2020

# A Quarter Century of International Copyright on Software

Marketa Trimble
*University of Nevada, Las Vegas -- William S. Boyd School of Law*

# A Quarter Century of International Copyright on Software

MARKETA TRIMBLE*

*Abstract*

*A quarter of a century ago, under the new TRIPS Agreement, copyright protection for software became the international norm. Whether copyright is in fact an appropriate form of protection for software was and still is debated; the issues raised in the ongoing high-profile Google v. Oracle dispute, now before the U.S. Supreme Court, are among the continuing and significant problems that the extension of copyright to software have generated. This article reviews the coexistence between software and copyright law; it focuses on their mutual effects and the adjustments in each that resulted from their coexistence. The article notes the technological changes that coincided with or followed the adoption of the international norm and that significantly affected the coexistence between software and copyright law.*

*Samuel S. Lionel Professor of Intellectual Property Law, William S. Boyd School of Law, University of Nevada, Las Vegas. The author thanks for their comments and suggestions Adela Faladova, Jorge Contreras, Christopher Heath, Jens Hillebrand Pohl, Jerome Reichman, and Pamela Samuelson. For the research support that they provided the author thanks Susan Wainscott (Engineering Librarian at UNLV), James Rich (Research Librarian and Assistant Professor at the Wiener-Rogers Law Library at UNLV), and Zahava Lieberman (J.D. '21). The author is indebted to Gary A. Trimble for his invaluable editing suggestions.

## TABLE OF CONTENTS



## INTRODUCTION

It has been a quarter of a century since copyright protection for software became the international norm. Article 10 of the TRIPS Agreement, which was concluded in 1994 and became effective on January 1, 1995, mandates that all members of the WTO extend their copyright laws to cover "computer programs, whether in source or object code."[1] The anniversary is an opportunity to evaluate the coexistence between software and copyright, as critics of the extension of copyright protection to software continue to challenge it. The extension continues to receive major attention because of high-profile disputes concerning copyright and software such as the *Oracle v. Google* dispute[2] now before the U.S. Supreme Court.[3]

It was not a foregone conclusion that copyright would protect software. The adoption of TRIPS Article 10 was preceded by approximately three decades of debates among experts and policy makers regarding the suitability of copyright for protecting investments in, and incentivizing of, the creation of software.[4] Before the TRIPS Agreement was concluded some

1. Marrakesh Agreement Establishing the World Trade Organization, Apr. 15, 1994, 1869 U.N.T.S. 299, Annex 1C, Agreement on Trade-Related Aspects of Intellectual Property Rights [hereinafter TRIPS Agreement], art. 10(1). Since July 29, 2016, the number of countries–WTO members is 164. *Members and Observers*, WORLD TRADE ORGANIZATION, https://www.wto.org/english/thewto_e/whatis_e/tif_e/org6_e.htm (last visited Jan. 2, 2020).

2. Oracle America, Inc. v. Google LLC, 886 F.3d 1179 (Fed. Cir. 2018), *cert. granted*, No. 18-956, 2019 S.Ct. WL 6042317 (Nov. 15, 2019). *See infra* section II.D.

3. Google LLC v. Oracle America, Inc., No. 18-956, 2019 S.Ct. WL 6042317 (Nov. 15, 2019).

4. *See, e.g.*, MICHAEL S. KEPLINGER, LEGAL PROTECTION FOR COMPUTER PROGRAMS: A SURVEY AND ANALYSIS OF NATIONAL LEGISLATION AND CASE LAW (1984); Stephen Breyer, *The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs*, 84 HARV. L. REV. 281, 340–50 (1970); Paul Goldstein, *Infringement of Copyright in Computer Programs*, 47 U. PITT. L. REV. 1119, 1120–21 (1986); Arthur R. Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?*, 106 HARV. L. REV. 977, 985–91 (1993); J.H. Reichman, *The TRIPS Component of the*

experts and policy makers had proposed that a sui generis form of protection, specifically tailored to software, would be a better suited form of protection.[5] The adoption of TRIPS Article 10, and later the corresponding provision in the 1996 WIPO Copyright Treaty,[6] essentially cemented copyright protection for software in national laws.[7] While the two provisions made future proposals for sui generis software protection futile, the debates regarding the suitability of copyright protection for software have continued.[8]

Early on, leading commentators predicted and even observed that the expansion of copyright protection to software would result in adjustments to copyright law and its application that were necessitated by the specificities of software as a protectible subject matter.[9] As this article illustrates, the effects of software on copyright law caused a number of changes to the law. Copyright law has affected software as well. Although the adjustments to copyright law have not eliminated all of the concerns of the critics of copyright protection for software, they have made copyright protection more suitable for software, and vice versa.[10]

The effects of software on copyright law and copyright law on software did not begin with the TRIPS Agreement. Before becoming an international norm in the TRIPS Agreement, copyright protection for software had been adopted into the national laws of many countries. In the United States, the U.S. Copyright Office began registering computer programs in 1964,

---

*GATT's Uruguay Round: Competitive Prospects for Intellectual Property Owners in an Integrated World Market*, 4 FORDHAM INTELL. PROP., MEDIA & ENT. L.J. 171, 229–35 (1993). *See* Gerardo Con Díaz, *The Text in the Machine: American Copyright Law and the Many Natures of Software, 1974–1978*, TECH. AND CULTURE 753, 762–71 (2016) (discussing the debates before and during the hearings of the National Commission on New Technological Uses of Copyrighted Works).

5. *See, e.g.*, Proposal by the Ministry for International Trade and Industry (MITI), Japan, 1982; Draft Treaty for the Protection of Computer Software, Feb. 24, 1983, WIPO Pub. No. LPCS/II/3; Peter S. Menell, *Tailoring Legal Protection for Computer Software*, 39 STAN. L. REV. 1329, 1371–72 (1987). *See generally* Pamela Samuelson, Randall Davis, Mitchell D. Kapor, & J.H. Reichman, *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994).

6. World Intellectual Property Organization Copyright Treaty, Dec. 20, 1996, 2186 U.N.T.S. 121 [hereinafter WIPO Copyright Treaty], art. 4.

7. Theoretically, countries could agree to change international treaties with respect to copyright protection for software, but a re-opening of treaty negotiations is unlikely; changing the treaties could be detrimental to other provisions of the treaties that some countries want to maintain. Also, a country could decide not to follow the treaties, though a violation of a country's international obligations could have repercussions for that country, particularly since the TRIPS Agreement comes with the—potentially effective—WTO dispute resolution mechanism. *See* TRIPS Agreement, *supra* note 1, at art. 64.

8. *See, e.g.*, Jacqueline D. Lipton, *IP's Problem Child: Shifting the Paradigms for Software Protection*, 58 HASTINGS L. J. 205, 250 (2006) (suggesting "scaling back some of the current legal protections" under copyright law); Pamela Samuelson, *The Uneasy Case for Software Copyrights Revisited*, 79 GEO. WASH. L. REV. 1746, 1776–81 (2011).

9. *See, e.g.*, Breyer, *supra* note 4, at 350, 370 ("[a]nd, if need is shown, efforts should be made to tailor protection to minimize the harms that copyright may cause"); Jane C. Ginsburg, *Four Reasons and A Paradox: The Manifest Superiority of Copyright Over Sui Generis Protection of Computer Software*, 94 COLUM. L. REV. 2559, 2572 (1994); Goldstein, *supra* note 4, at 1121; Raymond T. Nimmer & Patricia Krauthaus, *Classification of Computer Software for Legal Protection: International Perspectives*, THE INT'L LAW. 735, 748 (1987); J.H. Reichman, *Legal Hybrids Between the Patent and Copyright Paradigms*, 94 COLUM. L. REV. 2432, 2482 (1994); J.H. Reichman, *The Know-How Gap in the TRIPS Agreement: Why Software Fared Badly, and What Are the Solutions*, 17 HASTINGS COMM. & ENT. L.J. 763, 776 (1995); Pamela Samuelson, *A Case Study on Computer Programs*, in GLOBAL DIMENSIONS OF INTELLECTUAL PROPERTY RIGHTS IN SCIENCE AND TECHNOLOGY 284, 308 (Mitchel B. Wallerstein, Mary Ellen Mogee, & Roberta A. Schoen eds. 1993).

10. *But cf.* Lipton, *supra* note 8, at 248 (arguing that as "programming methodology and digital copyright law . . . develop, copyright appears to be a less and less comfortable fit for software code").

and after recommendations from the National Commission on New Technological Uses of Copyrighted Works (CONTU),[11] the U.S. Congress amended the Copyright Act in 1980 to explicitly include computer programs as a form of protectable subject matter.[12] Other countries followed. In 1985, for example, the United Kingdom, France, West Germany, and Japan included software in their copyright statutes.[13] Also, in 1991, the European Union (then the European Communities) adopted a directive on copyright protection for software.[14] By 1991 "at least 54 countries recognized copyright protection in computer programs."[15]

Although the history of copyright protection for software predates the TRIPS Agreement, there is good reason to focus on the period since the early 1990s when reviewing the mutual effects between software and copyright law. Since the early 1990s, software development has undergone some major changes that have profoundly affected the manner in which software has been created, disseminated, and used.[16] Some of the changes that occurred in the 1990s that are relevant to this article stemmed from a wide adoption of object-oriented programming and the rapidly expanding availability of the Internet, which made software development follow a course that differed from earlier practices in ways that are important for considering the mutual effects between copyright protection and software.[17] These changes occurred after copyright protection for software had become the international norm, and national copyright laws have had to find ways to navigate and adjust to the landscape that resulted from the changes.

