the agreement, as well as versions obtained as a result of updates... The

---

[97] See *Korneev V.A.* Computer Programmes, Databases and Integrated Circuit Topologies as Intellectual Property Rights [*Programmy dlya EVM, bazy dannykh i topologii integralnykh mikroskhem kak ob'ekty intellektualnykh prav*]. Moscow, Statut, 2010. 104 p.

[98] This position is supported by court practice, See Decision of the Court of IP Rights of 02.02.2016 in case No. A63-1829/2015.

[99] Most copyright holders do just that: after each update, even if it is free, the user has to accept the user agreement all over again.

[100] Draft Federal Law No. 47538-6 "On Amendments to Part One, Part Two, Part Three and Part Four of the Civil Code of the Russian Federation, as well as to some legislative acts of the Russian Federation" (as amended on 03.04.2012 and introduced to the State Duma).

[101] The bill is available at: https://my.sk.ru/foundation/legal/m/sklegal04/22873/download.aspx.

author or other rightsholder has the right to independently determine the time at which versions of the computer program or database are made available to the licensee before they become available...".

Thus, if passed, the bill would automatically give the licensee the right to use program updates without having to renegotiate the license agreement. However, there is a conceptual flaw in both the 2012 and 2019 bills. They allow the rightsholder to determine the moment of appearance of a new object of copyright without creating any new code. This contradicts the generally accepted approach that exclusive rights arise with the creative action[102].

In the real world, things can get complicated when several groups of developers are working on a program at the same time. For example, if they have different ideas about the project's future development and continue to work separately. As a result, the program's version tree branches out, and each branch is worked on by a different group of developers. This is a common scenario, especially in the free software community, where anyone can contribute to the project. A good example of this is the development tree of *Unix*-like operating systems, from *Unix* to *Linux* to *MacOS* (Fig. 5).

