Catholic University Journal of Law and Technology

Volume 30
Issue 1 *Fall 2021*

Article 3

2021

Hybrid IP Rights For Software, APIs, and GUIs: Understanding Copyright's Paradigm Shift

Howard Skaist
Berkeley Law & Technology Group, LLP

HYBRID IP RIGHTS FOR SOFTWARE, APIS AND GUIS: UNDERSTANDING COPYRIGHT'S PARADIGM SHIFT

Howard A, Skaist[*]

With advances in technology over many decades, copyright law's allocation of rights in our society and even its underlying assumptions have been called into question, particularly with respect to software. In times of technological

[*] Howard Skaist is the founder and senior counsel of Berkeley Law & Technology Group, LLP, (BLTG) an intellectual property law firm with offices in Oregon and Texas. Before founding BLTG, he was Director of Patents for Intel Corp. Likewise, he has taught law school in the past as an adjunct faculty member at Boalt Hall Law School (before its name was changed to the University of California, Berkeley *School of Law*); Lewis and Clark Law School; Willamette Law School; and Albany Law School. I wish to thank Sidney True for his encouragement in preparing this article. This article is based on some ideas I formulated in 2001. I started the article at that time, but the press of business caused me to put it aside, I thought permanently, but for what has turned into two decades. Considering the attention given to the case *Google v.* Oracle, recently decided by the Supreme Court on April 5, 2021, he encouraged me to complete it and, without his encouragement, I would not have done so.

change,[1] there usually are such claims from one extreme, for example, that copyright affords inadequate protection,[2] to another, that copyright law is operating as a hindrance to the creation of creative works.[3] This presents both a

---

1 *See, e.g.*, Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 820 (1st Cir. 1995) (Boudin, J., concurring) ("Congress has arguably recognized the tension and left it for the courts to resolve through the development of case law. And case law development is *adaptive*: it allows new problems to be solved with help of earlier doctrine, but it does not preclude new doctrines to meet new situations.") (emphasis in original); Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 711 (2d Cir. 1992) ("In this respect, our conclusion is informed by Justice Stewart's concise discussion of the principles that correctly govern the adaptation of the copyright law to new circumstances. In *Twentieth Century Music Corp. v. Aiken,* he wrote: 'The limited scope of the copyright holder's statutory monopoly, like the limited copyright duration required by the Constitution, reflects a balance of competing claims upon the public interest: Creative work is to be encouraged and rewarded, but private motivation must ultimately serve the cause of promoting broad public availability of literature, music, and the other arts.'").
Adapting copyright law in light of new technology is an age-old problem. For example, in discussing the introduction of photography as a technology, *Nimmer on Copyright* (hereinafter, 'NIMMER') notes the following:
And so copyright law had to confront questions of how to treat function-based productions. At the same time, the realistic portrayal of nature captured by a camera lens posed the related question of and how to treat fact-based productions. Those dilemmas regarding photography represent the beginning of the problems faced by copyright law in adjusting to new technology—but scarcely the end. Today's world, dominated by the Internet and with its profusion of software everywhere, poses unceasing pressure in defining the boundaries of where copyright law does and does not reach. In some sense, all the chapters of this treatise have been dragooned into those considerations.
1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.03 (2021).

2 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 711 ("At bottom, they claim that if programmers are not guaranteed broad copyright protection for their work, they will not invest the extensive time, energy and funds required to design and improve program structures."); Eileen McDermott, *Justices Look for Reassurance That the Sky Won't Fall When They Rule in Google v. Oracle,* IPWATCHDOG (Oct. 7, 2020), https://www.ipwatchdog.com/2020/10/07/justices-look-reassurance-sky-wont-fall-google-v-oracle/id=126052/ ("We've heard dire predictions from Google about the future of software innovation, but two different administrations would not be supporting us if our position were a threat to innovation."); Bill Donahue, *Justices Wary of 'Sky Falling' in Google-Oracle Case*, LAW360 (Oct. 7, 2020), https://www.law360.com/articles/1317786/justices-wary-of-sky-falling-in-google-oracle-software-case.

3 *See Comput. Assocs. Int'l, Inc*., 982 F.2d at 712 ("[S]erious students of the industry have been highly critical of the sweeping scope of copyright protection engendered by the *Whelan* rule, in that it 'enables first comers to 'lock up' basic programming techniques as implemented in programs to perform particular tasks.'"); Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1238 (3d Cir. 1986) ("As a consequence, this commentator argues, giving computer programs too much copyright protection will retard progress in the field. We are not convinced that progress in computer technology or technique is qualitatively different from progress in other areas of science or the arts. In balancing protection and dissemination . . . the copyright law has always recognized and tried to accommodate the fact that all intellectual pioneers build on the work of their

philosophical and a practical problem. Specifically, how copyright law should and/or how copyright law has accommodated the latest developments in technology.[4] Long ago, when computer technology and software first appeared on the scene, it was decided to provide some protection for software within the doctrines of copyright law.[5] Since then, questions about the challenges and/or desirability of affording meaningful protection for software and related technology under copyright law have continued to be debated by many.[6]

---

predecessors."); McDermott, *supra* note 2 ("Oracle has a copyright to the computer code, not a patent. That means the public, not Oracle, has the right to these functions."); Donahue, *supra* note 2.

4 Adapting copyright law in light of new technology is an age-old problem. For example, in discussing the introduction of photography as a technology, *Nimmer* notes the following:

And so copyright law had to confront questions of how to treat function-based productions. At the same time, the realistic portrayal of nature captured by a camera lens posed the related question of and how to treat fact-based productions. Those dilemmas regarding photography represent the beginning of the problems faced by copyright law in adjusting to new technology—but scarcely the end. Today's world, dominated by the Internet and with its profusion of software everywhere, poses unceasing pressure in defining the boundaries of where copyright law does and does not reach. In some sense, all the chapters of this treatise have been dragooned into those considerations.

NIMMER ON COPYRIGHT § 2A.03, *supra* note 1.

5 *See infra* notes 87–112 and accompanying text.

6 Pamela Samuelson, *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, 31 BERKELEY TECH. L. J. 1215, 1219 (2016) [hereinafter *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*]; *See, e.g.*, Pamela Samuelson, *Staking the Boundaries of Software Copyrights in the Shadows of Patents*, 71 FLA. L. REV. 243, 243–49 (2019) [hereinafter *Staking the Boundaries of Software Copyrights in the Shadow of Patents*]; 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. L. & TECH. (SPECIAL ISSUE) 305, 307–13 (2018); Timothy K. Armstrong, *Symbols, Systems and Software as Intellectual Property: Time for CONTU, Part II?*, 24 MICH. TELECOMM. & TECH. L. REV. 131, 131–37 (2019); Daria Vasilescu-Palmero, *APIs and Copyright Protection: The Potential Impact on Software Compatibility in the Programming Industry*, 16 J. MARSHALL REV. INTELL. PROP. L. 153, 154–56 (2016); Pamela Samuelson, *The Uneasy Case for Software Copyrights Revisited*, 79 GEO. WASH. L. REV. 1746, 1746–48 (2011) [hereinafter *The Uneasy Case for Software Copyrights Revisited*]; Peter Menell, *Tailoring Legal Protection for Computer Software*, 39 STAN. L. REV. 1329, 1329–32 (1987) [hereinafter *Tailoring Legal Protection for Computer Software*]; 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, 2559–60 (1994); Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 JURIMETRICS 33, 33–36 (1987); Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 STAN. L. REV. 1045, 1045–49 (1989) [hereinafter *An Analysis of the Scope of Copyright Protection for Application Programs*]; *see, e.g,* J.H. Reichman, *Computer Programs as Applied Scientific Know-How: Implications of Copyright Protection for Commercialized University Research*, 42 VAND. L. REV. 639, 641–42 (1989); Lloyd L. Weinreb, *Copyright for Functional Expression*, 111 HARV. L. REV. 1149, 1149–52 (1998); Steven R. Englund, *Idea, Process, or Protected Expression?: Determining the Scope of*

However, as is well-known, copyright law itself came into existence as a response to a new technology, the printing press.[7] Therefore, in some larger sense, perhaps it is only destiny of a sort that an intellectual property law, born of new technology, becomes reinvented as technology continues to evolve and change.[8]

Copyright Protection of the Structure of Computer Programs, 88 MICH. L. REV. 866, 866–67 (1990); *see generally* Arthur Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?***,** 106 HARV. L. REV. 977 (1993) (offering competing arguments about the scope of copyright protection for software); Pamela Samuelson et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994) [hereinafter *A Manifesto Concerning the Legal Protection of Computer Programs*]; David Nimmer et al., *A Structured Approach to Analyzing Substantial Similarity of Computer Software in Copyright Infringement Cases*, 20 ARIZ. ST. L. J. 625, 625–27 (1988) (acknowledging the complexity of litigating high-tech copyright infringement cases and advocating for a simplifying test inquiring whether the two programs are substantially similar); 4 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 13.03(F) (2021); JONATHAN BAND & MASANOBU KATOH, INTERFACES ON TRIAL: INTELLECTUAL PROPERTY AND INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY (1995) (weighing the scope of intellectual property protections within the context of technology and industry); JONATHAN BAND & MASANOBU KATOH, INTERFACES ON TRIAL 2.0 (Laura DeNardis & Michael Zimmer eds., 2011) (analyzing debates relating to interoperability, U.S. copyright cases, and contractual limitations on reverse engineering); *Special Issue: Software Interface Copyright*, 31 HARV. J. L. & TECH. (SPECIAL ISSUE) 303 (2018) [hereinafter *Special Issue: Software Interface Copyright*] (devoting its Spring Issue to the topic of Copyright and Software Interfaces with 9 articles by well-known intellectual property law experts including: Pamela Samuelson, Peter Menell, and Mark Lemley).

7 BEATRICE WARDE & S.H. STEINBERG, FIVE HUNDRED YEARS OF PRINTING 95 (Dover, 1st ed. 2017).

8 *See, e.g.*, Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 820 (1st Cir. 1995) (Boudin, J., concurrence) ("Congress has arguably recognized the tension and left it for the courts to resolve through the development of case law. And case law development is *adaptive*: it allows new problems to be solved with help of earlier doctrine, but it does not preclude new doctrines to meet new situations" (emphasis in original)); *Comput. Assocs. Int'l, Inc.*, 982 F.2d 711 ("In this respect, our conclusion is informed by Justice Stewart's concise discussion of the principles that correctly govern the adaptation of the copyright law to new circumstances. In *Twentieth Century Music Corp. v. Aiken*, he wrote: 'The limited scope of the copyright holder's statutory monopoly, like the limited copyright duration required by the Constitution, reflects a balance of competing claims upon the public interest: Creative work is to be encouraged and rewarded, but private motivation must ultimately serve the cause of promoting broad public availability of literature, music, and the other arts.'").

Adapting copyright law in light of new technology is an age-old problem. For example, in discussing the introduction of photography as a technology, NIMMER notes the following:

> And so copyright law had to confront questions of how to treat function-based productions. At the same time, the realistic portrayal of nature captured by a camera lens posed the related question of and how to treat fact-based productions. Those dilemmas regarding photography represent the beginning of the problems faced by copyright law in adjusting to new technology—but scarcely the end. Today's world, dominated by the Internet and with its profusion of software everywhere, poses unceasing pressure in defining the boundaries of

Legal scholars have spent decades debating the appropriate scope of copyright law protection for computer programs. In this debate, some advocate that the traditional bedrock principles of copyright law either *should not change and/or have not changed* to appropriately and suitably accommodate software.[9] This article will demonstrate, however, that *the current state of copyright law reflects a clear shift (referred to here as a "paradigm shift") of some fundamental copyright law principles* to suitably handle the rights to be afforded software under copyright law. This shift came about to address the unique problems that come from the nature of software as a form of copyrightable expression.

Rather, as many others have already observed, software represents a form of expression, unique in that it is both expressive and useful.[10] This aspect of

---

where copyright law does and does not reach. In some sense, all the chapters of this treatise have been dragooned into those considerations.
NIMMER ON COPYRIGHT § 2A.03, *supra* note 1.

9 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 706 ("This approach breaks no new ground; rather, it draws on such familiar copyright doctrines as merger, *scène à faire*, and public domain."); NIMMER ON COPYRIGHT § 13.03(F), *supra* note 6 ("Applying the 'successive filtering' test set form above merely involves examining the works in controversy in light of each of the doctrines canvassed in the preceding subsections."); *see, e.g.*, Samuelson, *supra* note 6; Menell, *supra* note 6; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1215, 1217–19; Armstrong, *supra* note 6, at 134–36; Vasilescu-Palmero, *supra* note 6, at 155–56; *The Uneasy Case for Software Copyrights Revisited*, *supra* note 6, at 1748; *Tailoring Legal Protection for Computer Software*, *supra* note 6, at 1331–32; Ginsburg, *supra* note at 6, at 2560–62; Karjala, *supra* note 6, at 33–35; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1046–50; Reichman, *supra* note 6, at 641–42; *A Manifesto Concerning the Legal Protection of Computer Programs*, *supra* note 6, at 2310–14; Weinreb, *supra* note 6, at 1150–54; Englund, *supra* note 6, at 866–67; Nimmer, *supra* note 6; NIMMER ON COPYRIGHT § 13.03(F), *supra* note 6; *see generally* Miller, *supra* note 6, at 1072 (arguing that no evidence suggests that CONTU or Congress was wrong in bringing copyright protection to "computer programs, databases, and computer-assisted works."); INTERFACES ON TRIAL: INTELLECTUAL PROPERTY AND INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY, *supra* note 6 (weighing the scope of intellectual property protections within the context of technology and industry); INTERFACES ON TRIAL 2.0, *supra* note 6 (analyzing debates relating to interoperability, U.S. copyright cases, and contractual limitations on reverse engineering); *see also Special Issue: Software Interface Copyright*, *supra* note 6 (devoting its Spring Issue to the topic of Copyright and Software Interfaces with 9 articles by well-known intellectual property law experts including: Pamela Samuelson, Peter Menell, and Mark Lemley).

10 *See, e.g.*, *Lotus Dev. Corp*., 49 F.3d at 819 (concurrence) ("The problem presented by computer programs is fundamentally different in one respect. The computer program is a *means* for causing something to happen; it has a mechanical utility, an instrumental role, in accomplishing the world's work. Granting protection, in other words, can have some of the consequences of *patent* protection in limiting other people's ability to perform a task in the most efficient manner. Utility does not bar copyright (dictionaries may be copyrighted), but it alters the calculus." (emphasis in original)); *Comput. Assocs. Int'l, Inc*., 982 F.2d at 712 ("This results from the hybrid nature of a computer program, which, while it is literary expression, is also a highly functional, utilitarian component in the larger process of computing.").

software makes it problematic in terms of traditional concepts of copyright law and, more specifically, the scope of protection to be afforded to software by copyright law.[11] In this article, we refer to the protection of features under copyright law that are both expressive and useful as *hybrid* intellectual property rights.[12] Traditional copyright law, in general, does not provide protection for

---

[11] *Nimmer* observes:
Wherever we look closely, the functional impinges on the artistic . . . Literary works are the oldest form of protectable compositions—but, defined to include any text composed in alphanumeric characters, that category currently embraces computer software. Software itself can be conceptualized as "a part of a machine," namely the component that instructs a computer how to operate . . . An inquiry into copyright protection for computer software generates limitless questions of what should be protected and where to draw the line for infringing similarity of competing codes.
NIMMER ON COPYRIGHT §2A.03(a), *supra* note 1.