This article begins with a brief explanation of two major technological changes—object-oriented programming and the Internet[18]–and continues by reviewing the adjustments that

---

11. National Commission on New Technological Uses of Copyrighted Works (CONTU), *Final Report on the National Commission on New Technological Uses of Copyrighted Works*, 3 Computer L.J. 53, 53 (1981).

12. Gov. Patent Policy Act of 1980, Pub. L. No. 96-517, 94 Stat. 3015. For more information on CONTU, *see generally* Díaz, *supra* note 4.

13. Copyright (Computer Software) Amendment Act 1985, HC Deb. Vol 89, cl. 7.44 ch. 41 (Eng.); Loi 85–660 du 3 juillet 1985 relative aux droits d'auteur et aux droits des artistes-interpretes, des producteurs de phonogrammes et de videogrammes et des entreprises de communication audiovisuelle [Law of July 3, 1985 relating to copyright and the rights of performers, producers of phonograms and broadcasting, and audiovisual communication companies], art. 1, JOURNAL OFFICIEL DE LA REPUBLIQUE FANCAISE [J.O.] [OFFICIAL GAZETTE OF FRANCE], Jul. 4, 1985, P.7495; Gesetz zur Änderung von Vorschriften auf dem Gebiet des Urheberrechts [Law Amending Regulations in the Field of Copyright Law], June 24, 1985, BGBl. I at 1137; Chosakukenhō [Copyright Act] Law No. 62 of 1985 (Japan). Apparently, the first time a court in the United Kingdom recognized a computer program as a literary work that was protectable by copyright was in 1982. Gates v. Swift, [1982] 13 R.P.C. 339 (Chancery Division). Australia preceded these countries with its 1984 law. Copyright Amendment Act 1984 (Cth).

14. Council Directive 91/250 of May 1991 on the Legal Protection of Computer Programs, 1991 O.J. (L 122) 1, 42, 42–46. The 1991 Directive was eventually replaced by Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the Legal Protection of Computer Programs. 2009 O.J. (L 111) 1, 16–22. A 2000 report on the implementation of the Directive noted that "[t]he Directive has been used as a model in a significant number of Central and Eastern European States as well as in Hong Kong, the Philippines and Australia." *Report from the Commission to the Council, the European Parliament and the Economic and Social Committee on the implementation and effects of Directive 91/250/EEC on the legal protection of computer programs*, at 16, COM (2000) 199 final (Apr. 10, 2000).

15. UNCTAD–ICTSD, RESOURCE BOOK ON TRIPS AND DEVELOPMENT 153 (Cambridge University Press 2005), *See generally* Report, Group of Experts on the copyright aspects of the protection of computer software, UNESCO/WIPO/GE/CCS/3, March 8, 1985, Annex B, *available at* https://www.wipo.int/mdocsarchives/UNESCO_WIPO_GE_CCS_1985/UNESCO_WIPO_GE_CCS_3_E.pdf; Nimmer & Krauthaus, *supra* note 9, at 745.

16. GERARD O'REGAN, A BRIEF HISTORY OF COMPUTING 63 (2012) ("The 1980s and 1990s were a time of fundamental change in the computing field.").

17. For a discussion of some other developments *see* Samuelson, *supra* note 8, at 1775–81.

18. *See, e.g.*, MARTIN CAMPBELL-KELLY, FROM AIRLINE RESERVATIONS TO SONIC THE HEDGEHOG: A

copyright law and software experienced in response to changes in the other. It is, of course, difficult to isolate precise causes of effects, and not all of the adjustments described in this article would have happened but for the effect of software on copyright law or copyright law on software.[19] The article aims to pinpoint the effects in copyright law and software where the effects in one arguably played a major role in the changes in the other.

It should be mentioned at the outset that this article focuses on the effects of software as a type of protectable subject matter; it leaves aside the separate topic of how software as a product or service has impacted the implementation and practice of copyright law—for example, how software has supported collective licensing of copyright, or how blockchain technology might augment the management and enforcement of copyright in the digital environment.[20] Similarly, this article does not address copyright issues that arise in connection with computer-generated works created by artificial intelligence—another possible "software and copyright" topic.[21] These topics, while fascinating and important, are beyond the scope of this article.

## I. SOFTWARE SINCE THE EARLY 1990S

The history of software development is a history of important changes: IBM's adoption of the unbundling decision, the emergence of "software product," the rise of the relational database, the introduction of personal computers and mobile technologies, and the release of the Google Maps application programming interface, to name a few.[22] All of the changes influenced software development, but some changes were more significant than others with respect to the relationship between software and copyright law. The following two sections discuss two such significant changes that occurred in the early 1990s.

### *A. Object-Oriented Programming*

One change that software development experienced in the 1990s was the wide adoption of object-oriented programming ("OOP")— a style of programming that had its roots in the mid-1960s but became popular and widely used in the 1990s:[23] OOP differs from the style of programming that prevailed before the 1990s: OOP is "radically different"[24] and its adoption represented "a paradigm shift in programming."[25]

---

HISTORY OF THE SOFTWARE INDUSTRY 99–103, 109–14, and 169 (I. Bernard Cohen & William Aspray eds., 2003) (for examples of other, earlier historic milestones of the software industry).

19. *See generally* Massimiliano Di Penta, Daniel M. German, Yann-Gaël Guéhéneuc & Giuliano Antoniol, An Exploratory Study of the Evolution of Software Licensing, *in* ACM/IEEE 32ND INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (May 2, 2010) (for an example of research relating to the effects of licensing on software).

20. *See generally*, Birgit Clark, *Blockchain and IP Law: A Match Made in Crypto Heaven?*, WIPO MAGAZINE, (Feb. 2018) (on the uses of blockchain to manage and enforce copyright).

21. *See, e.g.*, Miller, *supra* note 4, at 1042–72 (for an early discussion of copyright in computer-generated works).

22. *See generally* CAMPBELL-KELLY, *supra* note 18.

23. *See* JAMES M. GILLIES & ROBERT CAILLIAU, HOW THE WEB WAS BORN: THE STORY OF THE WORLD WIDE WEB 186 (2007) (noting that "[t]he idea took a long time to catch on . . . .").

24. *Id.*

25. O'REGAN, *supra* note 16, at 132.

Before the rise in the popularity of OOP, the programming techniques that were primarily in use were based on procedural programming,[26] with a computer program consisting of a list of individual instructions that a computer would perform.[27] OOP, on the other hand, operates with "objects" (created with programming code) that are "capable of sending and receiving messages and processing data," and "each object [acts] as an independent entity or actor with a distinct role or responsibility."[28] To print a page, procedural programming requires that a program list all the instructions, step by step, necessary for printing; OOP requires only a reference to the print "object" in the programming language library.

An everyday non-computer analogy might be helpful in explaining the difference between OOP and procedural programming: taking out the trash in OOP (i.e. reclaiming unused memory) is just one command: "take out the trash." A person taking out the trash in real life knows–having taken out trash before–and an OOP knows–because the object includes this information–where the trash can is, how to access the trash can, how to tie the trash bag, how to carry out the trash, where the trash bins are, how to open them, and how to deposit the trash bag in the trash bins. In a procedural programming-like real world, the person doing the task has never taken out trash before; each iteration of taking out the trash requires complete instructions to the person, step by step.

Two significant advantages of OOP are its reusability and portability: "[T]he objective is to allow a program to be written once and executed anywhere."[29] The taking out the trash example can also illustrate this OOP characteristic: In an OOP-like real world, a person is able, once told to take out the trash and with no additional instructions, to take out trash not only at home, but also at the grandparents' home, at the office, at a hotel, or anywhere else. In the real world this is because the person knows the trash removal principle and can figure out small nuances in different places; in OOP, an object can complete a task because the object operates in every environment.[30] In a procedural programming-like real world, step-by-step instructions have to be adjusted for the different places that a task might be performed.

OOP began to dominate the programming world in the late 1980s.[31] Its popularity was propelled by the development of the C++ programming language[32] in 1983 and by the release of the programming language Java in 1996, which, as two (not completely unbiased) authors recalled, "generated an incredible amount of excitement."[33] Java was developed in the early 1990s at Sun Microsystems[34] and grew to be extremely popular; according to one ranking of programming languages, Java moved from being the 13th most popular programming language in 1999 to being the most popular programming language in 2004, 2009, and 2019, only dropping to #2 in 2014.[35] Although Java is slowly yielding the top position to other

26. CAY S. HORSTMANN & GARY CORNEL, CORE JAVA, VOLUME I: FUNDAMENTALS 106 (2008).

27. O'REGAN, *supra* note 16, at 132.

28. *Id.*

29. *Id.* at 134.

30. GILLIES & CAILLIAU, *supra* note 23, at 186 (demonstrating that in OOP, "once an object has been defined it is not just confined to a single program: it can be reused").

31. O'REGAN, *supra* note 16, at 133.

32. *Id.* at 133 (demonstrating that C++ is "an object-oriented extension of the C programming language").