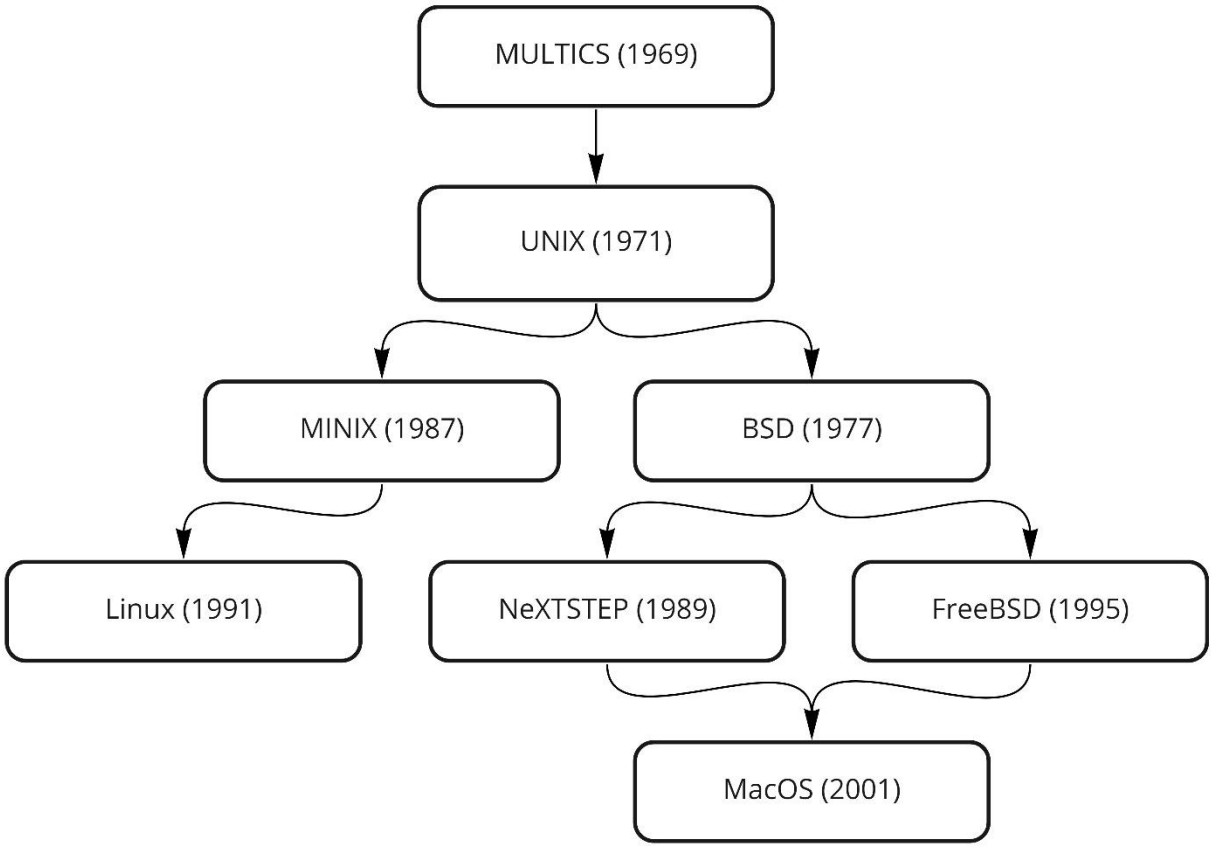

Fig. 5. Evolution of *Unix*-like operating systems

[102] See *Novoselova L.A.* Creation of a Result of Intellectual Activity as a Conferring Legal Fact [*Sozdanie rezultata intellektualnoi deyatelnosti kak pravoporozhdayuschiy yuridicheskiy fakt*]. Russian Judge [*Rossiiskiy sid'ya*]. 2016. No. 5. P. 8–12.

What should be done if a rightsholder transfers exclusive rights of different versions of a program to different parties? For example, a development company has granted exclusive rights of version 2.1 to a customer. Can the company create version 3.0 of the program? Can the customer modify “his” version of the program (e.g. by creating version 2.2), if it affects the shared core? The main problem here is that such versions usually contain a significant amount of shared source code, which allows the rights holders to restrict each other's actions.
Let's say the developer of version 2.0 of the program created version 2.1 based on it. Version 2.1 contains 40% custom code and 60% code shared with version 2.0 (the "core" of the program) (Fig. 6).

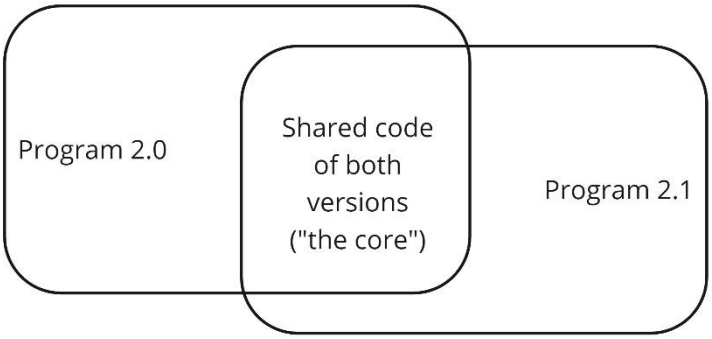

Fig. 6. Architecture of a derivative program that shares code with an original one

The developer has transferred the rights to version 2.1 of the program to the customer by an agreement that clearly and fully transfers the exclusive rights (Figure 7). It appears that the customer has full ownership of version 2.1 of the program, including the right to create derivative works based on it and to prevent others from using the program without permission.