[12] Likewise, without drifting too far from the main topic, it is observed that a more general problem may be considered to emanate from an intersection between utility (or functionality) and form (or appearance). That is, across a variety of types of intellectual property, a general problem may occur in situations in which protection is sought for a "res," so to speak, of which software is but one example, that exhibits features in which the utility of those features and the form of those features are intertwined sufficiently closely that a clear separation of utility and form may not be possible. Ostensibly, this may create a problem in these other areas of intellectual property because concerns around utility in intellectual property law generally are relegated to patent law, with its high standards of novelty and non-obviousness. *See* Baker v. Selden, 101 U.S. 99, 102–03 (1880) ("The novelty of the art or thing described or explained has nothing to do with the validity of the copyright. To give to the author of the book an exclusive property in the art described therein, when no examination of its novelty has ever been officially made, would be a surprise and a fraud upon the public. That is the province of letters-patent, not of copyright."); Pamela Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 DUKE L.J. 663, 732–35 (1984) (explaining reasons why utilitarian works have conventionally been excluded from copyright protection). Examples beyond copyright have arisen, for example, in the areas of design patents and trademark/trade dress protection. *See* Afori Fischman Orit, *The Role of the Non-Functionality Requirement in Design Law*, 20 FORDHAM INTELL. PROP. MEDIA & ENT. L.J. 847, 849–50 (2010); Robert G. Bone, *Trademark Functionality Reexamined*, 7 J. LEGAL ANALYSIS 183, 184 (2015). Many key differences, however, exist between the scope of protection provided by a patent versus a copyright, which, of course, necessarily includes software. For a patent – *the scope of protection is defined by the claims, which are prepared by the applicant*. *See*, *e.g.*, Vitronics Corp. v. Conceptonic, Inc., 90 F.3d 1576, 1582–83 (Fed. Cir. 1996). For a copyright, *the scope of protection is determined by substantial similarity of the expression of a copyrightable work to an accused work. See*, *e.g.*, Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1541, 1543–44 (11th Cir. 1995). It would be extremely difficult for the scope of protection available for a computer program via copyright, including with respect to non-literal protection, to approach the scope of protection able to be garnered via patent protection. Thus, it is believed that concerns about overlap may be overstated and/or misplaced. A patent typically covers making, using, selling, offering for sale or importing tangible apparatuses (or processes), for example, that fall within the scope of the patent claims. *See* 35 U.S.C. § 271(a) (2021). A copyright covers copying (used in the broad, non-literal sense) – albeit, even considering

subject matter that is both expressive and useful.[13] In this regard the words of the court in *Computer Associates v. Altai (hereinafter, Altai)* are apropos: "To be frank, the exact contours of copyright protection for non-literal program structure are not completely clear . . . . *This results from the hybrid nature of a computer program, which, while it is literary expression, is also a highly functional, utilitarian component in the larger process of computing*."[14]

This article will demonstrate that software, because of this overlap or intersection between utility and form, is *handled differently in fundamental ways from other more traditional forms* of copyrightable expression (e.g., literary works, photographs, music, movies, etc.). It is for this reason that in this article the treatment of software under copyright law is referred to as *hybrid* intellectual property rights, mimicking the previous quote from *Altai* observing the hybrid nature of software.[15]

This article will show that copyright law may be approached as *having two regimes of protection* – first, a traditional copyright law regime and, second, a *hybrid* protection regime, the latter regime provided specifically for software. The rights are hybrid rights in part because the expression is protected although the expressive features and *the useful features are not capable of being*

---

non-literal scope, this at most relates to 'use' only to the extent that copying of a program indirectly includes use – and this applies only where there are a myriad of ways to accomplish the same task (e.g., absent merger). *See*, *e.g.*, discussion *infra* Section II.B.1. We also observe that the intellectual property clause of the Constitution, Article 1, Section 8, Clause 8, gives Congress the power "[t]o promote the Progress of Science and Useful arts, by securing, for limited Times, to Authors and Inventors, the exclusive Right to their respective Writings and Discoveries," and on its face has few clear limits imposed regarding this particular Congressional power other than that the protection be "for limited times" and be with respect to "writings" and "discoveries." U.S. CONST. art. I, § 8, cl. 8. Several Supreme Court cases have sought to construe various terms of this provision, such as the term "writings." *See, e.g.*, Mazer v. Stein, 347 U.S. 201, 206 n.5 (1954); Feist Publ'ns, Inc v. Rural Tel. Serv. Co., 499 U.S. 340, 345–46 (1991). Granting that there may be limits on the exercise of this Congressional power, either by logical construction and/or court decisions, it should be noted that what Congress may be unable to do under one of its powers, it may still be able to do under another power, such as under the interstate commerce power, for example. *Cf.* Trade-Mark Cases, 100 U.S. 82, 96–97 (1879). In this latter case, the Supreme Court held a federal trademark statute unconstitutional; however, afterwards, Congress re-enacted another federal trademark statute and made it clear it was relying on the interstate commerce power and the statute since then appears to have been accepted as constitutional. *See Mazer*, 347 U.S. at 206–08.

13 *See, e.g.*, Kieselstein-Cord v. Accessories by Pearl, Inc., 632 F.2d 989, 993 (2d Cir. 1980) (finding that the conceptually separable artistic elements of belt buckles are copyrightable); Brandir Int'l, Inc. v. Cascade Pac. Lumber Co., 834 F.2d 1142, 1147–48 (2d Cir. 1987) (finding the ribbon design for a bicycle rack could not be protected by copyright law as a useful article lacking separable expression).

14 *Comput. Assocs. Int'l, Inc*., 982 F.2d at 712 (emphasis added).

15 *Id.*

separately delineated or disaggregated.[16] A point that has tripped up various courts, for example, as discussed *infra*.[17]

Significantly, the scope of protection that is employed to handle the balance of competition and protection is quite different for software.[18] However, rather than suggest that software should be approached more traditionally, as some have implied or even stated,[19] a thesis of this article is that this treatment for

---

16 As we shall see, another reason for referring to these rights as hybrid is that the rights share some considerations that make them seem a bit "patent-like," while also clearly having considerations that makes them "copyright-like." *See infra* notes 341–52 and accompanying text. Some commentators have taken note about the potential overlap between protection provided via a software copyright and protection afforded through patent protection. *See* Samuelson, *supra* note 6, at 283–85; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1287–88; 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.07(B) (2021). Many key differences, however, exist between the scope of protection provided by a patent versus a copyright, which, of course, necessarily includes software. For a patent – *the scope of protection is defined by the claims, which are prepared by the applicant*. *See, e.g*., *Vitronics Corp*., 90 F.3d at 1582; For a copyright, *the scope of protection is determined by substantial similarity to the expression of a copyrightable work to an accused work. See, e.g*., *Bateman*, 79 F.3d at 1545. It would be extremely difficult for the scope of protection available for a computer program via copyright, including with respect to non-literal protection, to approach the scope of protection able to be garnered via patent protection. Thus, it is believed that concerns about overlap may be overstated and/or misplaced. A patent typically covers making, using, selling, offering for sale or importing tangible apparatuses (or processes), for example, that fall within the scope of the patent claims. *See* § 271(a). A copyright covers copying (used in the broad, non-literal sense)—albeit, even considering non-literal scope, this at most relates to "use" only to the extent that copying of a program indirectly includes use—and this applies only where there are a myriad of ways to accomplish the same task (e.g., absent merger). *See, e.g., infra* notes 52–87 and accompanying text. It also observed that the intellectual property clause of the Constitution, Article 1, Section 8, Clause 8, gives Congress the power "[t]o promote the Progress of Science and Useful arts, by securing, for limited Times, to Authors and Inventors, the exclusive Right to their respective Writings and Discoveries," and on its face has few clear limits imposed regarding this particular Congressional power other than that the protection be "for limited times" and be with respect to "writings" and "discoveries." U.S. CONST. art. I, § 8, cl. 8. Several Supreme Court cases have sought to construe various terms of this provision, such as the term "writings." *See*, e.g., *Mazer*, 347 U.S. at 206 n.5; *Feist Publ'ns, Inc*., 499 U.S. at 345; Granting that there may be limits on the exercise of this Congressional power, either by logical construction and/or court decisions, it should be noted that what Congress may be unable to do under one of its powers, it may still be able to do under another power, such as under the interstate commerce power, for example. *Cf. Trade-Mark Cases*, 100 U.S. at 95–97. In this latter case, the Supreme Court held a federal trademark statute unconstitutional; however, afterwards, Congress re-enacted another federal trademark statute and made it clear it was relying on the interstate commerce power and the statute since then appears to have been accepted as constitutional. *See Mazer*, 347 U.S. at 206–07.

17 *See infra* notes 113–265 and accompanying text.

18 *See infra* notes 65–93 and accompanying text.

19 *See, e.g., Staking the Boundaries of Software Copyrights in the Shadows of Patents*,

software is appropriate and should be continued, at least based on the legal analysis presented.[20] The recently decided Supreme Court case, *Google LLC v. Oracle America*, Inc.[21] at least indirectly also supports this view.[22] Furthermore, as a matter of policy, discussion is provided to suggest that perhaps such treatment may also be appropriate for other examples of copyrightable expression that exhibit similarly hybrid characteristics (e.g., hybrid-like characteristics).

Some, if not many, of the considerations raised in this article are not necessarily new.[23] Rather, the various considerations are organized and presented in this article in a manner so that a more consistent and simpler analytical framework is emergent to explain the major court decisions and the various policy considerations for an otherwise relatively complex and potentially uncertain area of law. Such a presentation is provided to correct some discrepancies that may have crept into the discourse and, therefore, permit appropriate evaluation of the current direction of copyright law in the area of software protection.

In the first section, the traditional legal principles of copyright law, the statutory developments that took place to deal with software, and the seminal cases in which courts interpreted the consequences of these developments regarding software for the law of copyright will be reviewed. [24] In the second section, the resulting legal landscape will be discussed to demonstrate that, *if properly interpreted, a shift took place in which some traditional principles of*

---

supra note 6, at 256; Menell, *supra* note 6, at 438; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1215; Armstrong, *supra* note 6, at 180; Vasilescu-Palmero, *supra* note 6, at 169; *The Uneasy Case for Software Copyrights Revisited*, *supra* note 6, at 1781–82; *Tailoring Legal Protection for Computer Software*, *supra* note 6, at 1369–71; Ginsburg, *supra* note at 6, at 2572; Karjala, *supra* note 6, at 95–96; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1103–04; Miller, *supra* note 6, at 978; Reichman, *supra* note 6, at 642; *A Manifesto Concerning the Legal Protection of Computer Programs*, *supra* note 6, at 2429–30; Weinreb, *supra* note 6, at 1150; Englund, *supra* note 6, at 867; Nimmer, *supra* note 6, at 651; NIMMER ON COPYRIGHT § 13.03, *supra* note 6; INTERFACES ON TRIAL: INTELLECTUAL PROPERTY AND INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY, *supra* note 6, at 46–47; INTERFACES ON TRIAL 2.0, *supra* note 6, at 1–2; *See generally Special Issue: Software Interface Copyright*, *supra* note 6 (in which the Harvard Journal of Law and Technology devoted its Spring Issue to the topic of Copyright and Software Interfaces with 9 articles by well-known intellectual property law experts including: Pamela Samuelson, Peter Menell, and Mark Lemley).

20 However, in the discussion, *infra.*, procedural mechanisms may address underlying assumptions regarding copying and the public domain. *See infra* notes 284–340 and accompanying text.

21 Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1195 (2021).

22 *See infra* notes 409–47 and accompanying text.

23 To the extent a consideration raised in this article has been known to have been raised elsewhere, a source citation is, of course, provided.

24 *See infra* Section I.

copyright law are not applicable to software. An interpretation of this shift relative to traditional copyright law will also be discussed. [25] Finally,[26] in the third and final section, the nature of hybrid intellectual property rights in copyright law with respect to software will be discussed to consider other potential examples of hybrid-like situations and whether the approach taken in software would be workable for other forms of expression that may be hybrid-like, such as application programing interfaces (APIs) and graphical user interfaces (GUIs). APIs are discussed in more detail, *infra*. GUIs are a type of user interface in which users interact with a computer program or an electronic device using visual indicators on a display.

I. COPYRIGHT LAW BEFORE AND AFTER THE PARADIGM SHIFT

A. Traditional Principles

Copyright Law includes several basic principles or doctrines. Many sources for these principles are available. Therefore, the principles will be discussed only to the extent appropriate to appreciate how software has been handled differently than other types of copyrightable works. Copyright law protects the original expression of ideas, not the ideas themselves.[27] Section 102(a) of Title 17 defines copyrightable subject matter: "copyright protection subsists . . . in original works of authorship fixed in any tangible medium of expression . . . "[28] Therefore, the copyright statute establishes that copyrightability requires originality[29] and fixation.[30]

The notion that copyright law protects the original expression of ideas and not the ideas themselves is referred to as the idea/expression dichotomy.[31]

---

25 *See infra* Section II.

26 *See infra* Section III.

27 17 U.S.C. § 102(a) (2021); *see, e.g.*, Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1234 (3d Cir. 1986); Peter Pan Fabrics, Inc. v. Martin Weiner Corp., 274 F.2d 487, 489 (2d Cir. 1960).

28 § 102(a).

29 *Id.*; *see* Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1247 (3d Cir. 1983); Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 351 (1991) (stating that originality is a constitutional requirement).

30 § 102(a); *See Apple Comput., Inc*., 714 F.2d at 1247; Midway Mfg. Co. v. Artic Int'l, Inc., 547 F. Supp. 999, 1007–08 (N.D. Ill. 1982) (concluding that the changing display of a video game nonetheless met the fixation requirement).

31 *See Feist Publ'ns, Inc*., 499 U.S. at 350 ("This principle, known as the idea/expression or fact/expression dichotomy, applies to all works of authorship."); *cf.* Mazer v. Stein, 347 U.S. 201, 214 (1954) ("They must be original, that is, the author's

Judge Learned Hand is credited with providing the most articulate description of how this works.[32] In the *Nichols v. Universal Picture Corp.* case, he stated:

> Upon any work, and especially upon a play, a great number of patterns of increasing generality will fit equally well, as more and more of the incident is left out. The last may perhaps be no more than the most general statement of what the play is about, and at times might consist only of its title; but there is a point in this series of abstractions where they are no longer protected, since otherwise the playwright could prevent the use of his "ideas," to which, apart from their expression, his property is never extended . . . . Nobody has ever been able to fix that boundary, and nobody ever can.[33]

Likewise, in another famous case, *Peter Pan Fabrics v. Martin Weiner Corp.*, the esteemed judge observed:

> The test for infringement of a copyright is of necessity vague. In the case of verbal "works" it is well settled that although the "proprietor's" monopoly extends beyond an exact reproduction of the words, there can be no copyright in the "ideas" disclosed but only in their "expression." Obviously, no principle can be stated as to when an imitator has gone beyond copying the "idea," and has borrowed its "expression." Decisions must therefore inevitably be ad hoc.[34]

This latter point, stating that decisions are inevitable ad hoc, pervades all of copyright law. It also indirectly plays a role in understanding software protection relative to the protection provided to other types of copyrightable works.

Thus, in general, it is helpful to appreciate that under traditional and fundamental copyright law principles, some amount of copying of a copyrightable work is legally permissible. In particular, the ideas from any copyrightable work may, at least in theory, be copied with impunity.[35] To be more specific, if a copyrightable work is published, the ideas of the copyrightable work enter the public domain to be available for anyone to freely copy.[36] In a real sense, this is a part of the *quid pro quo* between the author of the work and the government (or society, in general) for providing

---

tangible expression of his ideas.").

32 Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931).

33 *Id.*

34 Peter Pan Fabrics, Inc. v. Martin Weiner Corp., 274 F.2d 487, 489 (2d Cir. 1960).

35 The text says "in theory" primarily because identifying the ideas, as opposed to the expression, in any given work may be a challenge.

36 NIMMER ON COPYRIGHT § 13.03(F)(4), *supra* note 6.

copyright protection for the work. The ideas are available for others to build upon and, it is hoped, create additional creative and expressive works for the enjoyment of others. Without such a bargain, the goal of copyright law, to increase the available number of copyrightable works by creating appropriate incentives, might not be realized.[37] In particular, in connection with copyright law, it has been stated: "If I have seen further, it is by standing on the shoulders of giants," suggesting that most new creations are based on something that has been created in the past.[38] One aspect of these past creations is the ideas from such works that enter the public domain immediately.[39]

However, per Learned Hand's observation, what guides the determination of the boundary between ideas and expression? One job for a court faced with determining the scope of protection for a work is to identify the appropriate idea/expression boundary.[40] In this manner, over time, for particular categories of works, guidance will exist as to roughly where that boundary is located.[41] This then makes it possible for individuals to conduct their affairs and transact business taking this into account.[42] This was possibly best articulated in a Ninth Circuit case, *Herbert Rosenthal Jewelry Corp. v. Kalpakian* (hereinafter *Herbert Rosenthal*), which also cited to the Judge Learned Hand's helpful observations quoted above:

> The critical distinction between "idea" and "expression" is difficult to draw. As Judge Hand candidly wrote, 'Obviously, no principle can

---

37 *See, e.g.*, Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 696 (2d Cir. 1992) ("[C]opyright law seeks to establish a delicate equilibrium. On the one hand, it affords protection to authors as an incentive to create, and, on the other, it must appropriately limit the extent of that protection so as to avoid the effects of monopolistic stagnation. In applying the federal act to new types of cases, courts must always keep this symmetry in mind."); NIMMER ON COPYRIGHT § 2A.03(B), *supra* note 1 ("[W]e must approach the field wearing bifocals—artistic creativity deserves protection at the same time that the evils of monopolizing functional activities must be avoided. Decisions must therefore be reached with sensitivity to both sides of the ledger: If according protection to a given form of expression threatens to forestall competition in a given field of endeavor, that consideration alone might counsel the opposite resolution.").

38 *See* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1236 n.33 (3d Cir. 1986) ("Long before the first computer, Sir Isaac Newton humbly explained that 'if [he] had seen farther than other men, it was because [he] had stood on the shoulders of giants.'"); Michael D. Birnhack, *The Idea of Progress in Copyright Law*, 1 BUFF. INTELL. PROP. L.J. 3, 42 (2001); *see also* NIMMER ON COPYRIGHT § 13.03(B)(2)(a), *supra* note 6 ("[A] court posited long ago that copyright protection is granted for the very reason that it may persuade authors to make their ideas freely accessible to the public so that they may be used for the intellectual advancement of mankind.").

39 Another aspect is that the works becomes available for copying after the copyright of the work expires.

40 Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971).