33. Horstmann & Cornel, *supra* note 26, at 2.

34. O'REGAN, *supra* note 16, at 134.

35. *Very Long Term History*, TIOBE, https://www.tiobe.com/tiobe-index/ (last visited July 25, 2019).

languages, such as JavaScript and Python, it continues to score high among programming languages.[36]

Java's popularity might seem surprising given that Sun Microsystems unveiled Java as a proprietary system. However, the Java Community Process that was created to control Java proved to be very developer-friendly, and Sun Microsystems "ma[de] most of its Java implementations available without charge."[37] Additionally, as the authors of Sun Microsystem's *Core Java* have explained, Java has been attractive because "Java is a whole *platform*, with a huge library, containing lots of reusable code, and an execution environment that provides services such as security, portability across operating systems, and automatic garbage collection."[38]

Currently, OOP–including programming in Java, Python,[39] and other languages–is "the dominant programming paradigm,"[40] and the *Core Java* authors have suggested that "it is inconceivable that a modern programming language would not use [OOP]."[41] Importantly for the purposes of this article, OOP advances collaboration in code writing because it allows multiple developers to work on a program and write different overlaying programs.[42]

### *B. The Internet*

Another factor that has significantly influenced the course of software development has been the Internet, whose early, major effects can also be placed in the 1990s. Although the Internet had its early visionaries[43] and a predecessor—the ARPANET[44]—it was only the launch of the world-wide web ("WWW") in 1990, coincidently in the same year that the U.S. government decommissioned the ARPANET,[45] "that has transformed the Internet from mainly academic use to where it is now."[46]

While the ARPANET was a U.S. creation, the WWW was developed in Europe by Tim Berners-Lee at the Conseil Européen pour la Recherche Nucléaire (CERN)—the European

---

36. *See* Stephen O'Grady, *The RedMonk Programming Language Rankings: June 2019*, REDMONK (July 18, 2019), https://redmonk.com/sogrady/2019/07/18/language-rankings-6-19/ (stating that according to a RedMonk ranking from June 2019, Java placed second, following JavaScript); *see also* Stephen Cass, *The 2018 Top Programming Languages*, (July 31, 2018), https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages (according to IEEE Spectrum's ranking, in 2018, Java ranked #3 after Python and C++.); *see also* MARJEE CHMIEL, PRINCIPLES OF PROGRAMMING AND CODING 165 (2018) (stating that JavaScript and Java are not the same language. "While JavaScript takes some of its naming conventions from Java, the languages have little else to do with one another. Unlike Java, which is intended for professional computer programmers, JavaScript was aimed at web designers and other non-programmers").

37. O'REGAN, *supra* note 16, at 135.

38. Horstmann & Cornel, *supra* note 26, at 2.

39. *See Object-Oriented Programming (OOP) in Python 3*, REAL PYTHON, https://realpython.com/python3-object-oriented-programming/ (last visited Aug. 6, 2019) (stating that since Python is a multi-paradigm language, it enables a programmer to choose whether he will use OOP or not).

40. Horstmann & Cornel, *supra* note 26, at 2.

41. *Id.* at 4.

42. John Shaeffer, *Software As Text*, 33 SANTA CLARA HIGH TECH. L.J. 324, 355–56 (2017).

43. In the 1940s, for example, Vannevar Bush, described his vision of an electronically-linked information management system. O'REGAN, *supra* note 16, at 102.

44. *See* O'REGAN, *supra* note 16, at 102–06 (describing the ARPANET as [insert text here]).

45. GILLIES & CAILLIAU, *supra* note 23, at 319.

46. O'REGAN, *supra* note 16, at 67.

Particle Physics Laboratory. Once CERN opened its connections to the outside world, it immediately "bec[ame] the biggest Internet site in Europe in terms of traffic" at that time.[47] In 1991, Berners-Lee published a summary paper on the WWW and in 1993, CERN announced that it was making the WWW software freely available.[48] Fueled by the WWW, the Internet spread rapidly, from about 10,000 WWW servers in the world at the end of 1994 to estimated almost 10 million WWW servers by the end of 1999.[49]

The Internet has provided a widely-available platform for the remote creation, dissemination, and use of software. Protocols to do all these functions were available long before the WWW, and the WWW is not indispensable for software sharing.[50] Indeed, the possibility of transmitting software "through geographically vast networks" was one of the issues in which CONTU was interested in the late 1970s.[51] Nevertheless, the WWW enabled the Internet to expand, become widely accessible, and become a vehicle for many uses. The quickly growing popularity of the Internet also propelled the development of a vast infrastructure which, in turn, facilitated dealings with and applications for software.[52]

The development of the infrastructure has enabled the connection of a large number and a wide variety of facilities and devices, thus advancing the emergence of the "cloud" phenomenon[53] and the "Internet of things."[54] In this deeply interconnected environment, software has acquired a role that is more important now than it ever was before. Software no longer operates only mainframe computers, space ships, large airplanes, and nuclear energy facilities—it runs on a much greater variety of many more interconnected facilities and devices: surgical instruments, autonomous vehicles, appliances, and many other devices. Software also plays an increasingly significant role as it is deployed, together with artificial intelligence, to make critical decisions concerning individuals and society at large.[55] Questions of the ownership of software, its utilization, cost, reliability, and safety directly affect the day-to-day functioning of society. The fact that a great deal of software is embedded in various devices might seem to speak against the importance of copyright protection as an incentive for the creation of software.[56] However, even when software is embedded in

47. GILLIES & CAILLIAU, *supra* note 23, at 87.

48. *Id.* at 261 ("In 1993, pushed on by Tim Berners-Lee and Robert Cailliau, CERN issued a statement putting the Web software in the public domain."); T.J. Berners-Lee, R. Cailliau & J.-F. Groff, *The World-Wide Web,* 25 Computer Networks and ISDN Systems 454, 458 (1992) ("Software provided by the various contributors to the W3 project include[d] browsers, servers and gateways.").

49. GILLIES & CAILLIAU, *supra* note 23, at 306.

50. For example, the FTP protocol was used to share the first Linux files in 1991. *See, e.g.,* Peter S. Menell, *Rise of the API Copyright Dead?: An Updated Epitaph for Copyright Protection of Network and Functional Features of Computer Software*, 31 HARV. J. OF L, & TECH. 305, 320 (2018) (describing the growth and success of Linux).

51. Díaz, *supra* note 4, at 762.

52. *See* Menell, *supra* note 50, at 348 (describing how the world wide web fueled Java development).

53. *See* Patt Morrison, *Evgeny Morozov, Internet Cassandra*, L.A. TIMES (June 19, 2013), https://www.latimes.com/opinion/la-xpm-2013-jun-19-la-oe-morrison-evgeny-morozov-internet-20130619-story.html ("[c]loud computing is a great euphemism for centralization of computer services under one server.").

54. *See* Joseph Gratz & Mark A. Lemley, *Platforms and Interoperability in Oracle v. Google*, 31 HARV. J. OF L, & TECH 603, 612 (2018) (describing interoperability as crucial to the development of the Internet of Things, which connects a wide variety of devices).

55. *E.g.*, Sonia K. Katyal, *The Paradox of Source Code Secrecy*, 104 Cornell L. Rev. 1184, 1184–85 (2019) (discussing a bill that would require agencies to use algorithms to publish source code); Robert Brauneis & Ellen P. Goodman, *Algorithmic Transparency for the Smart City*, 20 YALE J. L. & TECH 103, 114 (2018) ("Use of big data and predictive algorithms is a form of governance—that is, a way for authorities to manage individual behavior and allocate resources.").

56. Samuelson, *supra* note 8, at 1776–77.

devices, copyright plays an important role in promoting the creation of software—if not as an incentive to create, than as a means of protecting an investment in other stages of the lifecycle of the software.

## II. ADJUSTMENTS MADE TO COPYRIGHT LAW AND SOFTWARE SINCE THE EARLY 1990S IN LIGHT OF THEIR MUTUAL EFFECTS

Software has tested the flexibility of copyright law as courts have applied copyright law doctrines to this new type of protectable subject matter. In many instances, it was judicial interpretation that molded copyright protection onto software.[57] In other instances, legislators adjusted copyright law to make it better suited for software. Some legal questions concerning software were resolved before the 1990s. The following sections focus on adjustments to the law that occurred later, including an occasional glimpse into the rear-view mirror with views of pre-1990 developments. The sections also point to changes in software that reflect the application of copyright law to software.

### *A. Transactions in Software*

OOP and the Internet have intensified the collaborative nature of software development. Portability, a key characteristic of OOP, encourages collaboration in software development and layering—software development in which pre-existing objects are reused to write new software. The growth of the Internet has supported collaboration in software development, enabling software development to become transnational, with software development teams working across many countries.[58]

Developers may collaborate on only one piece of software, or they may develop separate but interrelated components of software, work on overlapping components, or create new layers based on pre-existing code and objects.[59] Developers may be working in different jurisdictions with different legal regimes, which may affect the issues of initial ownership of copyright and the alienability of copyright.[60] Neither of these issues has been harmonized in international treaties, and differences in their treatment persist among national laws.

Moral rights can become a significant hurdle in completing transactions in copyright-protected works. In many countries, moral rights can be neither alienated nor waived, and even when economic rights are transferred or licensed to another party, an author might eventually exercise his moral rights to the detriment of the party's exercise of the party's economic rights.[61] An author's exercise of the right of integrity or the right of withdrawal can adversely affect the copyright owner's enjoyment of economic rights to the work.

---

57. *See* Pamela Samuelson, *Evolving Conceptions of Copyright Subject Matter*, 78 U. OF PITT. L. REV. 17, 69 (2016) ("It is fair to say that courts have had to develop a sui generis form of copyright protection for programs through case-by-case adjudications.").

58. The Geography of Innovation: Local Hotspots, Global Networks, World Intellectual Property Report 2019, WIPO, 2019, p. 5 (on the widespread collaboration that is "increasingly cross-border in nature").

59. *See* Nimmer & Krauthaus, *supra* note 9, at 745 (explaining, for example, the issues that arise when a technology has different "authors" from various countries).

60. *See id.* at 737 ("The subissue that distinguishes purely national from international considerations involves determining not only what protection should be given to programs in, for example, Brazil, but how this interacts with protection in, for example, China, Japan, and Germany.").