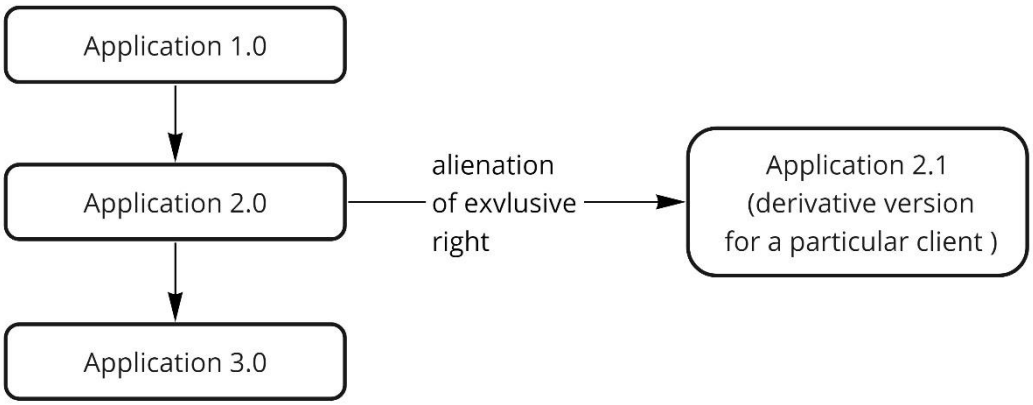

Fig. 7. Evolution of the disputed application

Despite transferring the exclusive rights of program version 2.1 to the client, the developer still holds the rights to the original work - program version 2.0. This includes the right to permit or prohibit the creation of derivative works based on it. If the client creates a new version of their existing program (program version 2.2), is it considered only a derivative of program version 2.1 or also a “second derivative” of program version 2.0? Can the client modify the core part of the program - the shared 60% of the source code? Logically, if the original developer can prohibit

modifications, he can also prohibit the distribution of the core part of the program to third parties, since both acts would infringe their rights. However, this approach contradicts the concept of a program as a separate object of exclusive rights.

In practice, when a mass-market commercial program is designed for modification and customization (such as game engines, map editors, etc.), the license agreement typically outlines the conditions for creating derivative works. There are significant risks for both the developer and the client if they have not carefully considered this issue. The developer may forbid the client from using the derivative program, while the client may claim ownership of the entire program to obstruct the developer[103].

3. Software made for hire in Russian law

In practice, there are significant challenges with regard to works made for hire (works created within the scope of the author's duties as an employee). This concept involves both contract law and labor law. The boundaries between them are so interconnected that the contractual issue of the transfer of exclusive rights is determined on the basis of labor law. These labor law provisions ensure greater protection for the employee as the weaker party in the relationship, and the employer bears any resulting copyright risks.

3.1. Computer programs as works made for hire

Why is the work for hire regulation not a good fit for computer programs? There are two important factors that distinguish programs made for hire from similar works, such as photographs or texts.

First, programming is typically a direct function of the employed person doing their job. This makes programmers' professions similar to other specialized professions like photographers, journalists, or copywriters. However, unlike other professions, programming is not always delivered piecemeal. For example, when working with an in-house illustrator, the employer can accept each work when it is completed. On the other hand, the development of a program is usually a continuous process in which employees add to and edit the source code without interrupting the workflow. Incorporating formal legal procedures into this process becomes more difficult.

Secondly, program development is usually collaborative and it's difficult to separate the parts created by a particular developer from the rest. For example, a photograph taken by a particular photographer or text written by a particular journalist can be separated from the content of a magazine or website. However, separating the lines of source code written by a particular developer from the rest of the program is much more difficult. In this respect, the job of a developer is similar to that of a film producer.

The regulation of work for hire varies considerably from one country to another. To illustrate the specificity of its regulation, we have defined three main questions. They determine how clear and unambiguous this issue is regulated in a given jurisdiction and whose interests it protects more: those of the employer or those of the employee.

---

[103] See example in Decision of the Court of IP Rights of 21.03.2017 in case No. A40-154016/2014.

1. Should the employee specifically express their will to transfer the rights to the work for hire to the employer and, if so, how?
2. How much of the employer's resources is sufficient for an employee to spend on the development of the work for hire in order to acquire the rights to it?
3. Should the employee receive compensation in addition to their salary for transferring the rights to the program?

Russian regulations are ambiguous on these issues. On one hand, they seek to protect the employee by imposing additional obligations on the employer to formalize the rights to the program and to pay compensation. On the other hand, these requirements are difficult to meet in practice during the development of the program, so that instead of legal protection, the only result for the parties is uncertainty and additional risks.

Before 2008, when the relevant (fourth) part of the Civil Code came into force, works for hire were considered to be works "created in the course of performing job duties or a task assigned by the employer." [104] The "task assigned by the employer" formula was vague and open to interpretation: it could include employers' internal policies and even a correspondence with the employee. This flaw became particularly apparent after the high-profile dispute between the *Rambler* holding and its former employee Igor Sysoev. Sysoev developed the well-known *Nginx* web server application while working for Rambler in 2002, and then sold it to *F5 Networks* in 2019 for $670 million[105].