41 *Id.*

42 *Id.*

> be stated as to when an imitator has gone beyond copying the "idea," and has borrowed its "expression"‘. . . At least in close cases, one may suspect, the classification the court selects may simply state the result reached rather than the reason for it. In our view, the difference is really one of degree as Judge Hand suggested in his striking "abstraction" formulation. . . . . *The guiding consideration in drawing the line is the preservation of the balance between competition and protection reflected in the patent and copyright laws*.[43]

Thus, the guiding consideration for a court seeking to identify an appropriate boundary between the ideas and the expression of a work is to strike a balance between competition and protection.[44] Furthermore, if it is not readily apparent, it stands to reason that the idea/expression line falls in a different place for a novel than it does for a photograph, for example. That is, the nature of a particular category of works suggests that this boundary be located differently for different categories of copyrightable works. In general, a court should seek to strike a balance so that incentives are structured to lead to the greatest number of works in the sense that the boundary should be located so that it is neither overprotective nor under protective.[45] It then follows that because different categories of works entail different forms of expression, this boundary should be placed differently for works in different categories, such as a novel in comparison to music or a novel in comparison to a movie, as simple examples.

In addition to originality and fixation, copyrightability requires some "minimal level of creativity."[46] Although time and effort may be involved in

---

43 *Id.* (emphasis added).

44 *Id.*

45 *See, e.g.*, Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 696 (2d Cir. 1992) ("[C]opyright law seeks to establish a delicate equilibrium. On the one hand, it affords protection to authors as an incentive to create, and, on the other, it must appropriately limit the extent of that protection so as to avoid the effects of monopolistic stagnation. In applying the federal act to new types of cases, courts must always keep this symmetry in mind."); Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1235 (3d Cir. 1986) ("In this regard, we must remember that the purpose of the copyright law is to create the most efficient and productive balance between protection (incentive) and dissemination of information, to promote learning, culture and development."); NIMMER ON COPYRIGHT § 2A.03(B), *supra* note 1 ("[W]e must approach the field wearing bifocals—artistic creativity deserves protection at the same time that the evils of monopolizing functional activities must be avoided. Decisions must therefore be reached with sensitivity to both sides of the ledger: If according protection to a given form of expression threatens to forestall competition in a given field of endeavor, that consideration alone might counsel the opposite resolution.").

46 *See* Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 358 (1991) ("Originality requires only that the author make the selection or arrangement independently (*i.e.*, without copying that selection or arrangement from another work), and that it display some minimal level of creativity.").

developing an idea into a creative form, it is not the time and effort that provide the copyrightable elements or expression. Rather, it is the creativity of the form.

How much creativity is enough? That is a lot like asking, "how much consideration does it take to form a contract?" Therefore, in the Supreme Court case, *Feist Publications, Inc. v. Rural Telephone Service Co.*, (hereinafter *Feist*) alphabetical listings of a white pages directory were held not to be sufficiently creative to justify copyright protection. [47] Nonetheless, in most cases, the amount of creativity it takes for a work to be eligible for copyright is relatively low, so that judges are not called upon to evaluate the quality of the creativity, much like in contract law, where judges are not called upon to evaluate the quality of a particular bargain once a threshold is met. [48]

Feist rejected the so-called "sweat of the brow" doctrine, a notion that somehow, time and effort may transform a non-creative work into a copyrightable work simply through expending sufficient effort. [49] *Feist* also determined that this is not simply a fanciful whim of Congress embedded in the statute. [50] This requirement goes to the constitutional core of copyrightability. [51]

While it may take countless hours to compile an alphabetical list of names, addresses and phone numbers for a white pages' compilation, that work is not copyrightable. Despite a major amount of effort put forth to create it, the work may be freely copied. [52] In contrast, the taking of a photo takes but a fraction of a second, and yet receives protection under the copyright act against unauthorized copying, to the point, even, that transfer of the copyright in the photo will prevent the original photographer from producing substantially similar photos that would constitute an infringement of the copyright transferred. [53]

Another important principle of copyright law is that the rights covered by a copyright do not include use rights. [54] This fundamental concept is attributed to a foundational Supreme Court case, *Baker v. Selden* (hereinafter *Baker v. Seldon* or *Baker*). [55] Rights over use, as it turns out, are relegated to

---

47 *Id.* at 363.
48 *Id.* at 349.
49 *Id.* at 352, 360.
50 *Id.* at 359–60.
51 *See id.* at 362 (stating that creativity is a constitutional requirement).
52 Assuming it has not been otherwise protected in another manner, such as, for example, via contractual restrictions and/or trade secret law.
53 *See, e.g.*, Gross v. Seligman, 212 F. 930, 931 (2d Cir. 1914).
54 Baker v. Selden, 101 U.S. 99, 105 (1880).
55 *See generally id.* (holding that the rights covered by a copyright do not include use).

patent law, rather than copyright law.[56] Again, this is a more than whimsical aspect of copyright law. Instead, it is a carefully orchestrated balancing of different intellectual property schemes that operate in conjunction to benefit society.[57] It appears that this was intended by the Framers of the Constitution from the way the relevant clauses of the Constitution are written and was intended by Congress when it exercised powers granted under the Constitution to create legislation to encourage such "writings."[58]

The Supreme Court in *Baker v. Selden*, does an excellent job of

---

[56] *Id.* at 102–03.

[57] Some commentators have taken note about the potential overlap between protection provided via a software copyright and protection afforded through patent protection. *See Staking the Boundaries of Software Copyrights in the Shadow of Patents*, *supra* note 6, at 289; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 57; NIMMER ON COPYRIGHT § 2A.07(D)(1), *supra* note 16. Many key differences, however, exist between the scope of protection provided by a patent versus a copyright, which, of course, necessarily includes software. For a patent – *the scope of protection is defined by the claims, which are prepared by the applicant*. *See, e.g.*, Vitronics Corp. v. Conceptronic, Inc., 90 F.3d 1576, 1583 (Fed. Cir. 1996). For a copyright, *the scope of protection is determined by substantial similarity to the expression of a copyrightable work to an accused work. See*, *e.g.*, Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1544 n.25 (11th Cir. 1995). It would be extremely difficult for the scope of protection available for a computer program via copyright, including with respect to non-literal protection, to approach the scope of protection able to be garnered via patent protection. Thus, it is believed that concerns about overlap may be overstated and/or misplaced. A patent typically covers making, using, selling, offering for sale or importing tangible apparatuses (or processes), for example, that fall within the scope of the patent claims. *See* 35 U.S.C. § 271(a) (2021). A copyright covers copying (used in the broad, non-literal sense) – albeit, even considering non-literal scope, this at most relates to 'use' only to the extent that copying of a program indirectly includes use – and this applies only where there are a myriad of ways to accomplish the same task (e.g., absent merger). *See*, *e.g.*, *supra* text accompanying notes 50–57; *infra* text accompanying note 58–86. We also observe that the intellectual property clause of the Constitution, Article 1, Section 8, Clause 8, gives Congress the power "[t]o promote the Progress of Science and Useful arts, by securing, for limited Times, to Authors and Inventors, the exclusive Right to their respective Writings and Discoveries," and on its face has few clear limits imposed regarding this particular Congressional power other than that the protection be "for limited times" and be with respect to "writings" and "discoveries." U.S. CONST. art. I, § 8, cl. 8. Several Supreme Court cases have sought to construe various terms of this provision, such as the term "writings." *See*, *e.g.*, Mazer v. Stein, 347 U.S. 201, 206–08 (1954); Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 346 (1991). Granting that there may be limits on the exercise of this Congressional power, either by logical construction and/or court decisions, it should be noted that what Congress may be unable to do under one of its powers, it may still be able to do under another power, such as under the interstate commerce power, for example. *Cf.* Trade-Mark Cases, 100 U.S. 82, 97–98 (1879). In this latter case, the Supreme Court held a federal trademark statute unconstitutional; however, afterwards, Congress re-enacted another federal trademark statute and made it clear it was relying on the interstate commerce power and the statute since then appears to have been accepted as constitutional. *See Mazer*, 347 U.S. at 206 n.5.

[58] *See Baker*, 101 U.S. at 102–103; *Mazer*, 347 U.S. at 219.

articulating the difference between rights afforded by copyright protection and rights related to use.[59] The copyrightable work at issue was a book explaining accounting, titled *Selden's Condensed Ledger, or Book-keeping Simplified.*[60] The alleged infringement revolved around copying the condensed ledger or sample tables within the book.[61] In concluding that copying the ledgers or using a similar ledger did not amount to copyright infringement, the Supreme Court stated:

> Now whilst no one has the right to print or publish his [i.e., Selden's] book, or any material part thereof, as a book intended to convey instruction in the art, any person may practice and use the art itself which he has described and illustrated therein. *The use of the art is a totally different thing from a publication of the book explaining it.* The copyright of a book on book-keeping cannot secure the exclusive right to make, sell, and use account-books prepared upon the plan set forth in such book. Whether the art might or might not have been patented, is a question which is not before us. It was not patented and is open and free to the use of the public.[62]

Thus, under *Baker v. Selden* where exclusive rights against copying of some otherwise copyrightable subject matter would, by providing protection under copyright, encompass or ensnare exclusive rights over use, then copyright protection must fail, as exceeding the scope of rights that are intended to be conveyed under copyright law.[63] Here, this doctrine is termed the "form-function doctrine," meaning that the scope of protection afforded by copyright may be limited where copyrightable form overlaps with uncopyrightable (e.g., utilitarian) function, although others may refer to this as the useful article(s) doctrine.[64] As discussed later, the exception afforded to software from this principle has been a source of confusion and consternation for courts charged with the task of determining the scope of protection for software in copyright law.[65]

Although seemingly easy to state, even before the advent of computers and software, this principle was not so easy to apply. The potential conflict between use rights and copyrightable subject matter with respect to the form-

---

59 *Baker*, 101 U.S. at 103.
60 *Id.* at 100.
61 *Id.*
62 *Id.* at 104 (emphasis added).
63 *Id.* at 107.
64 *See* Mark McKenna & Christopher Springman, *What's in, and What's out: How IP's Boundary Rules Shape Innovation*, 30 HARV. J. L. & TECH. 491, 535–39 (2017) (referring to the "useful articles" doctrine).
65 *See infra* notes 113–329 accompanying text.

function doctrine, thus, was again taken up by the Supreme Court in *Mazer v. Stein.*[66] (hereinafter *Mazer* or *Mazer v. Stein*) This case involved the copyrightability of a lamp base, although, in this case, the lamp base had the shape of a Balinese Dancer.[67] Therefore, in one sense, what was sought to be copyrighted was what the law viewed as a useful article, that is, a lamp.[68] However, when viewed as a work of art, the dancer that was part of the lamp appeared to meet the usual indicia of copyrightability.[69] Ultimately, the Court upheld the copyrightability of works of art that had been incorporated into useful articles, so long as certain conditions are met.[70]

After *Mazer*, the Copyright Office addressed this issue with detailed regulations, that have largely been codified into the 1976 Copyright Act (hereinafter, 'the Act' or the ''76 Act').[71] Relevant sections include the definitions of "pictorial, graphic, and sculptural works" and "useful article" in section 101, and section 113, which provides the scope of exclusive rights in pictorial, graphic, and sculptural works.[72] The Act appears to recognize protectable copyrightable expression in a useful article only if, and only to the extent that, such an article incorporates pictorial, graphic, or sculptural features that can be identified separately from, and are capable of existing independently of, the utilitarian aspects of the article.[73] Thus, these provisions clarify in a reasonably precise manner how the form-function doctrine may interact with copyrightable expression to affect the scope of protection with respect to traditional works subject to copyright, such as pictorial, graphic and sculptural works.[74] That is, the codification that followed *Mazer*, captures the form-function doctrine in connection with pictorial, graphical and sculptural works. Nonetheless, the doctrine that originated in *Baker* is not necessarily limited to the details of these particular provisions. More importantly, this codified approach is remarkably different than the approach employed regarding software protection.

Again, this latter point may at least partially be the source of confusion around software. That is, *Baker* is recognized primarily for the merger doctrine,[75] as discussed immediately below; whereas, issues around useful

---

66 Mazer v. Stein, 347 U.S. 201, 205 (1954).
67 *Id.* at 202.
68 *Id.* at 206.
69 *Id.* at 218.
70 *Id.* at 218.
71 17 U.S.C. § 101–1511 (2020); *see* § 101 (definition of "pictorial, graphical and structural works"; definition of "useful article"); § 113(c).
72 §§ 101, 113.
73 *See* § 101 (defining "pictorial, graphical and structural works").
74 *See* § 102 (including "pictorial, graphical and structural works").
75 Baker v. Selden, 101 U.S. 99, 104 (1880).

articles are generally analyzed under *Mazer* and its progeny.[76] Software is *not* typically comprehended in connection with useful articles.[77] Courts do not appear to appreciate the significance that software is excepted from a fundamental copyright law doctrine perhaps because useful articles are usually viewed as tangible items.

A careful review of these provisions of the Act reveals a dichotomy between copyrightable expression and rights related to use.[78] Where the utilitarian aspects of the work (e.g., a useful article) and its copyrightable elements cannot be sufficiently separated and are not sufficiently distinct, rights of the author related to such otherwise copyrightable expression fail, as exceeding the scope of the rights that were intended to be made available through copyright law.[79] A possible question to consider is whether for software, such a division between the copyrightable expression of program code (e.g., software) is able to be identified in a manner so that it is conceptually separable or conceptual distinguishable from the usefulness or the utility of that code.[80]

The Supreme Court case, *Baker v. Seldon* is also held to have established another important copyright law doctrine – "the doctrine of merger."[81]

---

76 Mazer v. Stein, 347 U.S. 201, 212–13 (1954).

77 *See* § 102 ("Copyright protection subsists . . . in original works of authorship fixed in any tangible medium of expression . . .").

78 *See* §§ 101, 113.

79 Clearly, software is not the only area in which the form-function doctrine may operate and raise challenging questions. Another area of concern is generally referred to as "applied art." *See* Kieselstein-Cord v. Accessories by Pearl, Inc., 632 F.2d 989, 993 (2d Cir. 1980) (finding that the conceptually separable artistic elements of belt buckles are copyrightable); Brandir Int'l, Inc. v. Cascade Pac. Lumber Co., 834 F.2d 1142, 1147 (2d Cir. 1987) (ribbon design for bicycle rack held unprotectible by copyright law as useful article lacking separable expression). *See, e.g.*, Star Athletica, LLC v. Varsity Brands, Inc., 137 S. Ct. 1002, 1016 (2017); Jane C. Ginsburg, *Courts Have Twisted Themselves into Knots: US Copyright Protection for Applied Art*, 40 COLUM. J. L. & ARTS 1, 6 (2016); Christopher Buccafusco & Jeanne C. Fromer, *Fashion's Function in Intellectual Property Law*, 93 NOTRE DAME L. REV. 51, 83 (2017).

80 *See infra* notes 112–13 and accompanying text.

81 It does not appear to be universally recognized that the merger doctrine and the form-function doctrine (sometimes referred to as the useful article doctrine) are separate doctrines. *Compare* McKenna, *supra* note 64, at 531, 533, 535 (distinguishing between the idea-expression dichotomy and the useful articles doctrine), *with* Kevin J. Hickey, *Reframing Similarity Analysis in Copyright*, 93 WASH. U. L. REV. 681, 696 (2016) (citing Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 708–10 (2d Cir. 1992)) (treating merger and functionality as essentially the same doctrine). For example, *Nimmer* instead characterizes *Baker v. Selden* as drawing a distinction between copying for use and copying for explanation – referred to as the "ends/means" distinction. NIMMER ON COPYRIGHT § 2A.03, *supra* note 1; 1 MELVILLE B. NIMMER, & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.04 (2021).

Although the ledgers of that case[82] otherwise met the appropriate qualifications for copyright protection (e.g., original, fixed in a tangible medium of expression, etc.), the Supreme Court determined that the ledgers were not eligible for copyright protection.[83]

According to the Court in *Baker*, accounting ledgers that were included in a book on accounting and asserted to have been copied themselves were a necessary element of an expression of the idea underlying the accounting ledgers.[84] That is, the ledgers were necessary to any expression of the underlying idea.[85] The idea and expression in this case had "merged" or were sufficiently close so that to entitle the author to copyright protection over the expression would provide control over the idea.[86] Since ideas are free and unprotectable, where the idea and the expression merge, copyright protection is not available.[87]

To be even more specific, the text of the book is copyrightable and within the scope of protection of copyright.[88] The ledgers may be viewed as uncopyrightable or as not within the scope of protection of copyright.[89] This latter distinction between copyrightability and scope of protection in some cases may be one without a difference, such as where the entire work might be subject to merger.[90] Examples might be titles or short phrases, which are generally not considered to be copyrightable,[91] except in unusual circumstances.[92] However, for software, as a contrast, to have merger apply to an entire program of code that is large and complex seems to be an unlikely scenario.[93]

---

82 As discussed earlier in this article, the case involved a book on accounting that included ledgers. The defendant had copied the ledgers from the book and began selling the copies and the author of the book sued based on copyright infringement.

83 Baker v. Selden, 101 U.S. 99, 107 (1880).

84 *See id.* at 103.

85 *Id.* at 103–04.

86 *Id.*

87 *Id.* at 105.

88 *Id.* at 101–02.

89 *Id.* at 102.

90 *Id.* at 101–02.