61. *See* U.S. COPYRIGHT OFFICE, AUTHORS, ATTRIBUTION, AND INTEGRITY: EXAMINING MORAL RIGHTS IN

In the United States, moral rights under copyright law were designed in 1990 so as not to endanger transactions in software. The U.S. Copyright Act does not provide moral rights in computer programs. Although it does provide moral rights in "works of visual art,"[62] which might include some elements of software,[63] it excludes from the enjoyment of moral rights all works made for hire[64] and also allows moral rights to be waived by the author.[65] The software industry can therefore largely avoid problems with moral rights in the United States.[66]

Recognizing that software development would face problems with moral rights, some countries chose to limit their existing moral rights when software is the protectable subject matter. For some countries, the decision to limit moral rights was remarkable because it was a concession from their tenacious position on protecting an important aspect of authors' rights. Even France—which is traditionally a strong proponent and defender of authors' moral rights— in its 1985 amendment to its copyright act eliminated, for computer programs, "the moral right to oppose unauthorized modification of the work and repent or cancel the license in the event of such modification."[67]

In some countries, rules of initial ownership and alienability of economic rights may also complicate transnational dealings in copyright-protected works. For instance, civil law countries traditionally afford author's rights initially to the author—the natural person who created the work—and some of these countries make economic rights non-assignable, allowing the rights to be licensed only. This approach is in stark contrast to the position of common-law countries, in which the work-made-for-hire doctrine vests copyright in the author's employer, and copyright is assignable.[68]

Recognizing the burden that differences in national rules create for software development,[69] the 1991 EU Directive required that all EU countries adopt a work-made-for-hire-type provision for computer programs.[70] The 1994 French implementing legislation–for example–in line with the 1991 EU Directive, introduced the rule that "the economic rights in

---

THE UNITED STATES, 13 (2019) (enumerating how moral rights are recognized around the world).

62. *See* 17 U.S.C. § 101 (1990) (the definition of a "work of visual art"); 17 U.S.C. § 106A(a) (1990) (delineating rights of attribution and integrity for the author of a work of visual art). The moral rights provision was added to the U.S. Copyright Act in 1990 to bring U.S. copyright law in compliance with the Berne Convention. *See also* U.S. Copyright Act, Pub. L. No. 101-650, § 603(b), 104 Stat. 5128 (1990) (codified as amended at 17 U.S.C. § 106A).

63. *See* 17 U.S.C. § 101 (noting that a "work of visual art does not include" audiovisual works and electronic publications).

64. *Id.*

65. *Id.* § 106A(e).

66. In the United States, doctrines from areas of law other than copyright law supplement the limited moral rights afforded by the U.S. Copyright Act. The doctrines from other areas of law might be available for software, but they are unlikely to cause the type of transactional difficulties that moral rights under copyright law may cause in other countries.

67. Nimmer & Krauthaus, *supra* note 9, at 752. *See also* Loi 85-660 du 3 juillet 1985 relative aux droits d'auteur et aux droits des artistes-interpretes, des producteurs de phonogrammes et de videogrammes et des entreprises de communication audiovisuelle [Law 85-660 of July 3, 1985 relating to copyright and the rights of performers, producers of phonograms and videograms and audiovisual communication companies], JOURNAL OFFICIEL DE LA REPUBLIGQUE FRANÇAISE [J.O.] [OFFICIAL GAZETTE OF FRANCE], July 4, 1985, p. 7495. https://www.legifrance.gouv.fr/jo_pdf.do?id=JORFTEXT000000693451&pageCourante=07495.

68. *E.g.*, 17 U.S.C. § 101 (the definition of a "work made for hire"), 201(b), 204(a); Copyright Act, R.S.C. 1985, c. C-42 (Can.), § 13(3)–(4) (Can.); Copyright, Designs and Patents Act 1988 § 11(2), c. 48 (Eng.).

69. *See* Council Directive 2009/24/EC, para. 4–5, 2009 O.J. (L 111) 16 (EC).

70. *Id.* at art. 2(3).

the software and its documentation created by one or more employees in the execution of their duties or following the instructions given by their employer shall be the property of the employer and he exclusively shall be entitled to exercise them."[71] Since 1993, the German copyright statute has provided employers with an exclusive right to exercise economic rights in such computer programs.[72]

These adjustments to national copyright laws were very important because in many cases no contract among parties could resolve problems associated with the alienability of rights. National laws may consider the rules that limit the alienability of copyright to be internationally mandatory, meaning that parties cannot avoid the application of the rules by selecting a different country's law to apply to their contract.[73] Regardless of the particular national law that parties may select in a choice-of-law provision, a court will always apply the national rule if the rule is internationally mandatory.[74]

Alienability issues therefore required statutory changes; although changes in national statutes improved the situation, the changes did not solve all of the problems, and transnational projects involving copyright-protected works therefore continue to need careful structuring.

### *B. Licensing and Copyright Exhaustion*

Licensing practices have had effects on software development and use, and software development and use have prompted changes in licensing practices. For example, the license under which Sun Microsystems released the Java Platform, Standard Edition Development Kit, had previously permitted licensees to make only "a single copy of the Licensed Software for archival purposes."[75] Consequently, developers could not include Java in Linux distributions, and "for many years, end-users had to manually download and install Java."[76] After working with the Free Software Foundation, Sun Microsystems in 2006 released Java 5.0 under GPL 2.0 with a special exception[77] which thereafter permitted developers to include Java in Linux distributions.[78]

The interaction between software and copyright law also influenced the development of the first sale doctrine (copyright exhaustion) and its applicability to software. The doctrine—a traditional limitation on copyright—was developed before the rise of digital technologies and its application to digital copies has presented significant challenges. The ability of a user to

---

71. CODE DE LA PROPRIETE INTELLECTUELLE [Intellectual Property Code], art. L113-9 (Fr.).

72. Gesetz über Urheberrecht und verwandte Schutzrechte [Urheberrechtsgesetz] [UrhG] [Copyright Act], Sept. 9, 1965, BGBl I at 1273 (Ger.).

73. *See* Joost Blom, *Public Policy in Private International Law and Its Evolution in Time*, 50 NETH. INT'L L. REV. 373, 382 (2003) (stating that "Laws of immediate application are, in effect, laws that include their own, unilateral choice of law rules").

74. A national court will always apply an internationally-mandatory rule, at least if the rule is part of the law of the court's own jurisdiction. A court might not apply the rule if it is a foreign law rule.

75. *Java 3D 1.2.1_03 Binary Code License Agreement, Sec. 2.3*, ORACLE, https://download.oracle.com/otndocs/jcp/7627-3d-1.2.1_03-class-oth-JSpec/7627-3d-1.2.1_03-class-oth-JSpec-license.html (last visited Aug. 15, 2019).

76. Di Penta et al., *supra* note 19, at § 2.2.

77. *GNU General Public License, Version 2, with the Classpath Exception*, OPEN JDK, https://openjdk.java.net/legal/gplv2+ce.html, (last visited Aug. 15, 2019).

78. *See* Di Penta et al., *supra* note 19, at § 2.2 ("Java programs could be released under any license as long as they satisfy the conditions stated in the CLASSPATH exception").

continue to utilize a work after the user has transferred either the original copy of the work or its unauthorized reproduction continues to worry copyright owners, regardless of the medium in which the copies of their works have been distributed.[79] But digital environment technologies have made the possibility of high-quality low-cost reproduction and distribution[80] a reality— to the great detriment of copyright owners' economic interests. Owners of copyright in works that are distributed in digital copies have therefore strived to limit the applicability of the first sale doctrine. For example, at the international level, copyright owners successfully lobbied for the introduction of rental rights, which limit the application of the first sale doctrine.[81]

Copyright owners have also tried to limit the applicability of the first sale doctrine by imposing restrictions on the alienation of copies, but courts have not been sympathetic to the use of these restrictions for non-digital copies.[82] Once digital copies were at issue, however, courts seemed more sympathetic to the attempts to limit the first sale doctrine and recognized licensing as a legitimate manner of avoiding a sale and excluding the first sale doctrine.

Some companies licensed their software products even before computer programs were explicitly included in copyright statutes as a category of protectable subject matter;[83] other companies began to license, rather than sell, their software later.[84] In the United States, the U.S. Court of Appeals for the Ninth Circuit held that customers of a software company that licensed its software were not owners of the software but only licensees.[85] By the time the court confirmed its holding in a later decision in 2006,[86] licensing was a well-established practice in the software industry.[87]

The seminal U.S. decision concerning the distinction between a sale and a license of a computer program was the 2010 decision by Ninth Circuit in *Vernor v. Autodesk, Inc.*[88] The court formulated three factors that should be considered when analyzing whether an agreement is a license or a sale: (1) "whether the copyright owner specifies that a user is granted a license," (2) "whether the copyright owner significantly restricts the user's ability to transfer the software," and (3) "whether the copyright owner imposes notable use restrictions."[89]

---

79. *See* 18 AM. JUR. 2d *Copyright and Literary Property* § 110 (2020) (discussing the "first sale" doctrine).

80. *See* Peter S. Menell, *Envisioning Copyright Law's Digital Future*, 46 N.Y.L. SCH. L. REV. 63, 118–29 (2002) (illustrating the "implications of digital content for the principal entertainment industries").

81. TRIPS Agreement, *supra* note 1, at art. 11; WIPO Copyright Treaty, *supra* note 6, at art. 7.

82. *See* Bobbs-Merrill Co. v. Straus, 210 U.S. 339, 350 (1908) (holding that "while protecting the owner of the copyright in his right to multiply and sell his production, [the copyright statutes] do not create the right to impose, by notice...a limitation at which the book shall be sold at retail by future purchasers, with whom there is no privity of contract").