Since 2008, works made for hire in Russia have been primarily governed by the Civil Code[106]. It recognizes a work as made for hire if (1) the author is in an employee and (2) the creation of the work is part of their job duties. Under these conditions the exclusive right arising from the employee[107] is automatically transferred to the employer the moment after the work is created. The employer retains the exclusive right if, within three years, it begins to use the work, transfers it to another person, or notifies the author that the work is confidential.

The previous option to create a work as part of "a task assigned by the employer" was not included into the Civil Code, so there is no need to provide an official task as a separate document to transfer rights for the work. This has significantly reduced the amount of paperwork involved in developing software.

The current regulation of works for hire in Russia has only been in place for 15 years, and related court practice is still developing. A landmark case is that of developer Anton Mamichev, who has been in litigation with his former employer for several years. The dispute concerns the ownership of a program developed outside the scope of his job duties. The case has been in litigation for more than four years and has been the subject of ten technical and economic investigations and 23 case files. Mamichev won the first case, but his former employer won the following

---

[104] See Article 14 of the Law "On Copyright and Related Rights" dated 09.07.1993 № 5351-1, Article 12 of the Law "On Legal Protection of Programs for Electronic Computers and Databases" dated 23.09.1992 № 3523-1.
[105] Rambler vs Nginx. Lawyers Tell Us Who's Right, *The Bell* (Dec. 13, 2019), https://thebell.io/rambler-protiv-nginx-kto-prav-otvechayut-yuristy.
[106] Article 1295.
[107] Article 1257.

appeals[108]. However, in June 2022, the Constitutional Court sent the case back for reconsideration[109].

3.2. Scope of "job duties" under Russian law

If an employee creates a work outside of their job duties, even if it is within the scope of a task assigned by management, copyright to that work does not transfer to the employer. For example, if a programmer is hired as a "Java Developer" and their employment contract only specifies *Java* programming language, but they write source code in *Python* to complete a task, the rights to that code will not be transferred to the employer. This approach may appear to be formalistic, but it is a working method and it causes the employer to bear additional risks.

The content of the work is not determined by contract law but by labor law, which regards the employee as the weaker party[110]. Therefore, in the event of a dispute, it is the employer who has the burden of proof that the work falls within the scope of the employee's job duties. For this purpose, the provisions of the employment contract[111] and internal work regulations (internal code of conduct[112], service instructions[113], etc.) will be evaluated. If an employee creates a work outside of their job, even if it is done with the employer's equipment and at the employer's request, such a work cannot be considered "work made for hire"[114].

If this question arises in a commercial court, the employer can refer to the practice of the Court of IP Rights (which is part of commercial courts system). It has stated that the determination of an employee's job function should not be based only on the wording of the employment documents, but should also take into account other circumstances, such as

1. the relation between the business of the employer and the field in which the work has been created;
2. the limits of the employee's job duties;
3. the location of the work performed;
4. the equipment and tools used to create the work;
5. the employer's control over the creation of the work;
6. the purpose of creating the work;
7. the subsequent conduct of both the employee and the employer, etc.[115]

---

[108] Anton Mamichev has collected all the procedural documents and opinions on the case on a separate website: emmcase.mamichev.ru. We leave it to the reader to familiarize himself with the arguments of the parties and the decisions of the courts.
[109] See Decision of the Constitutional Court of the Russian Federation of 16.06.2022, № 25-P.
[110] This doctrine was enshrined in the rulings of the Constitutional Court of the Russian Federation from 15.03.2005 № 3-P, from 25.05.2010 № 11-P, from 19.12.2018 № 45-P and other acts.
[111] Ruling of the Sixth Court of Common Pleas on 05.07.2022 in Case No. 88 12243/2022, 2-3138/2021.
[112] Ibid.
[113] Ruling of the Moscow City Court of 28.03.2013 in case № 11-9941/2013.
[114] See section 104 of Resolution of the Plenum of the Supreme Court of the Russian Federation of 23.04.2019 No.10.
[115] Decision of the Court of IP Rights of 29.03.2019 in case No. A40-256611/2017, of 15.03.2016 in case No. СИП-818/2014, of 16.05.2016 in case No. СИП-167/2015; Decision of the Presidium of the Court of IP Rights of 07.08.2015 in case No. СИП-253/2013.