91 *See, e.g.*, U.S. COPYRIGHT OFFICE, CIRCULAR NO. 33: WORKS NOT PROTECTED BY COPYRIGHT 2 (last revised March 2021) ("Words and short phrases, such as names, titles, and slogans, are uncopyrightable because they contain an insufficient amount of authorship. The Office will not register individual words or brief combinations of words, even if the word or short phrase is novel, distinctive, or lends itself to a play on words."); Ferman v. Jenlis, Inc., 224 F. Supp. 3d 791, 802 (S.D. Iowa 2016) (stating there is no protection for "No Trespassing" sign showing surveillance camera).

92 Mary Minow, *Copyright Protection for Short Phrases – Rich Stim*, STAN. LIBRARIES (Sept. 9, 2003), https://fairuse.stanford.edu/2003/09/09/copyright_protection_for_short/.

93 *See* 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.10(B)(2) (2021) (stating, in connection with merger, that "a given routine or component of the

Likewise, in reviewing these basic copyright law principles, some conceptual distinctions are helpful to aid the later discussion. First, there is a distinction between determining whether a work is subject to protection, i.e., copyrightable, and determining the scope of protection available for the work, assuming that it is subject to protection.[94] Some take the view that this follows from the statute itself in which section 102(a)[95] clarifies what is copyrightable and section 102(b)[96] clarifies the scope of protection. However, regardless of whether one subscribes to this statutory interpretation, these are two separate legal questions and courts sometimes confuse them or disagree about which question is the appropriate question to address.[97]

If a work is copyrightable, to make out a case of infringement, in addition to proving access to the work by the defendant, a plaintiff must prove that the copyrightable work and the accused work are substantially similar.[98]

---

software may properly fall within the scope of merger."); *See infra* notes 318–46 and accompanying text.

94 *See* 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.05(A)(2)(b) (2021) ("This approach treats the merger principle as one relating to the boundaries of permissible copying, rather than solely as a rule of copyrightability."); NIMMER ON COPYRIGHT § 2A.10(B)(4), *supra* note 93 (describing the view that merger is a defense to infringement as "[t]he better view."); NIMMER ON COPYRIGHT § 13.03(B)(3)(e), *supra* note 6 ("Confusion has arisen in the case law whether the merger doctrine should serve as a bar to copyright protection itself (element one) or, alternatively, as a negation of infringement via absence of actionable similarity (element two).").

95 17 U.S.C. § 102(a) (2021) ("Copyright protection subsists, in accordance with this title, in original works of authorship fixed in any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device.").

96 § 102(b) (2020) ("In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.").

97 *See* NIMMER ON COPYRIGHT § 2A.05(A)(2)(b), *supra* note 94 ("This approach treats the merger principle as one relating to the boundaries of permissible copying, rather than solely as a rule of copyrightability."); NIMMER ON COPYRIGHT § 2A.10(B)(4), *supra* note 93 (describing the view that merger is a defense to infringement as "[t]he better view."); Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1358 (Fed. Cir. 2014) ("In the Ninth Circuit, while questions regarding originality are considered questions of copyrightability, concepts of merger and scenes a faire are affirmative defenses to claims of infringement."). *But see* Sandra Ocasio, *Pruning Paracopyright Protections: Why Courts Should Apply the Merger and Scènes à Faire Doctrines at the Copyrightability Stage of the Copyright Infringement Analysis*, 3 SETON HALL CIR. REV. 303, 304 (2006); Pamela Samuelson, *The Story of Baker v. Selden: Sharpening the Distinction between Authorship and Invention*, *in* INTELLECTUAL PROPERTY STORIES 159, 192 (Jane Ginsburg & Rochelle Dreyfuss, eds., 2005).

98 *See, e.g.*, Soc'y of the Holy Transfiguration Monastery, Inc. v. Gregory, 689 F.3d 29, 49 (1st Cir. 2012), *cert. denied*, 568 U.S. 1167 (2013); Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1231–32 (3d Cir. 1986); Alan Latman, *"Probative*

This is how the scope of protection for a copyrightable work comes into play. Substantial similarity is a legal term used by courts to describe an amount of copying that is qualitatively and quantitatively sufficient for the court to conclude the defendant wrongfully appropriated the plaintiff's protected expression from the plaintiff's copyrightable work.[99] In some cases, copying may be literal copying (e.g., copying verbatim); while in other cases, although literal copying has not occurred, enough so called non-literal copying may have taken place so that the works are considered substantially similar.[100] Thus, substantial similarity refers to having copied a quantum of expression from a work to be legally liable, which is a legal analysis performed once some amount of factual copying has been shown to have taken place.[101]

In the software arena, as in essentially all other areas of copyright, what constitutes the non-literal scope of a copyrightable work is not always clear. A reason for this relates to the so-called levels of abstraction and the role of a court to seek a balance between competition and protection.[102] The protectible non-literal scope of a work lies somewhere between the literal work itself, which is protectible from literal copying, and the unprotectible ideas of the work.

However, some additional confusion regarding non-literal scope potentially may exist for software works than for other types of copyrightable works. For example, in an important 2014 Federal Circuit decision, *Oracle America, Inc. v. Google, Inc.* the Federal Circuit stated: "The non-literal components of a computer program include, among other things, the program's sequence, structure, and organization, *as well as the program's user interface*."[103] This case, discussed in more detail, *infra*., is a

---

Similarity" As Proof of Copying: Toward Dispelling Some Myths in Copyright Infringement, 90 COLUM. L. REV. 1187, 1188 (1990).

99 Shyamkrishna Balganesh et al., *Judging Similarity*, 100 IOWA L. REV. 267, 268 (2014).

100 Richard Raysman & Peter Brown, *Copyright Infringement of Computer Software and the 'Altai' Test*, 235 N.Y. L. J. 1, 1 (May 9, 2006), http://euro.ecom.cmu .edu/program/law/08-732/Copyright/CopyrightInfringementOfSoftware.pdf.

101 *See Whelan Assocs., Inc.*, 797 F.2d at 1232; Latman, *supra* note 98, at 1190, 1198; Pamela Samuelson, *A Fresh Look at Tests for Nonliteral Copyright Infringement*, 107 NW. U. L. REV. 1821, 1840 (2013); Hickey, *supra* note 81, at 690; Gabriel Godoy-Dalmau, *Substantial Similarity:* Kohus *Got it Right*, 6 MICH. BUS. L. REV. 231, 232 (2017); Christopher Jon Sprigman & Samantha Fink Hedrick, *The Filtration Problem in Copyright's "Substantial Similarity" Infringement Test*, 23 LEWIS & CLARK L. REV. 571, 576–77 (2019); NIMMER ON COPYRIGHT § 13.03(F)(1)(a), *supra* note 6.

102 *See* Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971); *see supra* notes 31–39 and accompanying text.

103 Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1355–56 (Fed. Cir. 2014) (citing

leading case on the question of the copyrightability and scope of protection for software under copyright law.[104] However, such a statement is not without controversy. It is true that execution of software may create a graphical user interface ("GUI"); however, the conventional view is that the GUI is a separate work from the software, separately registerable and separately protectible.[105] For example, in the Second Circuit case, *Altai*, cited previously and discussed at length *infra*., that court goes out of its way to distinguish generated screen displays from the software.[106] The Second Circuit specifically states: "As a caveat, we note that our decision here does not control infringement actions regarding categorically distinct works, such as certain types of screen displays."[107] Likewise, in the First Circuit decision, *Lotus Development Corp. v. Borland International, Inc.,* (hereinafter *Lotus* or *Lotus v. Borland)* also later discussed in some detail, the First Circuit makes a similar qualification, stating in no uncertain terms: "In the instant appeal, we are not confronted with alleged nonliteral copying of computer code."[108] The Third Circuit also recognizes this distinction, such as in the case *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*, (hereinafter *Whelan* or *Whelan v. Jaslow)* discussed in more detail *infra*.,[109] not simply between software and a screen display or user interface, but also between software and the file structures for

---

Johnson Controls, Inc. v. Phoenix Control Sys., Inc., 886 F.2d 1173, 1175 (9th Cir. 1980)) (emphasis added).

104 After this decision, there was another Federal Circuit decision involving this dispute in 2018. Ultimately, the Supreme Court granted *certiorari* to both cases, discussed in more detail, *infra* notes 409–16 and accompanying text.

105 *See* Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 703 (2d Cir. 1992); Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 814–15 (1st Cir. 1995). *See infra* notes 111, 501–502 and accompanying text. *See* discussion, *infra*, Section III.A.

106 *Comput. Assocs. Int'l., Inc*., 982 F.2d at 703.

107 *Id.*

108 *See Lotus Dev. Corp*., 49 F.3d at 814.

109 Interestingly, the *Whelan* court recognizes the possibility that while these other types of works have separate copyrights, still any similarity might be *indirect* evidence of copying as to the software code, stating, for example:

It is true that screen outputs are considered audio-visual works under the copyright code . . . and are thus covered by a different copyright than are programs, which are literary works . . . It is also true that Whelan Associates asserts no claim of copyright infringement with respect to the screen outputs. But the conclusion to be drawn from this is not, as defendants would have it, that screen outputs are completely irrelevant to the question whether the copyright in the program has been infringed. Rather, the only conclusion to be drawn from the fact of the different copyrights is that the screen output cannot be *direct* evidence of copyright infringement. There is no reason, however, why material falling under one copyright category could not be indirect, inferential evidence of the nature of material covered by another copyright.

Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1244 (3d Cir. 1986).

that software.[110]

Conventionally, a GUI is not typically considered to be within the non-literal scope of protection for the software. However, with that said, a GUI might be considered "software-like" (e.g., sufficiently hybrid-like in which expressive and useful elements intersect) that it should, perhaps, fall within the regime that applies to software, the regime we refer to here as hybrid intellectual property rights. A question, however, may be whether Congressional action is required to effectuate such a result, in light of the present language of the Act.[111]

Finally, another distinction that is often confused, as previously alluded, is the difference between the doctrine of merger and the form-function doctrine.[112] Confusion exists most likely because both doctrines originated in *Baker v. Selden*, mentioned previously. Part of the confusion may be because the ledgers at issue in the case were an example of both merger of idea and expression, since there are only a few ways to express the ledgers, *and* an example of expression that is useful or utilitarian.[113] However, it is believed that *Mazer v. Stein*, rather than *Baker v. Selden*, more clearly gave *independent* life, so to speak, to the form-function doctrine *apart from* the principle of merger.[114]

Yet another basis for confusion in this area may be related to the various interpretations that have been given to 17 U.S.C. section 102(b) as possibly a codification of one or more of these doctrines.[115] There does not appear to be uniform agreement on the interpretation of section 102(b).[116] More

---

110 *See id.* at 1242–43.

111 *See infra* notes 361–67 and accompanying text.

112 *Compare* McKenna, *supra* note 64, at 530–39 (distinguishing between the idea-expression dichotomy and the useful articles doctrine), *with* Hickey, *supra* note 81, at 696 (2016) (citing Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 708–10 (2nd Cir. 1992)) (treating merger and functionality as essentially the same doctrine). For example, Nimmer & Nimmer instead characterizes *Baker v. Selden* as drawing a distinction between copying for use and copying for explanation – referred to as the "means/ends" distinction. *See* NIMMER ON COPYRIGHT § 2A.03, *supra* note 1; NIMMER ON COPYRIGHT § 2A.04, *supra* note 81. *See supra* note 81 and accompany text (explaining this distinction may not be universally recognized). *See supra* text accompanying note 58.

113 *See* Baker v. Selden, 101 U.S. 99, 99–100 (1880).

114 *See supra* notes 66–77 and accompanying text. *Cf.* Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1252 (3d Cir. 1983) (citing Mazer v. Stein, 347 U.S. 201, 218 (1954)) ("We find nothing in the copyright statute to support the argument that the intended use or use in industry of an article eligible for copyright bars or invalidates its registration. We do not read such a limitation into the copyright law.").

115 17 U.S.C. §102(b) (2021); *see Apple Comput., Inc*., 714 F.2d at 1252; *Whelan Assocs., Inc*., 797 F.2d at 1242–43, 1243 n.41; Oracle Am., Inc., v. Google, Inc., 750 F.3d 1339, 1354–55 (Fed. Cir. 2014); 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.06 (2021).

116 For example, it is notable that *Nimmer* changed its view on this question around

traditionally, several courts view section 102(b) as primarily a codification of the idea/expression dichotomy[117] and that is largely the approach taken in this article. For example, the House Report of the '76 Act states:

> Section 102(b) in no way enlarges or contracts the scope of copyright protection under the present law. Its purpose is to restate, in the context of the new single Federal system of copyright, that the basic dichotomy between expression and idea remains unchanged. [118]

However, it also states:

> Copyright does not preclude others from using the ideas or information revealed by the author's work. It pertains to the literary musical, graphic, or artistic form in which the author expressed intellectual concepts. Section 102(b) makes clear that copyright protection does not extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.
>
> Some concern has been expressed lest copyright in computer programs should extend protection to the methodology or processes adopted by the programmer, rather than merely to the 'writing' expressing his ideas. Section 102(b) is intended, among other things, to make clear that the expression adopted by the programmer is the copyrightable element in a computer program, and that the actual processes or methods embodied in the program are not within the scope of the copyright law. [119]

Section 102(b) expressly states that, "in no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery" much like the idea/expression dichotomy.[120] Regardless, however, the two doctrines, merger

---

2016. *See* NIMMER ON COPYRIGHT § 2A.04, *supra* note 81; NIMMER ON COPYRIGHT § 2A.06, *supra* note 115. The current view in Nimmer appears consistent with this article. *See* NIMMER ON COPYRIGHT § 2A.04(A)(2), *supra* note 81 ("[T]he revised discussion emphasizes that Section 102(b) of the current Act independently bars copyright protection not only for ideas but also for methods, procedures, and other enumerated categories.").

117 *See Apple Comput., Inc*., 714 F.2d at 1252–53; *Whelan Assocs., Inc*., 797 F.2d at, 1234, 1237, 1243 n.41; *Oracle Am., Inc*., 750 F.3d at 1367. B*ut see Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1237 (discussing "the proper role of § 102(b) in computer program copyright cases"); NIMMER ON COPYRIGHT § 2A.06, *supra* note 115.

118 H.R. REP. NO. 94-1476, at 57 (1976).

119 *Id.* at 56–57.

120 § 102(b). An alternate view of section 102(b) is that it is a codification of the merger doctrine. *See* Toro Co. v. R & R Prod. Co., 787 F.2d 1208, 1212 (8th Cir. 1986) ("Appellee urges that we uphold the court's ruling by assuming that under the rubric of § 102(b) the

and form-function, while seemingly close in some respects, and originating from one set of facts in *Baker v. Selden*, address entirely different policy concerns.[121]

The merger doctrine addresses the concern that arises in situations where there are a limited number of ways to express an idea.[122] In such a situation, but for the merger doctrine, an author of expression that merges with its underlying idea would, in effect, have control over the idea; however, ideas are meant to be free and part of the public domain.[123] Thus, the merger doctrine operates to keep such ideas freely available to others.[124]

In contrast, the form-function doctrine addresses the concern that arises in a situation in which functionality and expressive content overlap or coincide.[125] In this situation, control over expression would provide rights over use, as in the case of a useful article whose form (e.g., shape, appearance, etc.) contains expressive content; however, patent law, rather than copyright law, with its much higher standards of patentability, is the appropriate domain for such subject matter.[126] The Act provides some helpful guidance on conditions under which expressive content does not run afoul of this doctrine for "pictorial, graphic and sculptural works," such as if the expressive elements of a useful article "can be identified separately from, and are capable of existing independently of, the utilitarian aspects" of the article.[127] However, more generally, it appears that, for a given work

---

court actually was applying the doctrine of merger."); *cf.* NIMMER ON COPYRIGHT § 2A.06, *supra* note 115 ("[T]he phrase cannot be taken to suggest that, merely because a work embodies subject matter that is excluded from protection in some way or form, it follows that the work as a whole is categorically to be denied protection."). In many ways, it appears to be a distinction without a difference which principal section 102(b) is interpreted to codify since both doctrines exist due to case law precedent and the statute itself does not suggest otherwise. However, in addition to the excerpts provided from legislative history, which support the view taken in this article, a reason for preferring the interpretation provided here is that the language of section 102(b) has some imprecision about it. *See, e.g.*, NIMMER ON COPYRIGHT § 2A.06, *supra* note 115 (stating that there is "a good degree of overlap" in the enumerated categories and that Congress gave no direction as to their meaning or scope). Thus, rather than refer to the language of 102(b) as a source of law for doctrines that clearly are established by case precedent, such as merger and/or form-function, and invite a consequential linguistic dissection of section 102(b), it is more intellectually appealing to see it as *largely* a codification of the idea/expression dichotomy.

121 *See generally* Baker v. Selden, 101 U.S. 99, 99–107 (1880).

122 *See, e.g.*, *Apple Comput., Inc.*, 714 F.2d at 1253; CCC Info. Servs., Inc. v. Maclean Hunter Market Reports Inc., 44 F.3d 61, 69 (2d Cir. 1994); Herbert Rosenthal Jewelry Corp., v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971).

123 *CCC Info. Servs., Inc.*, 44 F.3d at 68.

124 NIMMER ON COPYRIGHT § 13.03, *supra* note 6.

125 *See* McKenna, *supra* note 64, at 535 (referring to the useful articles doctrine).