83. *See* Watts S. Humphrey, *Software Unbundling: A Personal Perspective*, 24 IEEE ANNALS HIST. OF COMPUTING 59, 60 (2002) (discussing IBM's attempt to couple copyright with a license and "count[ing] on the license to provide the real protection").

84. *See* Triad Systems Corp. v. Southeastern Exp. Co., 64 F.3d 1330, 1333 (9th Cir. 1995) ("In 1986 . . . Triad began licensing rather than selling its software.").

85. MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 n.5 (9th Cir. 1993).

86. *See* Wall Data Inc. v. Los Angeles Cty. Sheriff's Dep't, 447 F.3d 769, 785 (9th Cir. 2006) (holding that "[g]enerally, if the copyright owner makes it clear that she or he is granting only a license to the copy of software and imposes significant restrictions on the purchaser's ability to redistribute or transfer that copy, the purchaser is considered a licensee, not an owner, of the software").

87. *See* Robert W. Gomulkiewicz, *Is the License Still the Product?*, 60 ARIZ. L. REV. 425, 432 (2018) (stating that "licensing has emerged as the dominant transaction model for software.").

88. Vernor v. Autodesk, Inc., 621 F.3d 1102, 1102 (9th Cir. 2010), *cert. denied*, 565 U.S. 820, 820 (2011).

89. *Id.* at 1110–11.

In the European Union, German courts had disagreed on the applicability of copyright exhaustion in cases involving software,[90] so the Court of Justice of the European Union ("CJEU") clarified the line between a sale and a license in 2012 in *Usedsoft GmbH v. Oracle International Corp.*[91] The CJEU tailored its ruling to the facts of the particular case and made a narrow ruling for a sale rather than a license when a copyright holder enables the downloading of software from the Internet, charges "a fee intended to enable him to obtain a remuneration corresponding to the economic value of the copy of the work," and grants the right to use the copy of the software for an unlimited period.[92]

Both *Vernor* and *UsedSoft* provided guidance to the software industry for structuring their business models to avoid the application of the first sale doctrine, and the court decisions had already reflected evolving software licensing practices. Some licensing features have been considered from the beginning—such as periodic payments instead of one-time up-front payments, indefinite duration of licenses, and technological measures to protect against unlicensed uses.[93] But the use of these features has been strengthened and encouraged by the copyright law's endorsement of and support for the features. This endorsement and support may be through drawing a line between a sale and a license, discussed here,[94] or through the introduction of legal safeguards for technological protection of copyrighted works.[95] It is apparent that court decisions favor the subscription-based, cloud-based, and software-as-a-service models, such as the model used by Google for its G Suite and Microsoft for its Office 365. Whether this is a positive outcome for consumers or society at large is debatable.

### *C. Open Source and Public Licenses*

Two significant imports into copyright law that came from the software industry are the open source movement and public licenses.[96] The origins of the open source movement have more to do with trade secrets than with copyright; the open source movement developed in response to the software companies' practice of protecting their computer programs source code as a trade secret.[97] In opposition to the spread of the practice, some developers began to

90. Oracle v. Usedsoft, Landgericht Munich I, 7 O 23237/05, January 19, 2006; Oberlandesgericht Munich, 6 U 1818/06, August 3, 2006; Microsoft v. Usedsoft, Landericht Munich, 315 O 343/06, June 29, 2006; Microsoft v. Usedsoft, Hanseatisches Oberlandesgericht, 5 U 140/06, February 7, 2007.

91. Case C-128/11, Usedsoft GmbH v. Oracle International Corp., 2012 ECLI:EU 407.

92. *Id.* para. 72.

93. Watts S. Humphrey, *Software Unbundling: A Personal Perspective*, IEEE ANNALS OF THE HISTORY OF COMPUTING 60, 60 (2002).

94. *See* Gomulkiewicz, *supra* note 91, at 429 (describing the possible effects of the U.S. Supreme Court's decision in *Impression Products*, and concluding that "despite the Court's emphatic rejection of patent remedies for conditional sales contracts and the link between the patent and copyright exhaustion doctrines, the Court has left the door ajar for using copyright remedies to enforce [end-user licensing agreements]").

95. WIPO Copyright Treaty, *supra* note 6, at art. 11; *See also* 17 U.S.C. § 1201 (1999) (outlining exemptions to anti-circumvention laws).

96. *See* Debra Brubaker Burns, *Titans and Trolls Enter the Open-Source Arena*, 5 HASTINGS SCIENCE AND TECHNOLOGY L.J. 33, 45–46, 57–82 (2013) (describing the effects of open source software on patent law and vice versa).

97. *See* Díaz, *supra* note 4, at 765 (describing how trade secrets have been an important means of protecting software in the United States. Before the copyright/trade secrets overlap was clarified, some critics had warned that copyright protection would be incompatible with trade secrets protection; for example, in the late 1970s a Bell Telephone Laboratories patent attorney warned that the adoption of copyright protection for software would deprive companies of the "main avenue of protection" through secrecy). *But see* UNITED STATES COPYRIGHT

promote "open source" code, meaning accessible and human readable, public, and offered on the basis of a public license.[98]

The beginning of the open source movement dates to 1983, when Richard Stallman started the GNU project to revive "the cooperative spirit that prevailed in the computing community in earlier days."[99] His efforts also resulted in the creation of the Free Software Foundation in 1985[100] and the drafting of the GNU General Public License ("GNU GPL"), which was first released in 1989.[101] A development that contributed to the popularity of the movement and the GNU GPL was the release of the Linux kernel under the license in 1992.[102] By 1998, the term "open source software" seemed to have prevailed over the original term "free software," which fell out of favor with some because use of the word "free" created the misconception that the software was free of charge, which was not necessarily the case. The abbreviations "FLOSS" and "FOSS" also emerged as a way to capture both terms.[103]

The misconception about the meaning of the terms "free" and "open" shaped the open source movement's impact on copyright law. Notwithstanding Stallman's attempts to make it understood that the "free" in his "free software" referred to "freedom,"[104] many continued to believe that "free," and later, "open," meant "not requiring any payment."[105] Further, it was likely this misconception that propagated the term "open" in other contexts, such as the term "open access" when used to describe the free-of-charge availability of academic articles[106] and "open data" when used for free-of-charge available data.[107] But "free software" was not meant to be free of charge; in fact, Stallman referred to "the freedom to *sell* copies [as]

---

OFFICE, CIRCULAR 61, COPYRIGHT REGISTRATION OF COMPUTER PROGRAMS, 3–4 (2017), https://www.copyright.gov/circs/circ61.pdf (describing how the Copyright Office adopted special rules for the copyright registration of works protected by trade secrets, eventually allowing applicants to file only a portion of the code without revealing any trade secrets).

98. Richard Stallman, *Overview of the GNU System*, GNU OPERATING SYSTEM (Mar. 12, 2020), http://www.gnu.org/gnu/gnu-history.html (describing an early example of open sound software).

99. *Id.*

100. *Id.*

101. FSFE, *Transcript of Richard Stallman at the 2nd International GPLv3 conference* (Apr. 21, 2006), https://fsfe.org/campaigns/gplv3/fisl-rms-transcript.en.html.

102. *See* Menell, *supra* note 83, at 182 (describing how "the growth and success of Linux has brought the open source movement into the mainstream of the computer software industry").

103. *See, e.g.*, Deliverable D18: Final Report, Int'l Inst. of Infonomics, Univ. of Maastricht & Berlecon Research GmbH, Free/Libre and Open Source Software: Survey and Study (June 2002), https://web.archive.org/web/20120512210644/http://flossproject.org/report/FLOSS_Final0.pdf (stating that free/libre and open source software are used interchangeably); THE MITRE CORP., USE OF FREE AND OPEN-SOURCE SOFTWARE (FOSS) IN THE U.S. DEPARTMENT OF DEFENSE 2 (January 2, 2003), https://dodcio.defense.gov/Portals/0/Documents/FOSS/dodfoss_pdf.pdf (stating software that qualifies as free almost always also qualifies as open source). *See also* Conference Paper, Working Group on Libre Software, Free Software / Open Source: Information Society Opportunites for Europe? (April 2000), http://eu.conecta.it/paper.pdf (demonstrating that in Europe, the term "libre" has been used as an alternative term to prevent the "free," as in "free of charge," misconception).

104. Richard Stallman, *The GNU Project*, GNU OPERATING SYSTEM, http://www.gnu.org/gnu/thegnuproject.html (last visited February 15, 2020) ("The term 'free software' is sometimes misunderstood—it has nothing to do with price. It is about freedom.").

105. Free Software Foundation, *What Is Free Software?*, GNU Operating System (Mar. 20, 2019), https://gnu.org/philosophy/free-sw.en.html.

106. *E.g.*, SPARC OPEN ACCESS, https://sparcopen.org/open-access/ (last visited Aug. 5, 2019).