However, only a small number of such disputes end up in arbitration; they are mostly dealt with by courts of general jurisdiction[116]. Therefore, if an employee in Russia creates a work outside the scope of their job duties, it will likely not be considered a work for hire, and the employer will not obtain the copyright under the employment contract. A separate civil law contract with the employee is required to for transfer of rights. The more equitable solution would be to transfer the right to the employer, subject to an obligation to provide additional remuneration for the employee.

Anglo-Saxon legal family jurisdictions take a similar approach: the use of employer resources by an employee is considered sufficient grounds for the employer to have exclusive rights to the work produced. For example, in the case of *Missing Link Software v. Magee*[117], a British developer created software source code during non-working hours and partly on his own equipment, but this did not help him retain copyright in the program. An American court made a similar decision in *Miller v. CP Chems., Inc.*[118]: a quality controller in a chemical laboratory was required to fill out electronic forms which he wrote a special computer program for during non-working hours. The court ruled that the copyright in the program belonged to his employer.

However, even in these jurisdictions intellectual property does not transfer to the employer if it was clearly created outside the scope of the corresponding job duties. In the United States, the Copyright Act provides that a *work made for hire* must be created within the scope of an employee's job duties[119]. The conditions for enforcing this right (including the burden of proof and possible exceptions to the rules) are set out in state labor laws[120].

In the UK, there have been cases where engineers and doctors have written books based on material from their work, but even though they were written during working hours and with the company's resources, the companies have not been able to obtain the rights to the books. This was the outcome in the cases of *Stevenson, Jordan & Harrison Ltd v. MacDonald & Evans*[121] and *Noah v. Shuba*[122]. In the first case, an engineer wrote a book based on his conference presentations and work at the company, while in the second one, a doctor wrote a book using the resources of his hospital. Despite using company resources and working on the books during working hours, the companies were unable to claim ownership of the books as intellectual property.

3.3. Remuneration for a program made for hire

If an employer has acquired the rights to a work made for hire, started using it, decided to keep it secret, or transferred it to another person, The Civil Code grants the employee the right to compensation[123]. However, if the employee has not received compensation, they are not entitled to claim the work, as the rights to the work have already passed to the employer[124]. In this situation,

---

[116] See section 3 of Resolution of the Plenum of the Supreme Court of the Russian Federation of 23.04.2019 No.10.
[117] FSR 361 (1989).
[118] 808 F. Supp. 1238, 1239 (D.S.C. 1992).
[119] 17 U.S.C. § 101.
[120] See Graves, C. T., Is the Copyright Act Inconsistent with the Law of Employee Invention Assignment Contracts?, 7 *NYU J. Intell. Prop. & Ent. L.* 8 (2018).
[121] 69 RPC 10, 10 TLR 101 (1952).
[122] FSR 14 (1991).
[123] Article 1295.
[124] In this sense, the employee's right to remuneration is analogous to a payment for the alienation of an exclusive right, rather than a payment for use under a license agreement.

the employee can only go to court with a claim for payment. This raises the question: how does the remuneration relate to the employee's salary, especially in cases where the employee's duties directly involve the creation of works for hire?

Employment contracts often state that the employee's salary already includes compensation for the transfer of intellectual property. Is this sufficient from a legal point of view? The case law on this issue was not clear until the recent decision of the Supreme Court[125], but now we can conclude that it is not. An employer is obliged to include into a contract with an employee not only the terms and conditions of remuneration, but also the size, conditions and method of payment of this remuneration[126]. Since it is not possible to specify these details for each remuneration in advance in the employment contract, we can conclude that, from the perspective of the law, the employer must make a separate remuneration and file a separate document for each.

For software developers, paying a separate fee for each piece of work seems like a meaningless formality. In their case, the creation of work is the subject of the employment contract. Why pay renumeration if the employee is paid for developing programs regardless? Due to this logic, the payment of renumeration is a formality even for those employers who scrupulously comply with the law.