126 *See Baker*, 101 U.S. at 102–03.

127 17 U.S.C. § 101 (2020).

containing copyrightable, expressive elements, where those expressive elements cannot be identified separately and are not capable of existing independently of utilitarian or useful features of the work, then those elements, though expressive, receive no protection under copyright law.

B. The 1976 Copyright Act and the 1980 Amendments

Issues involving the copyrightability of computer software preceded the passage of the 1976 Copyright Act (again, 'the Act' or the ''76 Act').[128] As computers developed and evolved, the Copyright Office was faced with the question in the '60s of whether to permit registration of these works.[129] In 1964, the Office determined to register such works under its "rule of doubt," but required deposit of a human-readable form of the program if the program had been published in only machine-readable form.[130] Of course, this did not really resolve any important questions about copyright law and software, but it did provide an easy to implement administrative approach to an otherwise thorny legal question.[131]

An important and unresolved question, for example, was whether using copyrighted materials in computers amounted to copyright infringement.[132] Therefore, Congress established a National Commission on New Technological Uses of Copyrighted Works to study the problem.[133] This has become known as "CONTU." Likewise, to prevent this issue from holding up passage of the '76 Act, a provision was included in the bill, referred to as old section 117.[134] This section carried forward the law in effect on

---

128 *See* 17 U.S.C. § 101–1511 (2020); Ralph Oman, *Computer Software as Copyrightable Subject Matter: Oracle v. Google, Legislative Intent, and the Scope of Rights in Digital Works*, 31 HARV. J. L. & TECH. 639, 639 (2018).

129 Jule L. Sigall, *Copyright Infringement Was Never this Easy: Ram Copies and Their Impact on the Scope of Copyright Protection for Computer Programs*, 45 CATH. U. L. REV. 181, 186 (1995).

130 *See* U.S. COPYRIGHT OFFICE, CIRCULAR NO. 61: COPYRIGHT REGISTRATION OF COMPUTER PROGRAMS 6 (March 2021) ("The Copyright Office strongly prefers that you submit your copyright application using a source code deposit. You can submit a deposit using the object code of the computer program; however, your claim will be subject to the Copyright Office's Rule of Doubt. The rule notifies interested parties that although the Office has accepted the claim submitted, it is unwilling to grant a presumption of validity to certain aspects of the claim. For more information about the Rule of Doubt, see chapter 600, section 607, of the *Compendium*.").

131 *See id.*

132 H.R. REP. NO. 94-1476, at 48 (1976).

133 Richard H. Stern, *Section 117 of the Copyright Act: Charter of the Software Users' Rights or an Illusory Promise?*, 7 W. NEW ENG. L. REV. 459, 459 (1985).

134 Sigall, *supra* note 129, at 187–88; *see* 17 U.S.C. § 117 (2021).

December 31, 1977[135], as to rights with respect to computer usage of copyrighted works.[136] Most commentators would likely say that the '76 Act intended that computer programs be protected under section 102 of the new Act, but a mostly academic question remains as to what rights did authors of such works necessarily receive?

It is well-known today what then transpired. The Commission recommended that old section 117 be repealed and replaced with a new section 117.[137] Furthermore, the Commission recommended that section 101 be amended to include a definition of "computer program."[138] That definition states: "A 'computer program' is a set of statements or instructions *to be used* directly or indirectly *in a computer* in order to bring about a certain result."[139] In 1980, Congress amended the Copyright Act and implemented CONTU's recommendations in their entirety, except that Congress used the word "owner" in place of "rightful possessor" in the recommended version of new section 117.[140] This sequence of events created a basis for believing that the CONTU report reflects the intent of Congress.[141]

Before launching into how courts have dealt with these significant statutory changes, a few points that might be lost on the casual observer are worth discussing. First, although the 1980 amendments made it clear that software was copyrightable, even if one did not agree that it was already clear under section 102 of the '76 Act, the statute did not clarify the appropriate category of copyrightable works in which to place computer programs.[142] This issue has some importance in that frequently the category affects the rights available under the statute.[143]

---

135 The '76 Act became effective on January 1, 1978. H.R. REP. NO. 94-1476, at 19 (noting, the '76 Act became effective on January 1, 1978).

136 *See* Stern, *supra* note 133, at 460 n.7.

137 *See* Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1248 (3d Cir. 1983); *see* Stern, *supra* note 133, at 460 n.7; *Nat'l Comm'n on New Tech. Uses of Copyrighted Works (CONTU), Final Rep. on the Nat'l Comm'n on New Tech. Uses of Copyrighted Works*, 3 COMPUT. L.J. 53 (1981).

138 *See Apple Comput., Inc*., 714 F.2d at 1247–48; *See* Stern, supra note 133, at 460 n.7; *Nat'l Comm'n on New Tech. Uses of Copyrighted Works (CONTU), supra* note 137.

139 17 U.S.C. § 101 (2020) (emphasis added).

140 17 U.S.C. § 117(a) (2021).

141 *See, e.g.*, *Apple Comput., Inc*., 714 F.2d at 1247; Midway Mfg. Co. v. Strohon, 564 F. Supp. 741, 750 n.6 (N.D. Ill. 1983).

142 17 U.S.C. § 102 (2020).

143 *See* 17 U.S.C. § 106 (2020):

Subject to sections 107 through 122, the owner of copyright under this title has the exclusive rights to do and to authorize any of the following:
(1) to reproduce the copyrighted work in copies or phonorecords;
(2) to prepare derivative works based upon the copyrighted work;
(3) to distribute copies or phonorecords of the copyrighted work to the public by sale or other transfer of ownership, or by rental, lease, or lending;

The legislative history of the '76 Act implies that computer programs qualify as literary works and several cases have followed that approach.[144] This is a helpful but imperfect model of how computer programs fit within the categories of protectible works enumerated by the Act.

Another unusual and important aspect of this attempt to bring computer programs within the regime of copyright law is the precise text of new section 117, at least considering traditional copyright law doctrines.[145] Specifically, the text of section 117 contemplates use in connection with a computer program by expressly referring to "utilization."[146] The statute expressly states it is not an infringement to copy or make an adaptation of a computer program provided that "such new copy or adaption is *created as an essential step in the utilization of the computer program* in conjunction with a machine . . . "[147] This provision likewise necessarily lines up with the definition of a computer program, previously cited, which states: "A 'computer program' is a set of statements or instructions *to be used* directly or indirectly *in a computer* in order *to bring about a certain result*."[148]

These provisions, particularly the new section 117, are significant and unique for copyright law.[149] These provisions recognize something previously thought outside the scope of copyright law.[150] The general principle is that if use rights somehow become entangled with copying rights, the right to copy gives way to the right to use, as a matter of legal

---

(4) in the case of literary, musical, dramatic, and choreographic works, pantomimes, and motion pictures and other audiovisual works, to perform the copyrighted work publicly;
(5) in the case of literary, musical, dramatic, and choreographic works, pantomimes, and pictorial, graphic, or sculptural works, including the individual images of a motion picture or other audiovisual work, to display the copyrighted work publicly; and
(6) in the case of sound recordings, to perform the copyrighted work publicly by means of a digital audio transmission.

144 *See Apple Comput., Inc*., 714 F.2d at 1247; Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1240 (3d Cir. 1986).

145 *See supra* notes 137–41 and accompanying text.

146 17 U.S.C. § 117(a)(1) (2021).

147 § 117(a) (emphasis added) ("Making of Additional Copy or Adaptation by Owner of Copy.— Notwithstanding the provisions of section 106, it is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided: (1) that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or (2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.").

148 17 U.S.C. § 101 (2020) (emphasis added).

149 *See* § 117.

150 *See id.*

policy.[151] That is, copyright protection is not provided in that situation. Here, however, the right to copy remains despite its intersection with the right to use.

It is almost an understatement to observe that for software the traditional approach is not the current approach of the law. While it appears that for computer programs the use rights and the copyrightable expression are entangled in a complex and inseparable way, this does not render the expression uncopyrightable. Recall the Supreme Court's statement in *Baker* that "[t]he use of the art is a totally different thing . . ."[152] Yet, here, the statute, by its very language, recognizes that copying (or modifying) a program may, in fact, be essential to use of the program, and, yet, this does not denigrate the copyrightability of the program, specifically per Congressional intent.[153] For example, in *Baker*, copying the ledgers was essential to their use.[154] To be even more pointed, current section 117 states that it is not an infringement to copy or modify the program if doing so is essential to using the program.[155] This provision, therefore, presupposes that doing so would otherwise be an infringement; however, where doing so is essential to using the program, then there is no infringement for this particular situation.[156] Instead, with respect to this particular type of copyrightable work, section 117 provides leeway to copy the program, and even modify it, without traditional infringement liability attaching regarding such acts.[157]

This statutory language is implicit, if not explicit, support to show that Congress intended for computer programs to be copyrightable without the condition that such programs are copyrightable only "if, and only to the extent that" the copyrightable elements "can be identified separately from, and are capable of existing independently of, the utilitarian aspects."[158] Congress clearly intended this result, based on the language of sections 101 and 117 (as well as legislative history[159]), realizing that computer programs

---

151 *See supra* notes 54–70 and accompanying text.

152 *See* Baker v. Selden, 101 U.S. 99, 104 (1880).

153 § 117(a)(1).

154 *Baker*, 101 U.S. at 103.

155 § 117(a)(1).

156 *See id.*

157 *See* Lateef Mtima, *So Dark the CON(TU) of Man: The Quest for a Software Derivative Work Right in Section 117*, 70 U. PITT. L. REV. 1, 26–27 (2007); Stern, *supra* note 133, at 463.

158 17 U.S.C. § 101 (2021).

159 *See, e.g.*, Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1252–53 (3d Cir. 1983); Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 818 (1st Cir. 1995); Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1367 (Fed. Cir. 2014).

are used and, therefore, are useful.[160]

The history of this provision and its language also make it clear that its scope is limited to software.[161] Nimmer refers to section 117 as providing a use privilege.[162] The section specifically immunizes copying that takes place "as an essential step" in the use of the computer program.[163] A legal question arises as to the scope of the language "essential step." Regardless, however, this provides Congressional recognition that it is necessary to copy software to use it.[164]

This statutory recognition makes software unique from other forms of copyrightable subject matter. As stated previously, it provides, either expressly or implicitly, an exemption for software from the form-function doctrine by making it clear that software, in general, cannot be used without being copied.[165] Notably, section 117 only applies to the owner of a copy of a computer program, and not to the owner of the copyright for the program.[166] This is consistent with the interpretation advanced here, that the

---

160 *See Apple Comput., Inc*., 714 F.2d at 1248 n.6 ("[t]he parties agree that this section is not implicated in the instant lawsuit. The language of the provision, however, by carving out an exception to the normal proscriptions against copying, clearly indicates that programs are copyrightable and are otherwise afforded copyright protection."); *id.* (citing H.R. REP. NO. 1476, at 116, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5731) ("Section 117 applied only to the scope of protection to be accorded copyrighted works when used in conjunction with a computer and not to the copyrightability of programs.").

161 *See* 2 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 8.08(B)(1) (2021) ("The current exemption applies solely to software.").

162 *See id.* ("It comes into being by according a privilege to copy software *when the owner of a copy in which the relevant program is embodied copies the software as an essential step in utilizing the program*. The various criteria are canvassed in the subsections below.") (emphasis added).

163 17 U.S.C. § 117(a)(1)(2021).

164 That you copy a computer program to use it is simply a technological fact. In general, the term "software" distinguishes computer code from "hardware." The term "firmware" is also used and firmware has been held to be copyrightable. The distinction for firmware is that it is loaded, like software, but then "burned in" the device, making it more like hardware due to its permanency. In a typical computer system, software, is stored within relatively long-term memory and then, to be executed, the software is loaded from storage in long term memory into storage in short term (and faster) memory that is capable of being accessed by digital circuitry. The digital circuitry, such as a central processing unit ("CPU") or a microprocessor, is then able to execute the recently loaded software instructions. The faster, shorter-term memory may be part of the CPU or microprocessor, or it may be separate from the CPU or microprocessor. It is this loading into memory for execution, however, that constitutes copying so that the software can be used.

165 NIMMER ON COPYRIGHT § 8.08(B)(1), *supra* note 161 ("The current exemption applies solely to software.").

166 *See id*. § 8.08(B)(1)(a); *id.* § 8.8(B)(1)(c)(i) (citing Krause v. Titleserv, Inc., 402 F.3d 119 (2d Cir. 2005)) (determining who qualifies as the owner of the copy, did not consider

owner of a copy would be unable to use the computer program with a legal restriction on copying, since to use the computer program requires that the computer program be "copied" into memory.

Likewise, assuming the foregoing interpretation is correct, it is also logical to conclude that to the extent section 102(b) might perhaps be implicated, then that provision is also implicitly amended to be consistent.[167] That is, without an implicit carve out for software, as we shall see, courts have the potential to vitiate the purpose of the 1980 amendments depending on how one might construe that provision.[168] However, this article interprets section 102(b) in a manner so such a carve out is superfluous.[169]

It may be worthwhile to consider how a computer program, as defined in the Act after the 1980 amendments, might be implicated by the definition of "useful article."[170] It could at least be argued that a computer program, once fixed in a tangible medium of expression, such as a semiconductor chip or another type of digital storage device, which would typically be as object code, rather than as source code, meets the definition of a useful article.[171] While the program itself may be separated from the article, it is not true that the utility of the code can be separated from the expression within the code, conceptually or otherwise.[172] Yet, as seems clear, copyrightable expression integrated with or within a useful article is entitled to no rights or protection *other than for computer programs* unless the copyrightable elements are capable of being identified separately from and capable of existing independently of the utilitarian aspects.[173]

Consequently, it appears that the copyright statute at least implicitly creates two regimes of protection.[174] One regime is the traditional regime of *Baker v. Selden* and *Mazer v. Stein*, which has been at least partially codified.[175] The other regime, however, is the regime regarding computer

---

title under state law and, instead, considered factors, such as who had the right to possess and destroy the copies, etc., and calling it "the better view.").

167 *See* 17 U.S.C. § 102(b) (2021).

168 *See generally* Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974 (N.D. Cal. 2012) (discussing definitions of terms used in Section 117 provisions and the adoption of those provisions in 1980); *Infra* text accompanying note 191–195. *See also supra* text accompanying notes 82–83 (discussing the house report associated with the 1980 amendments.)

169 *See supra* notes 115–20 and accompanying text; *infra* notes 355–70 and accompanying text.

170 *See* 17 U.S.C. § 101 (2021).

171 *See* §§ 101–102.

172 *See* § 101.

173 *See id.*

174 *See* 17 U.S.C. §§ 101–1401 (2021).

175 *See* § 102(b); *see generally* Baker v. Selden, 101 U.S. 99 (1879) (discussing the traditional form-function regime); Mazer v. Stein, 347 U.S. 201 (1954) (discussing the

programs, which are useful and yet are also subject to protection despite lack of compliance with the now codified test of *Mazer* or the form-function doctrine articulated by *Baker*.[176] It may very well be the case that CONTU and Congress did not *fully* comprehend how the changes made to the copyright statute in 1980 would strike at some fundamental copyright law principles and ultimately produce a shift in how courts deciding copyright law matters related to software approach the question of scope of protection.

C. Court Decisions Dealing with the Statutory Changes

An early case that recognized complex issues regarding the copyrightability and copyright infringement of software is *Apple Computer, Inc. v. Franklin Computer Corp.*.[177] (hereinafter *Apple* or *Apple v. Franklin*) In the case, the evidence was pretty clear that Franklin had copied Apple's operating system;[178] however, despite this, the district court denied granting a preliminary injunction to Apple, largely because the district court was not entirely sure that "an operating system program in 'binary code or one represented either in a ROM or by micro-switches' was an 'expression,' which could be copyrighted as distinguished from an 'idea,' which could not be."[179]

Franklin advanced a number of arguments to support the district court's decision, including that an operating system (OS) is not copyrightable essentially because the code is written strictly for computers not for humans.[180] Franklin also argued that object code is not copyrightable, that code embedded in read-only memory (ROM) is not copyrightable, and that, regardless, the OS needed to copy the code for compatibility reasons. [181]

The Third Circuit rejected the first argument that object code is not copyrightable.[182] This argument was laid to rest by an earlier Third Circuit decision, *Williams Electronics, Inc. v. Artic Int'l.* [183] The Third Circuit in *Apple* traced the history of the statutory changes around the 1980 amendments, which supports the notion that computer programs need not

---

traditional form-function regime and concept of separability).

176 *See Mazer*, 347 U.S. at 218; *see Baker*, 101 U.S. at 99–101, 103–04.

177 *See* Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1242, 1246–50 (3d Cir. 1983).

178 *See id.* at 1242–45.

179 *See id.* at 1246 (quoting Apple Comput., Inc. v. Franklin Comput. Corp., 545 F. Supp. 812, 821 (E.D. Pa. 1982)).

180 *See id.* at 1250–55.

181 *See id.*

182 *Id.* at 1246–47, 1249.