107. *Cf. About the ODI*, Open Data Inst., https://theodi.org/about-the-odi/ (last visited Aug. 5, 2019) (using the term "open" consistently with the original "open source" meaning); Hyon Kim, *Data.gov at Ten and the Open Government Data Act*, DATA.GOV (May 31, 2019), https://www.data.gov/meta/data-gov-at-ten-and-the-open-government-data-act/. (using the term "open" to mean without payment).

crucial"[108] and explained that "collections of free software sold on CD-ROMs are important for the community, and selling them is an important way to raise funds for free software development."[109]

Terminology confusion also seems to follow the GNU GPL when it is said that the GNU GPL follows a "copyleft" methodology.[110] The term created the impression that the adopters of the methodology were suggesting that copyright law be eliminated.[111] Regardless of the political ideology that the GNU GPL creators and supporters formerly adhered to or later adopted, the functioning of the GNU GPL relies on copyright law. Stallman explained that the GNU GPL "flips [copyright law] over to serve the opposite of its usual purpose: instead of a means for restricting a program, it becomes a means for keeping the program free."[112] Under the license, anyone is free to "to run the program, copy the program, modify the program, and distribute modified versions—but [is] not permi[ted] to add restrictions of their own."[113] The rule means that anyone who builds his software while using a pre-existing open source program must provide his program under the same conditions.

The GNU GPL model was followed by the Creative Commons,[114] which released its first Creative Commons license in 2002. The Creative Commons aims to simplify copyright licensing by providing a license suite—a set of license versions with different restrictions. The versions are named after the restrictions, such as "Attribution," "Attribution–NonCommercial," and "Attribution–NonCommercial–NoDerivs."[115] The individual license versions are identified by these abbreviated titles and by icons, which are designed to make it faster and easier to immediately recognize the terms under which a work is being licensed.

The open source movement experienced what Pamela Samuelson described as "a substantial rise in the use and economic significance of open-source software."[116] The GNU GPL evolved into its current (third) version and was accompanied by two modified versions, which were released in 2007.[117] The Creative Commons license is now, and has been since 2013, available in its fourth version.[118] Since the 1990s, the open source movement has matured, and it is no longer only the mission of individual software developers; large corporations have built their products with open source software as their main competitive

108. Stallman, *supra* note 108 (emphasis added).

109. *Id.*

110. *Id.*

111. *See* ZOHAR EFRONI, ACCESS-RIGHT: THE FUTURE OF DIGITAL COPYRIGHT LAW 430–31 (2011) (providing a criticism of the mixed ideological messages of the Creative Commons project).

112. Stallman, *supra* note 108.

113. *Id.*

114. *See Frequently Asked Question: About CC: What Is Creative Commons and What Do You Do?*, CREATIVE COMMONS, https://creativecommons.org/faq/#what-is-creative-commons-and-what-do-you-do (last updated Jan. 24, 2020) (stating that "Creative Commons is a global nonprofit organization that enables sharing and reuse of creativity and knowledge through free legal tools").

115. *See CC Licenses and Examples*, CREATIVE COMMONS, https://creativecommons.org/share-your-work/licensing-types-examples/ (last visited Aug. 5, 2019).

116. Samuelson, *supra* note 8, at 1777.

117. Matt Lee, *FSF Releases the GNU General Public License, Version 3*, FREE SOFTWARE FOUND. (June 29, 2007, 12:15 PM), https://www.fsf.org/news/gplv3_launched.

118. *General License Information, Which is the Latest Version of the Licenses Offered by Creative Commons?*, CREATIVE COMMONS, https://creativecommons.org/faq/#which-is-the-latest-version-of-the-licenses-offered-by-creative-commons (last visited August 15, 2019).

strategy (e.g., Red Hat, Inc.)[119] or have used open source software as one of their strategies (e.g., IBM).[120]

The maturing of the open source and public licensing movements revealed some problems. Open source licensing requirements—the disclosure of source code and the requirement that one offer further licenses under the same terms—mean that software that is created and based on pre-existing open source code may never become a trade secret. Converting open source software into a trade secret could lead to future challenges, including challenges in litigation. The open source origins of a computer program may pose problems in industries where disclosure of the source code is undesirable and even problematic, such as in the gambling industry. Open source origins may also deter potential acquirers of the software—or of entire businesses built on open source software—if the acquirers insist that software be protected as a trade secret. It is also possible that keeping open source-based software protected as a trade secret may go undetected (since this is the point of a trade secret), which would cause public licenses to be unenforced.[121]

The status of public licenses has been questioned; it was previously unclear how they should be classified as legal instruments, whether and how they would be enforceable, and what remedies—if any—should be available when they were violated.[122] The license drafters did not seem overly concerned about how courts might perceive the licenses.[123] The drafters might have thought of the licenses primarily as mission statements and pledges comporting to the mission of the open source movement.[124] But when disputes involving public licenses began to arise, courts questioned the legal status of the licenses. It was apparently in 2004 that a court first ruled that the GNU GPL was a valid legal instrument: a German court ruled in 2004 that the GNU GPL[125] was a valid legal instrument under German law.[126] The U.S. Court of Appeals for the Federal Circuit confirmed the validity of the GNU GPL under U.S.

---

119. *See Open Source: Creating better technology with open source*, RED HAT, https://www.redhat.com/en/about/open-source (last visited Feb. 11, 2020) (highlighting the fact that Red Hat is the largest open source company in the world).

120. *See generally* Pamela Samuelson, *IBM's Pragmatic Embrace of Open Source*, COMMC'NS OF THE ACM, Oct. 2006, at 21–25; *see also* Todd Moore & Chris Ferris, *IBM's approach to open technology*, IBM DEVELOPER (May 16, 2016), https://developer.ibm.com/articles/cl-open-architecture-update/ (stating that IBM has invested close to $1 billion and dedicated hundreds of open source developments, marketing, and evangelism resources over the past five years). In July 2019 Red Hat, Inc. became a subsidiary of IBM. *See also IBM Closes Landmark Acquisition of Red Hat for $34 Billion; Defines Open, Hybrid Cloud Future*, RED HAT (July 9, 2019), https://www.redhat.com/en/about/press-releases/ibm-closes-landmark-acquisition-red-hat-34-billion-defines-open-hybrid-cloud-future (stating that the acquisition between Red Hat and IBM will accelerate innovation and will unlock the true value of hybrid cloud for businesses).

121. On potential ways to avoid the application of the GPL *see generally* Theresa Gue, *Triggering Infection: Distribution and Derivative Works Under the GNU General Public License*, 2012 U. ILL. J. L. TECH. & POL'Y 95, 95–140 (2012).

122. For a discussion of the issues in U.S. courts *see id.* at 101–05 (discussing the case law, contract theory, and remedies); *see also* Burns, *supra* note 100, at 40–45, 50–56 (discussing copyright and contract law, copyright-ownership, Red Hat's settlement, and OSS licenses); *see also* EFRONI, *supra* note 115, at 431–36 (discussing theoretical critiques of copyright law).

123. Gue, *supra* note 125, at 101–05.

124. Burns, *supra* note 100, at 50–56.

125. District Court of Munich I, May 19, 2004, 21 0 6123/04 (Open Source–effectiveness of GPL) (Ger.).

126. Axel Metzger, *German Court Says GPL is Valid*, SLASHDOT (Jul. 23, 2004), https://yro.slashdot.org/story/04/07/23/1558219/german-court-says-gpl-is-valid.

law in 2008 in *Jacobsen v. Katzer*.[127] In some countries, legislative changes helped clarify the legal position of public licenses.[128]

The status of the public licenses must be clarified country by country because the status depends on local law; each legal system examines the licenses through the prism of its own rules. The licenses must therefore comport with the various legal systems, and should be equipped with appropriate provisions to operate within the systems. Public licenses have therefore faced problems with localization—their applicability in and potential adjustments necessary to conform to the requirements of different jurisdictions. Originally, the GNU GPL and the Creative Commons licenses were drafted for the U.S. legal environment and took little or no account of non-U.S. legal systems. However, the collaborative and transnational nature of software development, and particularly when it was considered in the context of open source software, soon exposed the need for the licenses to be enforceable not only in individual foreign countries but also in multiple jurisdictions simultaneously.[129]

Realizing the need for its licenses to be localized, the Creative Commons, with the assistance of its partners in various countries, gradually created localized—or "ported"—versions of its 3.0 licenses, which were adjusted to various legal systems. In version 4.0, the Creative Commons adopted the approach taken by the 2007 GNU GPL and created internationalized, "unported" versions of its licenses.[130] Building on its experience with translating and adapting earlier versions to local conditions in more than 60 jurisdictions, the Creative Commons released version 4.0 in 2013 as "ready-to-use around the world, without porting,"[131] declaring that the version "has been drafted with particular attention to the needs of international enforceability."[132] The licenses now bear a designation "international" and are available in multiple languages.[133]

It is extremely difficult to design universally usable licenses for transnational scenarios, and it is questionable whether the internationalized versions of the public licenses serve such scenarios better than the earlier localized versions. The internationalized versions are drafted

---

127. *See* Jacobsen v. Katzer, 535 F.3d 1375, 1381(2008) ("Copyright holders who engage in open source licensing have the right to control the modification and distribution of copyrighted material."). For a commentary on the decision and subsequent decisions, *see generally* Robert W. Gomulkiewicz, *Clarifications and Complications in Enforcing Open Source Software Licenses*, *in* RESEARCH HANDBOOK ON INTELLECTUAL PROPERTY LICENSING 76, 76–98 (Jacques de Werra ed., Edward Elgar, Cheltenham, 2013).

128. Autorský zákon, §46(5) and (6), introduced by zákon č. 216/2006 Sb., kterým se mění zákon č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) (Czech Republic). For a comparative review of the treatment of open source licenses in national legal systems *see generally* FREE AND OPEN SOURCE SOFTWARE (FOSS) AND OTHER ALTERNATIVE LICENSE MODELS: A COMPARATIVE ANALYSIS (Axel Metzger ed. 2016).