Surprisingly, there was no provision for additional remuneration for the creation of works for hire in Soviet labor legislation, which was generally progressive for its time. This was justified by the fact that the author already received a salary, and payment of an author's fee would be a second form of compensation[127].

Current statutory and case law require employers to pay employees for the use of their work for hire, even if the employee's job duties consist solely of creating such work. The result is that employers pay compensation amounts that are usually small and infrequent, and sometimes do it only when the contract is terminated. Employees do not gain any additional guarantees or protection of their rights from this formal requirement[128]. Both sides see the process as a formality[129].

Conclusion

In this article, we have discussed a number of problems in implementing and protecting intellectual property rights for computer programs. As it turns out, the legal status of software has many peculiarities, both in substantive and procedural aspects. Difficulties exist in protecting rights to computer programs and in identifying plagiarism of the original text. The mechanisms provided by legislation and judicial practice, such as depositing programs with Rospatent and disclosing the source code as evidence, are insufficient and do not take into account the technical aspects of software development and use. The issue of versioning of computer programs is also insufficiently regulated (although amendments to the law in this regard have been proposed on several

---

[125] See Ruling of the Supreme Court of the Russian Federation of 05.06.2020 No. 78-KG20-1, 2-5974/2018.
[126] According to Article 1295 of the Civil Code.
[127] See Gavrilov, E. P., *Avtorskoe pravo. Izdatelskie dogovory. Avtorskiy gonorar* [Copyright. Publishing Contracts. Copyright Royalties] 29 (Yuridicheskaya literatura, Moscow 1988).
[128] See also Maltsev, N. M., Pravo na voznagrazhdenie za sluzhebnoe proizvedenie [Right to Remuneration for an Official Work], *Patenty i litsenzii. Intellektualnye* prava [Patents and Licenses. Intellectual Rights], No. 4, at 57-66 (2019).
[129] Cf. Kasyan, V., & Gareev, M., Advice on the Protection of Exclusive Rights to Software, *Advokatskaya Gazeta (Mar. 16, 2020)*,https://www.advgazeta.ru/ag-expert/advices/sovety-po-zashchite-isklyuchitelnykh-prav-na-programmnoe-obespechenie/.

occasions), and there is a lack of case law in the area of open-source software. These problems can be resolved by the competent actions of the high courts – the recent review of the Court of IP Rights is a good example.

We have also addressed the important issue of employees who create software (computer programs made for hire). It is important to keep a balance between the interests of employees (developers) and employers (IT companies). Unfortunately, our legislation in this area (at least compared to foreign legislation) appears to be biased in favor of employees, which creates additional risks for companies involved in developing software. The situation is further complicated by conflicting case law, especially regarding the remunerations to authors of programs made for hire.

Of course, we have not covered all the relevant issues. There are numerous problems related to license agreements and the transfer of rights to programs, the identification of programs in contracts, and the determination of the origin of programs (for example, for public procurement purposes). Following the decision of the Constitutional Court in the case of Anton Mamichev, the legislator is required to revise the regulations governing composite programs. This year it was announced that the compulsory licensing mechanism would be extended, and computer programs could be included. It is our hope that legal scholars will also find doctrinal solutions to these problems.