183 *Id.* at 1247; Williams Elecs., Inc. v. Artic Int'l, Inc., 685 F.2d 870, 877 (3d Cir. 1982).

necessarily be readable by a human to be copyrightable.[184]

The Third Circuit also rejected the argument that object code embedded in ROM is never copyrightable.[185] This was, in some sense, merely an argument about the useful or utilitarian nature of the ROM, similar to what was mentioned previously in connection with *Mazer*.[186] In other words, a ROM is a useful article. However, computer programs are copyrightable expressions by statute. Even where a program is not embedded in a ROM, the program is no more able to qualify as capable of being "identified separately from . . . the utilitarian aspects."[187] In other words, the form, being embedded in a ROM, appears to not be germane to the question of copyrightability since it does not change the fact that a program is unable to qualify under such a test.[188] In this regard, this early decision correctly handled an issue that appears to, on occasion, still confuse courts when, instead, addressing the scope of protection for software.

The next two arguments were not so easily dealt with, although the Third Circuit goes on to resolve them.[189] Franklin argued, in essence, that the operating system is more like a part or extension of a machine, i.e., a computer, and, therefore, is uncopyrightable.[190] In *Apple*, the Third Circuit also addressed arguments regarding whether OS qualified as a "process," "system," or "method of operation" under Section 102(b); this continues to draw scrutiny from courts today.[191] The Third Circuit stated:

> Franklin's attack on operating system programs as 'methods' or 'processes' seems inconsistent with its concession that application programs are an appropriate subject of copyright. Both types of programs instruct the computer to do something. Therefore, it should make no difference for purposes of section 102(b) whether these instructions tell the computer to help prepare an income tax return (the task of an application program) or to translate a high level language program from source code into its binary language object

---

184 *See Apple Comput., Inc*., 714 F.2d at 1247–48 ("[I]t is clear from the language of the 1976 Act and its legislative history that it was intended to obliterate distinctions engendered by White-Smith . . . . We reiterate the statement we made in Williams when we rejected that argument: 'the answer to the defendant's contention is in the words of the statute itself.'").

185 *See id.* at 1249 ("[W]e reaffirm that a computer program in object code embedded in a ROM chip is an appropriate subject of copyright.").

186 *See supra* notes 66–77 and accompanying text.

187 Varsity Brands, Inc. v. Star Athletica, LLC, 799 F.3d 468, 481–82 (6th Cir. 2015) (quoting 17 U.S.C. § 101 (2021)).

188 *See id.*

189 *See Apple Comput., Inc*., 714 F.2d at 1251 (finding CONTU more persuasive than *Franklin*).

190 *Id.*

191 *Id.* at 1250–51.

> code form (the task of an operating system program such as "Applesoft" . . . Since it is only the instructions which are protected, a "process" is no more involved because the instructions in an operating system program may be used to activate the operation of the computer than it would be if instructions were written in ordinary English in a manual which described the necessary steps to activate an intricate complicated machine. There is, therefore, no reason to afford any less copyright protection to the instructions in an operating system program than to the instructions in an application program. [192]

Furthermore, the Third Circuit also relied on CONTU, which indicated that programs are not considered machine parts, stating "we can consider the CONTU report as accepted by Congress since Congress wrote into the law the majority's recommendations almost verbatim."[193]

Franklin, thus, had argued that the purpose of the program is strictly utilitarian in that it runs a computer.[194] The appellate court correctly rejected Franklin's argument by relying on the statute itself.[195] As the appellate court points out, there is no distinction in the statute between application programs, which Franklin accepts as copyrightable, and operating system ("OS") programs, which Franklin does not accept as copyrightable.[196] Both meet the language of the statute and are, therefore, copyrightable.[197]

Franklin's argument here is similar in some ways to arguments about being object code and about being embedded in ROM, but perhaps, the argument could be read to raise a deeper question. Franklin argued, in effect, that the code was more utilitarian in terms of what it is designed to do than the typical computer program, not unlike the position recently taken by the US Supreme Court in *Google v. Oracle*, which relied instead on the fair use doctrine in that case.[198] Therefore, while an OS[199] is copyrightable by application of the words of the statute, this does not entirely address the role of utility, which conceivably might affect the scope of protection for computer programs.

To reformulate the issue to highlight its significance, let us consider: do computer programs contain expression? Clearly. Thus, under copyright law, expressive elements are usually worthy of copyright protection; however,

---

192 *Id.* at 1251.
193 *Id.* at 1252.
194 *Id.* at 1249, 1251.
195 *Id.* at 1248–49.
196 *Id.* at 1252.
197 *Id.* at 1249, 1252, 1253.
198 *Id.* at 1249–53, 1255; Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1190 (2021).
199 It is worth noting, as a factual matter, that the operating system of the case is much different than the operating systems we are familiar with today.

under traditional copyright law principles, where protection of the expression would also ensnare rights over use, copyright law gives way and leaves the expressive elements unprotected.[200] Computer programs appear to be an exception to that principle considering the historic amendments made to the Act to accommodate software.[201] It is the nature (or more precisely the nature and *the scope*) of this exception that has created havoc for copyright law ever since.

The final aspect of this case relates to compatibility, another tricky issue introduced into copyright law by computers and technology. Franklin cleverly attempted to frame the compatibility issue as one regarding the idea/expression dichotomy and the merger doctrine.[202]

In responding to this, the Third Circuit in *Apple* made a reasonably insightful observation: "Franklin may wish to achieve total compatibility with independently developed application programs written for the Apple II, but that is a commercial and competitive objective which does *not* enter into the somewhat metaphysical issue of whether particular ideas and expressions have merged."[203] What makes this statement insightful is that the court was not mislead. It is true that copyright law does strike a balance between competition and protection, which may relate to the desirability of compatibility. In fact, earlier in the case, the appellate court cited the famous statement from *Herbert Rosenthal* that the line between ideas and expression is a pragmatic one which takes into consideration "the preservation of the balance between competition and protection reflected in the patent and copyright laws."[204] However, striking this balance does not necessarily implicate merger. Whether merger is implicated depends on the expressions themselves.[205] As the court states: "In essence, this inquiry is no different than that made to determine whether the expression and idea have merged, which has been stated to occur when there are no or few other ways of expressing a particular idea."[206]

---

200 *Apple Comput., Inc.*, 714 F.2d at 1247–49, 1252–54.

201 *See id.*

202 *See id.* at 1254 ("Franklin claims that whether or not the programs can be rewritten, there are a limited 'number of ways to arrange operating systems to enable a computer to run the vast body of Apple-compatible software' . . . This claim has no pertinence to either the idea/expression dichotomy or merger."). *But see Google LLC*, 141 S. Ct. at 1197, 1202 (recognizing that, as Google argued, declaring code can be treated differently from implementing code under fair use). *See supra* notes 177–201 and accompanying text; *infra* notes 203–12 and text accompanying.

203 *Apple Comput., Inc.*, 714 F.2d at 1253 (emphasis added).

204 *Id.* (citing Herbert Rosenthal Jewelry v. Kalpakain, 446 F.2d 738, 742 (9th Cir. 1971)).

205 *See id.* at 1253.

206 *Id.*

The Third Circuit, here, was not confused about the difference between merger, which *does* apply to computer programs, and the form-function doctrine, which *does not* apply, an issue that seems to confuse later courts. Rather, Franklin tried to argue that an OS is so utilitarian that it should not be subject to protection.[207] Some courts have viewed this consideration like merger and/or form-function or have relied on section 102(b).[208] However, the Third Circuit was not convinced by Franklin's argument.[209]

Several additional points may be made here. As suggested, the balance of competition and protection factor into the appropriate boundary for the idea/expression dichotomy in general. If this principle is accepted, then it follows that, for software, this balance is different due to the utilitarian nature of software.[210] Of course, the Third Circuit, in *Apple,* does not attempt to address that question.[211] This is likely because it was not properly raised or more likely because it did not appreciate its significance, in general, for the scope of protection that copyright law should provide to software (as distinguished from the question of software copyrightability). However, as to the question of merger, it does not seem particularly plausible that merger should have applied to the entire program that was allegedly copied.[212]

---

[207] *Id.* at 1249, 1251.

[208] *See, e.g.*, Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 703–04 (2d Cir. 1992); Hickey, *supra* note 81, at 696 (citing Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 708–10 (2d Cir. 1992) (treating merger and functionality as essentially the same doctrine); *cf.* Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 815 (1st Cir. 1995), *aff'd*, 516 U.S. 233 (1996) (holding "the Lotus menu command hierarchy is an uncopyrightable 'method of operation'" under 17 U.S.C. § 102(b)). *See also infra* note 318–46 and accompanying text.

[209] *Apple Comput., Inc*., 714 F.2d at 1242.

[210] *See Lotus Dev. Corp*., 49 F.3d at 819 (Boudin, J., concurring) ("Utility does not bar copyright (dictionaries may be copyrighted), but it alters the calculus."); *Comput. Assocs. Int'l., Inc*., 982 F.2d at 704 (quoting Peter G. Spivack, Comment, *Does Form Follow Function? The Idea/Expression Dichotomy in Copyright Protection of Computer Software*, 35 UCLA L. REV. 723, 755 (1988)) ("The essentially utilitarian nature of a computer program further complicates the task of distilling its idea from its expression . . . In order to describe both computational processes and abstract ideas, its content 'combines creative and technical expression.' . . . The variations of expression found in purely creative compositions, as opposed to those contained in utilitarian works, are not directed towards practical application.").

[211] *See, e.g., Apple Comput., Inc.,* 714 F.2d at 1255.

[212] *Cf.* NIMMER ON COPYRIGHT § 2A.10(B)(2) n.44, *supra* note 93 (stating, in connection with merger, that "a given routine or component of the software may properly fall within the scope of merger"); NIMMER ON COPYRIGHT § 13.03(e)(2)(b), *supra* note 6 (discussing computer searching and sorting algorithms as potentially good examples of merger); Note, however, that the appellate court did remand regarding merger. *See Apple Comput., Inc*., 714 F.2d at 1253 ("Although there seems to be a concession by Franklin that at least some of the programs can be rewritten, we do not believe that the record on that issue is so clear that it can be decided at the appellate level. Therefore, if the issue is pressed on remand, the

While *Apple v. Franklin* dealt with the question of copyrightability of software and literal infringement, a later case, *Whelan v. Jaslow,* dealt with a more esoteric and, in some sense, more metaphysical subject: non-literal infringement of a copyright in computer software. [213]

The facts of the case are relatively straight-forward. Elaine Whelan had written a program for Jaslow Laboratory that assisted in running a dental lab.[214] The computer program, referred to as the "Dentalab" program, ran on an IBM mainframe computer.[215] However, Rand Jaslow, an officer and shareholder in Jaslow Lab, later developed his own program, referred to as the "Dentcom" program, to assist in running a dental lab.[216] This latter program, however, ran on a personal computer.[217] Jaslow Lab then ended its business relationship with Whelan.[218]

Whelan is a difficult or troubling case in many respects. Jaslow clearly had access to Whelan's program and it might even be argued that it was clear that, in a high-level sense, because Jaslow modeled his own program after Whelan's, some copying took place.[219]

The hard question for the court was: did enough copying take place to create liability under copyright law?[220] More specifically, since Jaslow did not literally copy Whelan's program, the question presented was whether copyright law provides protection when a computer program is non-literally copied.[221] In other areas of copyright law, non-literal protection had been established.[222] Partly for this reason, therefore, the Third Circuit decided that computer programs are also entitled to non-literal protection.[223] The appellate

---

necessary finding can be made at that time.").

213 *Compare Apple Comput., Inc*., 714 F.2d at 1246, *with* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1241 (3d Cir. 1986).

214 *Whelan Assocs., Inc.*, 797 F.2d at 1226.

215 *Id.*

216 *Id.*

217 *Id.* at 1226–27.

218 *Id.* at 1226.

219 *Id.* at 1228 ("Dr. Moore testified that although the Dentcom program was not a translation of the Dentalab system, the programs were similar in three significant respects. He testified that most of the file structures, and the screen outputs, of the programs were virtually identical . . . He also testified that five particularly important 'subroutines' within both programs—order entry, invoicing, accounts receivable, end of day procedure, and end of month procedure—performed almost identically in both programs.").

220 *Id.* at 1225.

221 *Id.* at 1224–25.

222 *Id.* at 1234.

223 *Id.* at 1234, 1238 ("By analogy to other literary works, it would thus appear that the copyrights of computer programs can be infringed even absent copying of the literal elements of the program . . . Thus, copyright principles derived from other areas are applicable in the field of computer programs."). Other courts, notable, the Fifth Circuit, declined to follow *Whelan* and initially rejected the notion of non-literal protection for

court also needed to decide, however, whether, under the facts of this case, non-literal copyright infringement had occurred.[224] The district court determined that it had.[225]

Unlike the Third Circuit in *Apple* which dealt with copyrightability and literal infringement, this Third Circuit case was one of the first cases to address the scope of non-literal protection available for software under copyright law.[226] The issue, its complexity, and its legal significance might be easier to understand if the question for the court is framed from another perspective.

The question for the court might be stated as follows: is the similarity between Whelan's and Jaslow's programs due to the similarity of copyrightable expression appearing in Jaslow's program and originating from Whelan's program or is it due to the similarity of noncopyrightable elements appearing in Jaslow's program and originating from Whelan's program (e.g., the ideas therein)?[227] That is, the court needed to address a fundamental question about the scope of protection that copyright law should afford to computer programs, apart from the more basic question, already answered, regarding whether or not computer programs are copyrightable at all.[228]

Unfortunately, possibly due to an unsophisticated understanding of computers and computer programming, it is now generally recognized while the Third Circuit in *Whelan* may have reached the right outcome the appellate court's reasoning had some logical flaws.[229] Experts in the field have heavily criticized the opinion, and very few circuits have adopted the Third Circuit's approach.[230] Nonetheless, the *Whelan* court appears to be one of the first to refer to the structure, sequence and organization ("SSO") of a computer

---

computer programs. *See* Plains Cotton Coop. Ass'n v. Goodpasture Comput. Serv., Inc., 807 F.2d 1256, 1262 (5th Cir. 1987). However, in a later decision, the Fifth Circuit changed its position regarding non-literal protection for computer programs. *See* Eng'g Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335, 1342 (5th Cir. 1994) (adopting the abstraction-filtration-comparison test from Comput. Assocs. Int'l., Inc. v. Altai).

224 *Whelan Assocs., Inc.*, 797 F.2d at 1248.

225 *Id.*

226 *Compare* Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1242 (3d Cir. 1983), *with Whelan Assocs., Inc*., 797 F.2d at 1224–25.

227 *See Whelan Assocs., Inc*., 797 F.2d at 1224–25.

228 *See id.* at 1224.

229 *See, e.g.*, Englund, *supra* note 6, at 879–82.

230 *See generally* Englund, *supra* note 6; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1082–83; Mark Kretschmer, *Copyright Protection For Software Architecture: Just Say No!*, 1988 COLUM. BUS. L. REV. 823 (1988); Peter G. Spivack, Comment, *Does Form Follow Function? The Idea/Expression Dichotomy In Copyright Protection of Computer Software*, 35 UCLA L. REV. 723, 747–65 (1988); Plains Cotton Coop. Ass'n v. Goodpasture Comput. Serv., Inc., 807 F.2d 1256, 1262 (5th Cir. 1987) (declining to follow *Whelan*).

program,[231] a construct still used by courts in connection with assessing non-literal infringement of software. One, for example that is later employed by the Federal Circuit in its 2014 decision in Oracle v. Google discussed in more detail, *infra.*[232]

In addressing the question above, the *Whelan* court ultimately needed to apply Learned Hand's levels of abstractions[233] to determine the line between ideas and expression. However, most likely, in a misguided attempt to provide sufficient protection for computer programs, the appellate court drew the line too far in favor of overprotection, stating "the purpose or function of a utilitarian work would be the work's idea, and everything that is not necessary that purpose or function would be part of the expression of the idea."[234]

Part of *Whelan*'s misstep appears to be conflating the merger doctrine with the form-function doctrine, an issue that often confuses courts because, under the facts of *Baker*, both principles emerged.[235] In discussing *Baker*, the *Whelan* court stated:

> In deciding this point, the [*Baker*] Court distinguished what was protectable from what was not protectable as follows:
>
> "Where the art [*i.e.*, the method of accounting] it teaches cannot be used without employing the methods and diagrams used to illustrate the book, or such as are similar to them, such methods and diagrams are to be considered as necessary incidents to the art, and given to the public." . . . Applying this test, the Court held that the blank forms were necessary incidents to Selden's method of accounting, and hence were not entitled to any copyright protection . . . . The Court's test in *Baker v. Selden* suggests a way to distinguish idea from expression. [236]

The statement from *Baker* appears to be more a statement about use rather than idea/expression,[237] but the *Whelan* court purports to derive a rule about idea

---

231 *See Whelan Assocs., Inc.*, 797 F.2d at 1224 ("[I]n this case of first impression in the courts of appeals, we must determine whether the structure (or sequence and organization) of a computer program is protectable by copyright, or whether the protection of the copyright law extends only as far as the literal computer code.").

232 Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1191 (2021). *See also supra* text accompanying notes 183–232; *infra* notes 330–57 and accompanying text.

233 *See* Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931); *see also supra* notes 31–39 and accompanying text.

234 *Whelan Assocs., Inc.*, 797 F.2d at 1236.