129. *See* Axel Metzger, *Internationalisation of FOSS Contributory Copyright Assignments and Licenses: Jurisdiction-Specific or "Unported"?*, 10 SCRIPTED: A J. OF L., TECH. & SOC'Y 177, 184 (2013) (stating that "typical FOSS projects are driven by international communities of software developers situated in different jurisdictions, especially in the US, Europe and Asia").

130. *See id.* at 203–04 (noting that, when published, the 4.0 process was still under discussion, but "the expectation [was] that there [would] be few, if any, 'ported' licenses").

131. *What's New in 4.0*, CREATIVE COMMONS, https://creativecommons.org/version4/ (last visited Feb. 12, 2020).

132. *Frequently Asked Questions*, CREATIVE COMMONS, https://creativecommons.org/faq/#what-are-the-international-unported-creative-commons-licenses-and-why-does-cc-offer-ported-licenses (last visited Feb. 12, 2020).

133. *See, e.g.*, *Attribution 4.0 International*, CREATIVE COMMONS, https://creativecommons.org/licenses/by/4.0/legalcode (last visited Feb. 12, 2020).

to be consistent with the terminology of international treaties on intellectual property law and presumably the licenses aim to be applicable in transnational contexts.[134] But the internationalization can result in the licenses not being well suited for use in any single jurisdiction, which is a significant problem because it is precisely one particular jurisdiction in which the license will be evaluated, interpreted, and applied. As some commentators have pointed out, version 4.0 does not include provisions on choice of court and choice of law, which are, quite possibly, the most important provisions for a transnational instrument.[135]

Outside of the software context, the adoption of the term "open"—a term that is clouded by misunderstanding regarding its meaning—has fostered another set of problems. The "open" movements that have adopted the term with the meaning "not requiring payment" often overlook or underestimate the question of funding.[136] While funding might not be an issue with government-financed work or projects, where access to funding is a part of "open government" (as in "transparent government") initiatives, funding becomes a significant problem when no pre-existing funds are available to support creative endeavors. Stallman was well aware that "free software" cannot exist without some funding; the Free Software Foundation was created specifically for the purpose of seeking funding for the development of free software.[137] Removing payment for access, as is the case with "open access," means that some funding other than a charge for access must be sought. In the world of academic literature, for example, a lack of revenue led publishers to require payments from authors for their works to be published.[138] Some authors might be able to cover such payments from grants and similar sources, but many authors might not be able to do this. A debate about the appropriate distribution of costs is beyond the scope of this article.

### *D. Scope of Protection and Interoperability*

Among the software-related issues in copyright law that required clarification was the scope of protection of computer programs—defining the elements of a computer program that copyright would protect. TRIPS Article 10 states that copyright protection should be available for "computer programs, whether in source or object code,"[139] but copyright law may cover much more than just code. Computer programs include or generate other works that may qualify for protection in various other categories of copyrightable subject matter, such as

134. Metzger, *supra* note 133, at 200–01.

135. *Id.* at 204–05; Matěj Myška, 2014, *The New Creative Commons 4.0 Licenses*, Conference on Grey Literature and Repositories: proceedings 2014: The Value of Grey Literature in Repositories, Prague: National Library of Technology, https://invenio.nusl.cz/record/175812/files/idr-806_4.pdf.

136. Stallman, *supra* note 108.

137. *Id.*

138. Andrew V. Suarez & Terry McGlynn, *The Fallacy of Open-Access Publication*, THE CHRONICLE OF HIGHER EDUCATION (Nov. 15, 2017), https://www.chronicle.com/article/The-Fallacy-of-Open-Access/241786. It is no coincidence that legal academics in the United States seem much more attracted to open access than their European colleagues, who are paid much less and rely on royalties from their publications to supplement their income.

139. TRIPS Agreement, *supra* note 1.

graphical, pictorial, musical, and audiovisual works.[140] Preparatory works leading to coding involve work that may also be protected by copyright.[141]

A number of cases called for courts to clarify the scope of copyright protection for computer programs and associated works.[142] Scope is important for several reasons. First, as in the case of any other works, the scope defines the baseline delineation between the copyright "monopoly" and the public domain; exceptions and limitations and other factors may expand the public domain further,[143] but the scope of protectable subject matter is the first and arguably most significant delineation. Second, the adjustments that have been made to copyright law to accommodate computer programs assign significance to whether a work is classified as a computer program and covered by copyright law in its adjusted form, or whether it is classified as another category of copyrightable subject matter and covered by copyright law without the adjustments. Third, classifying works as computer programs and protecting them by copyright impacts the flexibility of achieving interoperability among computer programs—a feature on which software development heavily relies.

With OOP, questions of protectable scope became even more complicated than they were before the advent of the wide adoption of OOP. With OOP, developers create objects in a programming language and use the objects to create larger programs, spawning disputes that are presented as disputes over programming languages. The law seems to have been ambivalent about the protectability of programming languages. When the TRIPS agreement was being negotiated, Japan proposed that the draft agreement explicitly exclude from copyright protection "any programming language, rule or algorithm use[d] for making such works,"[144] but the Japanese proposal was not included in the final version of the TRIPS Agreement. The 1991 EC Directive mentions programming languages only once; one of its recitals states that "to the extent that logic, algorithms and programming languages comprise ideas and principles, those ideas and principles are not protected under this Directive,"[145] but the recital immediately thereafter adds that "the expression of those ideas and principles is to be protected by copyright."[146] A 2000 report on the implementation of the Directive noted that "there [was] no reason for extending the exceptions to the scope of protection under the Directive to include programming languages."[147]

---

140. *See, e.g.,* Navitaire Inc v. Easyjet Airline Co. [2004] EWHC 1725 (Ch) para. 98; Bundesgerichtshof [BGH] [Federal Court of Justice] Oct. 6, 2016, Case I ZR 25/15 *World of Warcraft I*, para. 34 (illustrating that computer program copyright law may apply not just to source or object code, but also to other types of copyrightable work). Computer programs may also include components that are protected by other types of IP rights, such as patents, trade secrets, and sui generis protection for databases.

141. Council Directive 91/250/EEC of May 14, 1991, on the legal protection of computer programs, 1991 O. J. (L 122) 42.

142. *E.g.*, Menell, *supra* note 50, at 322 (detailing the judicial struggle to develop software copyright jurisprudence).

143. *See, e.g.,* ALEXANDER PEUKERT, DIE GEMEINFREIHEIT. BEGRIFF, FUNKTION, DOGMATIK (2012) (discussing various definitions of the "public domain").

144. Resource Book on TRIPS and Development: An Authoritative and practical guide to the TRIPS Agreement by UNCTAD and ICTSD, 143, U.N. Doc. UNCTAD/ICTSD/2005/1.

145. Council Directive 91/250/EEC of May 14, 1991, *supra* note 145, at 43.

146. *Id.*

147. *Rep. from the Commission to the Council, the European Parliament and the Economic and Social Committee on the implementation and effects of Directive 91/250/EEC on the legal protection of computer programs*, at 16, COM (2000) 199 final (Apr. 10, 2000).

In Europe, the CJEU opined about the copyrightability of programming languages in *SAS Institute Inc. v. World Programming Ltd.*[148] The CJEU ruled that "the programming language and the format of data files used in a computer program in order to exploit certain . . . functions constitute a form of expression of that program and, as such, are *not* protected by copyright in computer programs."[149] At the same time, the CJEU recognized that "the. . . language and the format of [the] data files might be protected, as works, by copyright. . .if they are their author's own intellectual creation."[150]

The CJEU's holding concerning the SAS programming language might seem self-contradictory but it is consistent with the CJEU's 2010 decision in *Bezpečnostní softwarová asociace*,[151] where the CJEU held that a graphic user interface was also "merely . . . one element of [the computer] program by means of which users make use of the features of that program"[152] and therefore "cannot be protected specifically by copyright in computer programs by virtue of [the EC Software Directive],"[153] although it could be protected by copyright as a separate (e.g., pictorial) work.[154]

After *SAS Institute* returned from the CJEU to the national (English) court, Justice Arnold expressed his skepticism regarding the copyrightability of programming languages:

> [A] language is the material from which works (including dictionaries and grammars) may be created. . . Even when a language is created from scratch, however, what it amounts to is a system of rules for the generation and recognition of meaningful statements. Programming languages such as the SAS Language are no different in this respect.[155]

The case was eventually decided in the United Kingdom and the United States without a final ruling on the copyrightability of the SAS programming language in either country.[156]

At about the time the *SAS Institute* UK and US cases were filed, the issue of copyrightability of the elements of Java was raised in the US in *Oracle v. Google*. Oracle, after it acquired Java from Sun Microsystems, Inc., sued Google in 2010 for infringement of its copyright to Java, on which Google built its Android operating system.[157]

---

148. Case C-406/10, SAS Institute Inc. v. World Programming Ltd, 2012 EUR-Lex CELEX LEXIS (May 2, 2012).

149. *Id.* para. 46.

150. *Id.* para. 45.

151. Case C-393/09, Bezpečnostní softwarová asociace v. Ministerstvo kultury, 2010 CELEX 62009CJ0393 (Dec. 22, 2010).

152. *Id.* para. 41.

153. *Id.* para. 42.

154. *See id.* para. 44 (stating that it is appropriate to ascertain whether the graphic interface user can be protected by the ordinary law of copyright).