References

1. Ahmedov, G. A., *Problemy regulirovaniya modifikatsii programmnoho obespecheniya* [Problems of Regulating Software Modification], *Zhurnal Suda po intellektualnym pravam* [Journal of the Court of Intellectual Property Rights], No. 2 (2020).
2. Akhobekova, R. A., *Perspektivy ispol'zovaniya sudebnogo opyta SSHA v reshenii voprosa o predmete pravovoy okhrany programm dlya EVM* [Prospects for Using US Judicial Experience in Addressing the Issue of the Subject of Legal Protection of Computer Programs], *Zakon* [The Statute], No. 5 (2021).
3. Akhobekova, R. A., *Tolkovanie pererabotki (modifikatsii) programm dlya EVM v sudebnoy praktike* [Interpretation of Processing (Modification) of Computer Programs in Judicial Practice], *Intellektual'naya Sobstvennost'. Avtorskoe Pravo i Smezhnye Prava* [Intellectual Property. Copyright and Related Rights], No. 5 (2020).
4. Vitko, V. S., *O priznakakh ponyatiya «proizvodnoe proizvedenie»* [On the Signs of the Concept "Derivative Work"], *IS. Avtorskoe Pravo i Smezhnye Prava* [IS. Copyright and Related Rights], No. 9 (2018).
5. Gavrilov, E. P., *Avtorskoe pravo. Izdatel'skie dogovory. Avtorskiy gonorar* [Copyright. Publishing Contracts. Author's Fee], Moscow: Yuridicheskaya Literatura (1988).
6. Eliseev, V. I., *Grazhdansko-pravovoy rezhim proizvodnykh ob'ektov intellektual'nykh prav* [Civil-Law Regime of Derivative Intellectual Property Objects] (PhD thesis., Moscow 2017).
7. Eliseev, V. I., *Pravo na pererabotku proizvedeniy po rossiyskomu zakonodatel'stvu* [The Right to Process Works According to Russian Legislation], *Vestnik Moskovskogo Universiteta. Seriya "Pravo"* [Bulletin of Moscow University. Law Series], No. 1 (2017).
8. Edward J. Imwinkelried, Computer Source Code: A Source of the Growing Controversy Over the Reliability of Automated Forensic Techniques, 67 *DePaul L. Rev.* 1 (2017).
9. David McGowan, Legal Implications of Open-Source Software, 2001 *U. Ill. L. Rev.* 1.
10. Korneev, V. A., *Programmy dlya EVM, bazy dannykh i topologii integral'nykh mikroskhem kak ob'ekty intellektual'nykh prav* [Computer Programs, Databases, and Topologies of Integrated Circuits as Intellectual Property Objects], Moscow (2010).
11. Peter B. Maggs, The Uncertain Legal Status of Free and Open Source Software in the United States, in *Free and Open Source Software (FOSS) and other Alternative License Models: A Comparative Analysis* (Axel Metzger ed., Springer 2016).
12. Maltsev, N. M., *Pravo na voznagrazhdenie za sluzhebnoe proizvedenie* [The Right to Remuneration for an Official Work], *Patenty i Litsenzii. Intellektual'nye Prava* [Patents and Licenses. Intellectual Rights], No. 4 (2019).
13. Kenneth Nichols, *Inventing Software: The Rise of "computer-related" Patents* (Praeger 1998). Lydia Pallas Loren & Andy Johnson-Laird, Computer Software-Related Litigation: Discovery and the Overly-Protective Order, 6 *Fed. Cts. L. Rev.* 75 (2012).
14. Novoselova, L. A., *Sozdanie rezul'tata intellektual'noy deyatel'nosti kak pravoporozhdayushchiy yuridicheskiy fakt* [The Creation of Intellectual Activity Result as a Legal Fact], *Rossiyskiy Sudy'a* [Russian Judge], No. 5 (2016).

15. Novoselova, L. A., *Ob osobennostyakh nekotorykh pravoporozhdayushchikh faktov v patentnom prave* [On the Features of Some Law-Generating Facts in Patent Law], *Zhurnal Suda po Intellektual'nym Pravam* [Journal of the Court of Intellectual Property Rights], No. 12 (2016).

16. Savel'ev, A. I., *Aktual'nye voprosy sudebnoy praktiki v sfere oborota programmnogo obespecheniya v Rossii* [Actual Issues of Judicial Practice in the Field of Software Turnover in Russia], *Vestnik VAS RF* [Bulletin of the Supreme Arbitration Court of the Russian Federation], No. 4, pp. 4-36 (2013).

17. Sobol', I. A., *Svobodnye litsenzii v avtorskom prave Rossii: monografiya* [Free Licenses in Copyright Law of Russia: Monograph], Moscow: Justiciinform (2014).Tait Graves, Is the Copyright Act Inconsistent with the Law of Employee Invention Assignment Contracts?, 7 *NYU J. Intell. Prop. & Ent. L.* 1 (2018).

18. Mikko Valimaki, Dual Licensing in Open Source Software Industry, 8 *Systemes d'Information et Management* 1 (2003).

19. Zenin, I. A., & Meshkova, K. M., *Svobodnaya litsenziya v seti Internet* [Free License on the Internet], *Informatsionnoe Pravo* [Information Law], No. 4 (2011).