235 *See supra* notes 66–77 and accompanying text.

236 *Whelan Assocs., Inc.*, 797 F.2d at 1236 (quoting Baker v. Selden, 101 U.S. 99, 103 (1880)).

237 Baker v. Selden, 101 U.S. 99, 103 (1880).

versus expression, suggesting some confusion.[238] Likewise, *Whelan* recognizes the wisdom from Learned Hand that the line between idea and expression is "inevitably ad hoc," but does not take it to heart, stating: "Although we acknowledge the wisdom of Judge Hand's remark, we feel that a review of relevant copyright precedent will enable us to formulate a rule applicable in this case."[239] Thus, leading to the statement quoted above, which has been criticized by others.[240]

Despite the *Whelan* court's overly general statement, when it comes to analyzing the work at issue, it was able to appropriately distinguish an idea from an expression.[241] For example, the Third Circuit states: "Because that idea could be accomplished in a number of different ways with a number of different structures, the structure of the Dentalab program is part of the program's expression, not its idea."[242] Later, the court also states: "The conclusion is thus inescapable that the detailed structure of the Dentalab program is part of the expression, not the idea, of that program."[243] Furthermore, the *Whelan* court draws an insightful analogy between computer code, compilations, and other pre-existing works, an analogy discussed in more detail later:

> The Copyright Act of 1976 provides further support, for it indicates that Congress intended that the structure and organization of a literary work could be part of its expression protectable by copyright. Title 17 U.S.C. § 103 (1982) specifically extends copyright protection to compilations and derivative works. Title 17 U.S.C. § 101, defines "compilation" as "a work formed by the collection and *assembling* of preexisting materials or of data that are selected, *coordinated, or arranged* in such a way that the resulting work as a whole constitutes an original work of authorship," and it defines "derivative work," as one "based upon one or more preexisting works, such as . . . *abridgement, condensation*, or any other form in which a work may be *recast*, transformed, or adapted." [] Although the Code does not use the terms "sequence," "order" or "structure," it is clear from the definition of compilations and derivative works, and the protection afforded them, that Congress was aware of the fact

---

238 *Whelan Assocs., Inc*., 797 F.2d at 1235.

239 *See id.*

240 *See* Englund, *supra* note 6, at 867–73; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1051–57; Kretschmer, *supra* note 230, at 824–27; Spivack, *supra* note 230, at 729–31; *see also* Plains Cotton Coop. Ass'n v. Goodpasture Comput. Serv., Inc., 807 F.2d 1256, 1262 (5th Cir. 1987) (declining to follow Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222 (3d Cir. 1986)).

241 *Whelan Assocs., Inc*., 797 F.2d at 1236.

242 *See id.* at 1236 n.28.

243 *See id.* at 1239.

> that the sequencing and ordering of materials could be copyrighted, *i.e.*, that the sequence and order could be parts of the expression, not the idea, of a work.[244]

Also, in deliberating about the idea/expression dichotomy, the *Whelan* court considers a copyright law doctrine: scène à faire.[245] The appellate court does little with this doctrine in terms of applying it to the situation at hand, however, it becomes important for the next case, *Computer Associates v. Altai*,[246] with respect to the scope of protection of computer software, in general. Therefore, this doctrine is worth considering here, by way of introduction.

The concept behind this latter doctrine is that some forms of expression have become so commonplace as a means to express a particular idea that to allow such expressions to be copyrightable would provide a party claiming ownership of the expression an unwarranted amount of control over the underlying idea.[247] That is, permitting protection would provide more rights than is intended to be conveyed via copyright. Therefore, although in these situations, a merger has not occurred, nonetheless, similar to a merger, the interest of having certain underlying ideas freely available trumps the interest of otherwise providing protection for this particular expression.

To illustrate this point, consider a western movie that includes a town drunk. A town drunk portrayed in a movie might be sufficiently specific or concrete that it amounts to expression. However, a town drunk in a western is also sufficiently commonplace that if a particular western movie is accused to be a copyright infringement of another western, the mere fact that both westerns include a town drunk is generally not sufficiently probative evidence to show that one was a copy of the other. Rather, if the similarity of the works that flows from including a town drunk were considered probative of copying, that might discourage legitimate incorporation of a town drunk as a "stock" or "commonplace" feature of a western.

That both westerns may include a town drunk is generally not probative of copying given the commonplace nature of the expression. In this way, the scène à faire doctrine may be employed to limit the scope of non-literal protection for a copyrightable work. Today, scène à faire, as well as the doctrine of merger, have taken on significance in evaluating the scope of non-literal protection available for computer software.[248]

It might, perhaps, in 20-20 hindsight be slightly unfair at this point to

---

244 *Id.* (emphasis in original). *See also supra* note 128–56 and accompanying text.

245 *See Whelan Assocs., Inc*., 797 F.2d at 1236.

246 *See generally* Comput. Assocs., Int'l., Inc. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992).

247 *See, e.g.*, Walker v. Time Life Books, 784 F.2d 44, 50 (2d Cir. 1986); Atari, Inc. v. N. Am. Philips Consumer Elecs. Corp., 672 F.2d 607, 616 (7th Cir. 1982), *cert. denied*, 459 U.S. 880 (1982); See v. Durang, 711 F.2d 141, 143 (9th Cir. 1983).

248 *see infra* note 511; *See infra* notes 575–82 and accompanying text.

observe that the Third Circuit in *Whelan* greatly simplified its job. Deciding the case became quite simple using the court's previously cited distinction between the idea and expression of the program at issue. Virtually everything in Whelan's program was protectable expression, both literally and non-literally.[249] Therefore, while the facts of this case raised potentially challenging legal and factual questions regarding the scope of protection for software (e.g., how much copying might be legally permissible), these issues ultimately were not fully resolved.

The rationale for non-literal protection in this context, of course, is that, without it, unscrupulous copyists potentially could make small changes to a literal work and avoid infringement.[250] Likewise, in terms of creating the proper incentives for authors, there could be an undersupply of software if those who might have had the incentive to create software do not expend the effort because their creations can be quite easily "ripped off."[251] In this vein, to credit the Third Circuit in *Whelan* somewhat, evidence of the amount of work that it takes or took to create a program may have influenced its decision.[252] We now know, of course, that the amount of effort expended is not relevant to copyrightability, and, likewise, it is irrelevant to the scope of protection provided.[253] However, at the time this case was decided, the "sweat of the brow doctrine" was alive and well because this decision predates the *Feist* decision of the Supreme Court.[254]

Nonetheless, even considering that factor, the *Whelan* court, though appreciating its role in striking a balance,[255] struck the balance too far in favor of overprotection. Whereas, providing no non-literal protection would likely produce an under supply of software, likewise, overprotecting the non-literal aspects of the work may produce a similar under supply. Instead of foregoing producing software for lack of protection, in the face of an overly protective approach, some that would otherwise produce or create software may opt to not create software due to the potential risks associated with inadvertently non-literally infringing the work of another. In a very real sense, then, as was previously indicated, the job of a court in resolving the idea/expression dichotomy is to provide a balance between protection on

---

249 *Whelan Assocs., Inc*., 797 F.2d at 1243.

250 *See* Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931).

251 *Whelan Assocs., Inc*., 797 F.2d at 1237.

252 *See* Comput. Assocs., Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 711 (2d Cir. 1992).

253 *Id.*

254 Feist Publ'ns, Inc. v. Rural Tel. Co., 499 U.S. 340, 353 (1991).

255 *See Whelan Assocs., Inc*., 797 F.2d at 1235 n.27 ("Achieving the proper incentive has been a longstanding task of courts.").

one hand and competition on the other.[256] It is in this respect that the *Whelan* court mis-stepped.

The job of a court, when faced with a copyright law situation involving the scope of protection, is to determine the appropriate idea/expression boundary.[257] So, too, then, it is for courts to resolve where the line should fall for computer programs to provide the best outcome for society, while also being true to the purpose of Congress in providing copyright protection.[258] By the very nature of copyright law, then, the line for computer programs will be placed differently than for other forms of expression that also receive copyright protection.[259] However, it was left for later courts, coming after *Whelan*, to attempt to resolve the appropriate balance regarding scope of protection with respect to software. Furthermore, the calculus of where such a line should be drawn, as difficult as such calculus may be, is made that much more difficult due to how the form-function doctrine and its exemption regarding software comes into the calculation, so to speak.

Computer Associates v. Altai (hereinafter Altai), illustrates the enormous complexities facing courts seeking to determine an appropriate scope of non-

---

256 *See, e.g.*, *Comput. Assocs. Int'l., Inc*., 982 F.2d at 696 ("[C]opyright law seeks to establish a delicate equilibrium. On the one hand, it affords protection to authors as an incentive to create, and, on the other, it must appropriately limit the extent of that protection so as to avoid the effects of monopolistic stagnation. In applying the federal act to new types of cases, courts must always keep this symmetry in mind."); NIMMER ON COPYRIGHT § 2A.03(B), *supra* note 37 ("[W]e must approach the field wearing bifocals—artistic creativity deserves protection at the same time that the evils of monopolizing functional activities must be avoided. Decisions must therefore be reached with sensitivity to both sides of the ledger: If according protection to a given form of expression threatens to forestall competition in a given field of endeavor, that consideration alone might counsel the opposite resolution.").

257 *See Whelan Assocs., Inc.*, 797 F.2d at 1233 (determining the scope of copyright protection for a computer program).

258 *See, e.g.*, *Comput. Assocs. Int'l., Inc*., 982 F.2d at 696 ("[C]opyright law seeks to establish a delicate equilibrium. On the one hand, it affords protection to authors as an incentive to create, and, on the other, it must appropriately limit the extent of that protection so as to avoid the effects of monopolistic stagnation. In applying the federal act to new types of cases, courts must always keep this symmetry in mind."); Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (2d Cir. 1971) ("The guiding consideration in drawing the line is the preservation of the balance between competition and protection reflected in the patent and copyright laws."); NIMMER ON COPYRIGHT § 2A.03(B), *supra* note 37 ("[W]e must approach the field wearing bifocals—artistic creativity deserves protection at the same time that the evils of monopolizing functional activities must be avoided. Decisions must therefore be reached with sensitivity to both sides of the ledger: If according protection to a given form of expression threatens to forestall competition in a given field of endeavor, that consideration alone might counsel the opposite resolution."). *See also supra* notes 235–57 and accompanying text; *infra* notes 259–65 and text accompanying.

259 *See supra* notes 31–39 and accompanying text.

literal copyright protection for software. [260] It would not in any sense be an exaggeration to observe that this Second Circuit case is not only the most important case discussed so far, but also, that it may be, in terms of its influence, the most important decision in the area of non-literal copyright protection for computer software. Since the decision, every circuit that has considered the issue has essentially adopted this approach.[261]

The case involved an appeal by Computer Associates' ("CA") of a decision of non-infringement regarding a program written by Altai, called OSCAR 3.5.[262] At the trial level, Altai's program OSCAR 3.4 was held to infringe Computer Associates' copyright in its program, CA-SCHEDULER, but Altai did not appeal that holding.[263] Computer programs that schedule jobs for execution on IBM mainframe computers were the subject of the litigation.[264] The main function of such software is to create a schedule to specify when certain tasks are to be executed by a computer, and then to control the computer as those tasks are executed.[265]

Computer Associates had a program, referred to as CA-SCHEDULER, that performed this function.[266] CA-SCHEDULER included a sub-program, referred to as ADAPTER, that translated from one programming language to another.[267] ADAPTER essentially provided operating system compatibility.[268] Thus, ADAPTER ensured that CA-SCHEDULER may be executed on several different computers that have a number of different operating systems.[269] When it was necessary to engage the operating system, ADAPTER executed the appropriate operating system function call for the particular function being performed.[270]

Altai had its own scheduling program, called ZEKE.[271] Before the facts that led to the litigation, Altai, due to consumer demand, decided that it needed to rewrite ZEKE so that it could run on another operating system.[272] It had been written to run only with a VSE operating system.[273]

---

260 *See generally Comput. Assocs. Int'l, Inc.* , 982 F.2d at 693.
261 *See infra* notes 287–301 and accompanying text.
262 *Comput. Assocs. Int'l., Inc*., 982 F.2d at 696.
263 *Id.* at 696–97.
264 *Id.* at 698.
265 *Id.*
266 *Id.*
267 *Id.*
268 *Id.* at 698–99.
269 *Id.* at 699.
270 *Id.*
271 *Id.*
272 *Id.*
273 *Id.*

A high-level executive of Altai then convinced a CA employee, Arney, to come work for Altai.[274] Arney, unknown to Altai and to the executive that recruited him, was familiar with CA's ADAPTER program and, in fact, took source code versions of that program with him when he left CA.[275] Arney then convinced Altai to use a common system interface, that is, the functional approach that ADAPTER implemented, but Arney never told Altai about his familiarity with ADAPTER and the source code copies he had in his possession.[276] Arney then developed a program called OSCAR 3.4 to perform the interface function.[277] About 30% of OSCAR 3.4's code was copied from ADAPTER.[278] CA ultimately suspected what had transpired and sued Altai for copyright infringement and trade secret misappropriation.[279] After being sued, Altai learned of Arney's copying.[280]

Upon advice of counsel, Altai rewrote OSCAR.[281] Arney was excluded from the process.[282] The code for ADAPTER was locked away and programmers unfamiliar with OSCAR 3.4 were used.[283] This produced OSCAR 3.5.[284]

The district court had held that OSCAR 3.5 and ADAPTER were not substantially similar, and CA appealed that decision, arguing the district court failed to protect the non-literal elements of the copyright in ADAPTER.[285]

The Second Circuit determined that the district court below was correct to reject the approach employed in the *Whelan* case.[286] The appellate court adopted, instead, a successive filtering approach, referred to as abstraction-filtration-comparison ("AFC").[287]

Under this approach, a court applies the levels of abstraction approach to

---

274 *Id.*

275 *Id.* at 699–700.

276 *Id.* at 700.

277 *Id.*

278 *Id.*

279 CA asserted that Altai had constructive knowledge of the misappropriation. One of the issues in the case was whether copyright law pre-empted the state law trade secret misappropriation claim. Not infrequently, a copyright infringement case for software begins as a trade secret misappropriation case or vice-versa. *See, e.g.*, Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1227 (3d Cir. 1986) (noting the case started as a trade secret misappropriation suit).

280 *Comput. Assocs. Int'l., Inc*., 982 F.2d at 700.

281 *Id.*

282 *Id.*

283 *Id.*

284 *Id.*

285 *Id.* at 701.

286 *Id.* at 702.

287 *Id.* at 706.

copyrighted code (abstraction).[288] At each so-called abstraction level, certain elements deemed unprotectable are filtered out (filtration).[289] Then, at each level, what is left after filtration is compared with the accused product (comparison).[290] If there is substantial similarity between the accused work and the copyrighted work at any level determined to include expression, then copyright infringement has occurred.[291]

The filtration step, again, referred to as successive filtering, is a crucial step in the analysis and has proven to be the most controversial step.[292] According to the Second Circuit, per the *Altai* decision, a court is to filter out (1) elements attributable to or dictated by efficiency, (2) elements attributable to or dictated by external factors, and (3) elements attributable to or taken from the public domain.[293] The rationale for filtering out these elements is that they are not protectable under copyright law. For example, the merger doctrine is considered to prevent elements under category (1) from being treating as protected expression, and the scène à faire doctrine is considered to prevent elements under category (2) from being treated as protected expression.[294]

An important aspect of the opinion is a so-called "clean" room that Altai employed.[295] This refers to the process CA employed to rewrite OSCAR.[296] CA excluded Arney, used programmers not familiar with OSCAR 3.4, and prevented anyone from having access to the source code versions that Arney took from CA.[297] One reason why CA employed this approach was to break the

---

288 *Id.* at 707.

289 *Id.*

290 *See id.* at 710. It was noted by the appellate court in *Altai* that the district court filtered out unprotectible elements from OSCAR 3.5, rather than from ADAPTER. This may show the challenge presented to courts by the complexity of this type of analysis in that the lower court got confused regarding which program is to be filtered. *See also infra* note 478 and accompanying text.

291 *Comput. Assocs. Int'l., Inc*., 982 F.2d at 706.

292 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1358 (Fed. Cir. 2014). Note also, that this step as a practical matter may be viewed as the most important step in terms of determining the scope of protection. *See Comput. Assocs. Int'l., Inc*., Inc., 982 F.2d at 707 (quoting Brown Bag Software v. Symantec Corp., 960 F.2d 1465, 1475 (9th Cir)) ("Strictly speaking, this filtration serves 'the purpose of defining the scope of plaintiff's copyright.'") (endorsing 'analytic dissection' of computer programs in order to isolate protectable expression); *see also* Andrew B. Hebl, *A Heavy Burden: Proper Application of Copyright's Merger and Scènes à Faire Doctrines*, 8 WAKE FOREST INTELL. PROP. L.J. 128, 137 (2008); Ocasio, *supra* note 97, at 315.

293 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 708, 710.

294 *See id.* at 709.

295 *See* Evan Finkel, *Update to: Copyright Protection for Computer Software in the Nineties*, 8 SANTA CLARA HIGH TECH. L.J. 99, 105 (1992).