155. SAS Institute Inc. v. World Programming Ltd. [2013] EWHC (Ch) 69 [33], [2013] R.P.C. 17 (Eng.).

156. SAS Institute Inc. v. World Programming Ltd. [2013] EWCA (Civ) 1482, [2014] R.P.C. 8 (Eng.).; *see* SAS Institute Inc. v. World Programming Ltd., 64 F.Supp.3d 755, 778 (E.D.N.C. 2014) (holding that the defendant did "not infringe any of the plaintiff's copyrights"); *see* SAS Institute Inc. v. World Programming Ltd., 874 F.3d 370, 390 (4th Cir. 2017) (holding the copyright issue as moot)., *cert. denied*, World Programming Limited v. SAS Institute, Inc., 139 S.Ct. 67 (2018); SAS Institute Inc. v World Programming Ltd. [2018] EWHC (Comm), [2019] F.S.R. 30 (Eng.).

157. *See* Menell, *supra* note 50, at 374–75. Oracle's acquisition of Sun was completed in 2010.

The case is similar to *SAS Institute* in that it involves replications of programming objects. In *SAS Institute*, the objects at issue were "SAS Components,"[158] and the defendant created a program that enabled customers to execute, using the defendant's program, their applications that were originally created for SAS Components.[159] In *Oracle v. Google*, the objects were elements of Java's Application Programming Interface ("API"). Google copied the names and functions of 37 API packages to make it easy for developers using Java to write programs for Android, but Google wrote its own code to implement most of the packages. The U.S. Court of Appeals for the Federal Circuit held in 2014 that Google had copied "the declaring code and the structure, sequence, and organization of the 37 Java API packages"[160] which were protected by copyright.[161] In 2018 it held that Google's conduct was not protected by the fair use doctrine.[162] In January 2019 Google filed a petition for writ of certiorari, which the U.S. Supreme Court granted in November 2019.[163]

The copyrightability of objects in OOP presents yet another set of issues for which copyright law seeks to accommodate the specificities of software as a protectable subject matter. Some commentators have predicted that OOP will exacerbate the problems of applying copyright law to computer programs;[164] only time will tell if copyright law can continue to be flexible enough to accommodate this complex subject matter.

If there is to be an accommodation, part of any accommodation will be meeting the demands for interoperability, reverse engineering, and testing. Concerns about interoperability have always permeated the discussions regarding copyright law and software,[165] and *SAS Institute* and *Oracle v. Google* both emphasized the concerns.[166] Interoperability is crucial because of the nature of software development; the importance of interoperability has increased in the interconnected environment facilitated by the Internet, particularly for the Internet of things.[167] Related to the interconnected environment is the

158. Object-oriented Programming and the SAS Component Object Model, SAS(R) Component Language 9.2: Reference, http://support.sas.com/documentation/cdl/en/sclref/59578/HTML/default/viewer .htm#a000635650.htm (last visited August 16, 2019).

159. *See* SAS Institute Inc. v. World Programming Ltd. [2013] EWHC (Ch) 69 [33] ("WPL sought to emulate much of the functionality of the SAS Components as closely as possible in the sense that, subject to only a few minor exceptions, it tried to ensure that the same inputs would produce the same outputs. This was so as to ensure that WPL's customers' application programs executed in the same manner when run on WPS as on the SAS Components.").

160. Oracle America, Inc. v. Google Inc., 750 F.3d 1339, 1381 (Fed. Cir. 2014).

161. *Id.*

162. *See generally* Oracle America, Inc. v. Google LLC, 886 F.3d 1179 (Fed. Cir. 2018), *cert. granted* Google LLC v. Oracle America, Inc., No. 18-956, 2019 WL 6042317, at *1 (2019).

163. *See generally* Google LLC v. Oracle America, Inc., No. 18-956, 2019 WL 6042317, at *1 (2019).

164. *See* Lipton, *supra* note 8, at 213–14 (noting that although some judges have suggested the drafting of more unusual or original codes to avoid the merger doctrine and attract copyright protections as programs become more object oriented, this approach might ultimately lead to inefficiencies in programming).

165. *See* Menell, *supra* note 50, at 323–42 (outlining the early cases that addressed copyright protection for interoperable software and the later development in courts of alternative approaches to copyright protection).

166. Oracle America, Inc. v. Google Inc., 872 F.Supp.2d 974, 999–1000 (N.D.Ca. 2012), *rev'd,* 750 F.3d 1339 (Fed. Cir. 2014); *but cf.* Oracle America, Inc. v. Google LLC, 886 F.3d at 1206 n.11 ("Google has abandoned the arguments it once made about interoperability. This change in course is not surprising given the unrebutted evidence that Google specifically designed Android to be *incompatible* with the Java platform and not allow for interoperability with Java programs."). [also, 886 F.3d was already cited in note 2, so can we make this a supra?]

167. *See generally,* Gratz & Lemley, supra *note* 54, at 612; *See* Mehmet Bilal Ünver, *Turning the Crossroad for A Connected World: Reshaping the European Prospect for the Internet of Things*, 26 INT'L J. L. AND INFO. TECH. 2, 93–118 (2018) (providing an introduction to the concepts of "IoT" and "interoperability").

pressure that is now on the law to allow for decompilation and reverse engineering. In *SAS Institute*, for example, reverse engineering was employed to achieve interoperability. Not surprisingly, both *SAS Institute* and *Oracle v. Google* involve popular and widely adopted programming languages.

Courts and legislators had addressed interoperability issues long before the OOP languages cases mentioned above were brought.[168] In the United States, Paul Goldstein suggested in 1986 that "courts may seek to resolve the compatibility problem by resort to the copyright doctrine of fair use,"[169] and the fair use doctrine has indeed facilitated reverse engineering and testing conducted for the purposes of achieving interoperability.[170] The EU Directive responded to interoperability demands by allowing developers "to observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program,"[171] and also to decompile code if this is "indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs."[172]

Demands for interoperability also prompted carve-outs to the provisions on protection of technological protection measures, which were newly introduced into copyright law in the 1990s. The WIPO Copyright Treaty, which solidified protection for the measures at the international level, does not mention computer programs,[173] but national legislators responded to the needs of interoperability. For example, section 1201 of the US Copyright Act allows for circumvention of the measures for purposes of "identifying and analyzing those elements of the [computer] program that are necessary to achieve interoperability."[174] The EU Information Society Directive, which mandates the protection of technological protection measures in the European Union, explicitly excludes computer programs from the scope of the provisions and refers to the EU Software Directive to "exclusively determine exceptions to the exclusive rights applicable to computer programs."[175] The free trade agreements that have been concluded since TRIPS include exceptions similar to the exception included in the U.S. Copyright Act.[176]

## CONCLUSION

Not long ago, two prominent commentators remarked that the relationship between copyright and software "was never a happy marriage."[177] As do the relationships in long marriages, the copyright and software relationship has required hard work on both sides to resolve the inevitable frictions that have arisen and to make the relationship work. As the

168. *See* Charles R. McManis, *Taking TRIPS on the Info. Superhighway: Int'l Intell. Prop. Protection and Emerging Computer Tech.*, 41 VILLANOVA L. REV. 207, 232–52 (1996).

169. Goldstein, *supra* note 4, at 1129.

170. *E.g.*, Sega Enterprises v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992); Computer Associates International v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992); *see also* Menell, *supra* note 50.

171. Council Directive 2009/24, art. 5, 2009 O.J. (L 111) 3 (EC).

172. *Id.* at art. 6(1).

173. WIPO Copyright Treaty, *supra* note 6, at art. 11.

174. 17 U.S.C. §1201(f)(1) (1999).

175. Council Directive 2001/29, 2001 O.J. (L 167) (EC).

176. *See* Jonathan Band, *The Global API Copyright Conflict*, 31 HARV. J. OF L. & TECH. 615, 635–36 (2018) (quoting the exceptions in the US-Korea Free Trade Agreement and including a list of other Free Trade Agreements with similar exceptions).

177. Lothar Determann & David Nimmer, *Software Copyright's "Oracle" from the Cloud*, 30 BERKELEY TECH L. J. 161, 163 (2015).

overview in this article suggests, each side has, to some extent, accommodated the other, and the relationship has undeniably enriched both parties. The result of each side's adjustments is that, as another prominent commentator stated, "the case for copyrighting computer programs did become easier over time."[178]

The adjustments made to copyright law to fit it to computer programs could be viewed as "deviat[ions] from the classical copyright paradigm of the past,"[179] but they have become part of the current copyright paradigm. Two commentators suggested that the potential "tragedy" of computer programs "radically transforming copyright law and policy" had been "mostly" avoided.[180] Many questions remain, of course, and new questions continue to arise—questions concerning the impact of cloud computing,[181] the localization of GPLs, and the utility of collective licensing for computer programs and their elements[182] are a few examples.

Other commentators have contributed rich and comprehensive descriptions of the development of the law since the advent of the TRIPS Agreement and even prior to TRIPS.[183] This brief overview shows that both copyright law and software have significantly advanced since the days of the TRIPS negotiations and, despite the continuing skepticism of critics, it seems reasonable to suggest that in the unlikely event that TRIPS were to be re-opened for new negotiations, copyright protection for software would no longer be questioned. After a quarter century of marriage, software and copyright are not the same as when they first met.

178. Samuelson, *supra* note 8, at 1747.

179. Reichman, *The Know-How Gap*, *supra* note 9, at 776.

180. Daniel Gervais and Estelle Derclaye, *The Scope of Computer Program Protection after SAS: Are We Closer to Answers?*, 34(8) EIPR 565, 565 (2012).

181. *See generally*, Determann & Nimmer, *supra* note 181, at 201–08.

182. *See generally*, Bezpečnostní softwarová asociace v. Ministerstvo kultury, E.C.J., C-393/09, December 22, 2010.

183. *See* Menell, *supra* note 50 (providing a historical overview of copyright litigation and legislation in the United States).