296 *See id.*

297 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 700.

connection often made between factual similarity and legal or substantial similarity for the purpose of determining copyright infringement.[298] Altai had access to CA's ADAPTER program and legally could not rebut that element of the case against it.[299] However, the clean room approach provided a basis for Altai to assert that similarities between its "clean" program, here OSCAR 3.5, and CA's program, were not from having had access to ADAPTER.[300] Thus, while not dispositive, a clean room process provides a powerful evidentiary tool available to a defendant in such cases.[301]

Another important aspect of the case relates to the issue of software compatibility.[302] Recall that this issue came up previously in connection with the *Apple* case before the Third Circuit.[303] Compatibility of software had become an increasingly significant issue with the proliferation of software technology. For example, as software produced by different venders becomes layered in complex systems and networks that interact, interoperability becomes an important feature that consumers desire.

However, whether copyright infringement occurs when one vender without obtaining authority writes software to interoperate with software from another vender remains unclear.[304] In other words, is compatibility a consideration not unlike the hardware platform on which the code will execute, such that, those aspects of the code included as a result of the particular hardware platform are not subject to copyright protection or, instead, is compatibility without first obtaining appropriate permission attempting to leverage the popularity of

---

298 *See id*. at 701.

299 *See id.*

300 *See id.* at 702.

301 *See id.*

302 *See id.* at 698.

303 *See* Apple Comput. Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1243 (3d Cir. 1983).

304 *See* Samuelson, *supra* note 6, at 263; Vasilescu-Palmero, *supra* note 6, at 171; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1047; *see also* Menell, *supra* note 6, at 459; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 64–65; Armstrong, *supra* note 6, at 131; *The Uneasy Case for Software Copyrights Revisited*, *supra* note 6, at 1747; *Tailoring Legal Protection for Computer Software*, *supra* note 6, at 1330; Ginsburg, *supra* note at 6, at 2559; Dennis, *supra* note 6, at 33; Miller, *supra* note 6, at 978; Reichman, *supra* note 6, at 641; *A Manifesto Concerning the Legal Protection of Computer Programs*, *supra* note 6, at 2310; Weinreb, *supra* note 6, at 1150; Englund, *supra* note 6, at 866; Nimmer, *supra* note 6, at 626; NIMMER ON COPYRIGHT § 13.03(F), *supra* note 6; INTERFACES ON TRIAL: INTELLECTUAL PROPERTY AND INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY, *supra* note 6; INTERFACES ON TRIAL 2.0, *supra* note 6. *See generally Special Issue: Software Interface Copyright*, *supra* note 6 (devoting Harvard Journal of Law and Technology's Spring Issue to the topic of Copyright and Software Interfaces with 9 articles by well-known intellectual property law experts including: Pamela Samuelson, Peter Menell, and Mark Lemley).

someone else's creative endeavor?

This is no simple question and, aside from the complexity of this question as a matter of policy, the answer may likewise vary depending on the details of a given situation. Two important cases after *Altai*, the 2014 Federal Circuit decision in *Oracle v. Google* and the First Circuit decision in *Lotus v. Borland*, attempt to grapple with this question and reach different conclusions employing different approaches.[305]

However, one should understand the position taken by the Second Circuit in *Altai*. The court is not necessarily saying that it is acceptable to copy expression for purposes of compatibility and, thus it is an external factor to be filtered.[306] Rather, in determining the scope of protection, the court is seeking to discern those elements that, while potentially leading to similarity, would not be probative of illegal or illicit copying.[307] For example, suppose that to encourage others to write applications that will execute on its iPhone device, Apple makes freely available to programmers certain routines to execute some relatively basic functions, such as time keeping, location determining, orientation of the phone in three dimensions, etc. In that case, if one application program that executed on an Apple iPhone device is accused of being an infringement of another application program that also executes on the Apple iPhone device, it may be that such expression within the code relating to those aspects would *not* be probative of copying and should appropriately be filtered out under the AFC approach.[308]

On one hand, the *Altai* court asserts that it "breaks no new ground"[309] and, instead, draws on "familiar copyright doctrines."[310] On the other hand, the Second Circuit in *Altai* recognizes that its approach will result in "narrowing the scope of protection" for computer programs.[311] These assertions are not easily reconciled. If the court simply applies well-known or well-established copyright law principles, why should it expect that to result in narrowing the scope of protection for computer programs?

There are at least two plausible reasons why the Second Circuit concluded that narrowing protection for computer programs would result, even assuming

---

305 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1353 (Fed. Cir. 2014) (finding copyright protects the expression of a process or method); Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 813 (1st Cir. 1995) (holding that the menu command hierarchy was uncopyrightable because it was a method of operation). *See supra* notes 330–357, 367 and accompanying text; *see infra* notes 459–467 and accompanying text.

306 *See Lotus Dev. Corp.*, 49 F.3d at 818.

307 *See id.*

308 *See* Comput. Assocs. Int'l., Inc. v. Altai, Inc., 982 F.2d 693, 706 (2d Cir. 1992).

309 *See id.*

310 *See id.*

311 *See id.* at 712.

that it was merely drawing on familiar copyright law doctrines.[312] The first reason may relate to the court's desire to balance competition and protection.[313] That is, less protection is probably appropriate for software because the market incentives that exist do a reasonably good job of encouraging the creation of such works. In other words, without protection, some amount of code would still be written even with the knowledge that others would be free to copy it. However, it may be that this amount of code would still be less than might result if some protection is afforded. Since the goal of protection is to encourage the creation of works ultimately, less protection may be more desirable here than is provided in other areas, such as for books, as an example. More to the point, if too much protection is provided,[314] it may very well undermine the incentive to generate such works. This point appears consistent with the general understanding regarding how much protection to afford software.[315] Potentially, due to a reasonably well-functioning market, software appears to be generated even if accorded less protection than other types of copyrightable works. Some appear to believe, for example, that software should receive no copyright protection (or at least significantly less protection) precisely because such protection undermines appropriate incentives by being overprotective.[316]

Another reason the court might have expected a narrowing of protection for computer programs, as the Second Circuit states in *Altai*, relates to the utilitarian nature of software.[317] The court may have seen the utilitarian nature of software as operating as a sort of bound on the creative form that might otherwise be expressed. In other areas, using literary works as a typical example, practical considerations of workability, effectiveness, allocation of resources, etc., generally do not restrict the creative endeavors of an author. However, because software is useful and ultimately seeks to perform a given task, such considerations may limit or bound the universe of creative expressions that might otherwise result. The *Altai* court may have expected some similarity in approaches to result even in the context of totally independent development of creative expressions.[318] In attempting to arrive at a goldilocks-like balance

---

312 *See id.*

313 *See id.* at 696.

314 *See supra* notes 31–39, 156–60 and accompanying text; *See infra* notes 409–13 and accompanying text (regarding the Supreme Court decision in *Google v. Oracle*).

315 *See* Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1196 (2021).

316 *See supra* notes 4, 156–60 and accompanying text. *See infra* notes 409–13 and accompanying text (regarding the Supreme Court decision in *Google v. Oracle*).

317 *See Comput. Assocs. Int'l. Inc*., 982 F.2d at 708.

318 See *id.* at 709 ("[I]t is quite possible that multiple programmers, working independently will design the identical method employed in the allegedly infringing work . . . . Under these circumstances, the fact that two programs contain the same efficient structure may as likely lead to an inference of independent creation as it does to one of copying."); *see also* NIMMER ON COPYRIGHT § 13.03(F), *supra* note 6 ("[E]ven two programs that have

between competition and protection, this understanding might lead a court to conclude a narrower scope of protection would be appropriate so as to not inadvertently establish hurdles for advancing further creative endeavors.

The Second Circuit was correct that its approach did narrow the scope of non-literal protection for programs.[319] However, the court did more than simply apply well-known principles.[320] To the contrary, the decision produced a shift (termed here a "paradigm shift") in the approach applied by courts to determine the scope of non-literal protection afforded computer programs under copyright law.[321] The opinion itself is well-reasoned and clear in most places. It is also a watershed case in that it largely sets the standard to be applied for resolving the scope of non-literal copyright protection for software, at least until the Supreme Court chooses to address the issue.[322]

Nonetheless, there is an important point on which the *Altai* court was confused, and it strikes at the heart of the confusion that many courts appear to have in this particular area. In this instance, the *Altai* court stated incorrectly: "we conclude that those elements of a computer program that are *necessarily* incidental to its *function* are similarly unprotectable."[323] Here, the court has confused the restriction on use rights, the doctrine referred to herein as the form-function doctrine, with the merger doctrine.[324]

Under the merger doctrine, expression "necessarily incidental" to an idea being expressed is unprotectable (e.g., there are only a few ways to express it), as the court later correctly observes.[325] However, the merger doctrine is not consistent with the previous statement made by the court, which instead relates to utility or functionality.[326] Thus, while later in the decision, the court applies the merger doctrine correctly, it nonetheless has some confusion about how merger differs from the notion, applicable to traditional works under copyright law, that where the scope of protection is necessarily entangled with use of the work, then no protection is provided under copyright law.[327] This distinction is important because, for software, the latter principle does not apply. Rather,

---

been created independently may appear similar in many respects.").

319 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 712.

320 *See id.*

321 *See id.* at 703.

322 *See generally* Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183 (2021). *See infra* notes 409–13 and accompanying text.

323 *Comput. Assocs. Int'l, Inc.*, 982 F.2d at 705 (emphasis added); *cf.* Hickey, *supra* note 81, at 696 (treating merger and functionality as essentially the same doctrine and citing *Altai*).

324 *See Comput. Assocs. Int'l. Inc*., 982 F.2d at 708.

325 *See id.* at 708–09.

326 *See id.*

327 *See id.*

section 117, for example, makes it clear that copying is an essential step in the utilization of a software program. [328] Given clear Congressional intent that software be protectable by copyright and recognition by statute, it cannot logically be the case that the form-function doctrine applies to computer programs.[329] For example, copying the ledgers in *Baker* would be an essential step in their use. When courts confuse these two doctrines, the results have the potential to limit the scope of protection for software more than is appropriate.

The next case we consider is the 2014 Federal Circuit opinion in *Oracle v. Google* (hereinafter, *Google (2014)*) which was handed down by the Federal Circuit.[330] This case is complex, and the Federal Circuit attempted to address most, if not all, of the challenging issues previously mentioned. Not unlike *Altai*, the case is reasonably clear and well-reasoned, but still manages to confuse a few issues in its discussion explaining its decision.

Essentially, Google, having purchased Android, wanted to develop a Java platform for Android phones.[331] Java had been developed by Sun and Sun had been purchased by Oracle.[332] Although Google and Oracle attempted to negotiate a deal, the negotiations failed apparently because Oracle wanted Google's implementation to be interoperable with Oracle's Java Virtual Machine, which is central to the Java platform.[333] In general, the basic idea behind Java was to enable programmers to write code that would execute on different types of computer platforms without a developer rewriting the code.[334] Compatibility with the Java Virtual Machine makes this possible.[335] Oracle did not want to grant Google a license because Google wanted to implement Java on Android phones in a way that would not satisfy Java's goal of "write once,

---

328 *See supra* notes 149–66 and accompanying text.

329 *See id.*

330 *See generally* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339 (Fed. Cir. 2014). The Supreme Court denied *certiorari* for this case, which primarily dealt with copyrightability and copyright infringement; however, a later decision by the Federal Circuit in the same case was handed down in 2018, in which the Federal Circuit had to address a fair use question. That later decision was ultimately granted *certiorari* so that the Supreme Court appeared to be headed to resolving issues from the 2014 case and issues from the 2018 case. Oral argument in the case took place on October 7, 2020, and an opinion deciding the case based only on fair use was issued on April 5, 2021. *See infra* notes 409–13 and accompanying text (regarding the Supreme Court decision in *Google v. Oracle*).

331 *See Oracle Am., Inc*., 750 F.3d at 1348.

332 *See id.*

333 *See id.* at 1350; *See also* Samuel J.M. Hartiens, Note, *The Battle of Big Tech: Distinguishing Fair Use and Copyright Infringement with APIs*, 21 FLA. COASTAL L. REV. 1 (2021) (Noting this is worthy of mention because Oracle's goals might be viewed as more pro-interoperability than Google's and, yet, the Supreme Court apparently saw things differently in its ultimate fair use analysis). *Cf. supra* text accompanying notes 250–52.

334 *See Oracle Am., Inc*., 750 F.3d at 1348–49.

335 *See id.* at 1348.

run anywhere."[336]

Java had 166 so-called "API packages" (application programming interface packages) with 6000 methods making up more than 600 classes.[337] 37 of the 166 packages were at issue in the case, three of which were considered "core" Java packages.[338] It is noted that these 37 packages were part of Google's implementation of Java called Dalvik, which included 168 API packages.[339]

Understanding the nature of these so-called API packages is important to appreciate the issues in the case. The case provides a helpful description, as follows:

> Sun wrote a number of ready-to-use Java programs to perform common computer functions and organized those programs into groups it called 'packages.' These packages, which are the application programming interfaces at issue in this appeal, allow programmers to use the prewritten code to build certain functions into their own programs, rather than write their own code to perform those functions from scratch. They are shortcuts. Sun called the code for a specific operation (function) a 'method.' It defined 'classes' so that each class consists of specified methods plus variables and other elements on which the methods operate. To organize the classes for users, then, it grouped classes (along with certain related 'interfaces') into 'packages.'. . . The parties have not disputed the district court's analogy: Oracle's collection of API packages is like a library, each package is like a bookshelf in the library, each class is like a book on the shelf, and each method is like a how-to chapter in a book.[340]

Furthermore,

> Every package consists of two types of source code— what the parties call (1) declaring code; and (2) implementing code. Declaring code is the expression that identifies the prewritten function and is sometimes referred to as the 'declaration' or 'header.' As the district court explained, the 'main point is that this header line of code introduces the method body and specifies very precisely the inputs, name and other functionality.' . . . The expressions used by the programmer from the declaring code command the computer to execute the associated implementing code, which gives the computer the step-by-step instructions for carrying out the declared function.[341]

---

336 *See id.* at 1350.
337 *Id.* at 1349.
338 *Id.*
339 *Id.* at 1350.
340 *Id.* at 1349.
341 *Id.* One might suspect that this distinction originates from the creative endeavors of Google's counsel rather than from engineering considerations.

Finally,

> To use the district court's example, one of the Java API packages at issue is "java.lang." Within that package is a class called "math," and within "math" there are several methods, including one that is designed to find the larger of two numbers: "max." The declaration for the 'max' method, as defined for integers, is: "public static int max(int x, int y)," where the word "public" means that the method is generally accessible, "static" means that no specific instance of the class is needed to call the method, the first "int" indicates that the method returns an integer, and "int x" and "int y" are the two numbers (inputs) being compared . . . A programmer calls the "max" method by typing the name of the method stated in the declaring code and providing unique inputs for the variables "x" and "y." The expressions used command the computer to execute the implementing code that carries out the operation of returning the larger number.[342]

Thus, for the 37 packages at issue, Google copied the declaration portion of the code verbatim and conceded as much.[343] However, Google wrote its own implementing code.[344] Apparently, Google expected that programmers would want to use the same names with the same associated functions as used in Java.[345] It was Google's contention, and the district court ultimately agreed, that this declaring code was not copyrightable.[346]

Oracle, on the other hand, contended that the declaring code was copyrightable and that, therefore, Google's verbatim copying was copyright infringement. [347] Oracle further contended that Google, in effect, by copying the declaring code with the purpose of providing the same names and functions as Java had also "copied the elaborately organized taxonomy within its code of names, methods, classes, interfaces, and packages."[348] In other words, Oracle contended that Google had copied the overall sequence, structure and organization (SSO) of Oracle's code.[349] As to this latter contention, the district court ruled in favor of Google as well, finding that it was, in essence, a command

---

342 *Id.*
343 *Id.* at 1350–1351.
344 *Id.* at 1351.
345 *Id.* at 1350.
346 *Id.* at 1348.
347 *Id.* at 1353.
348 *Id.* at 1350–51.
349 *See id.* at 1364–66; *cf.* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1224 (3d Cir. 1986) ("[I]n this case of first impression in the courts of appeals, we must determine whether the structure (or sequence and organization) of a computer program is protectible by copyright, or whether the protection of the copyright law extends only as far as the literal computer code.").

structure, a system or method of operation and, thus, was not entitled to copyright protection under section 102(b) much like had been ruled by the First Circuit in *Lotus v. Borland*.[350]

Ultimately, the Federal Circuit reversed the district court decision regarding copyrightability of the declaring code and found that Google did infringe Oracle's copyright by its verbatim copying.[351] It also found that Google had copied the SSO of Oracle's code which also constituted copyright infringement.[352] It distinguished *Lotus v. Borland*, but also found that the Ninth Circuit, whose law it was required to apply, had rejected the approach in *Lotus*.[353] The Federal Circuit ultimately remanded the case for retrial on the issue of fair use, which produced yet another appellate decision in 2018 and that decision was ultimately granted *certiorari* by the Supreme Court and resulted in a Supreme Court decision.[354]

A few statements or analysis may be confusing and imply the Federal Circuit was confused on a few subtle points in this 2014 decision; however, for the sake of clarity, in an extremely complex and challenging case, the appellate court correctly resolves the issues. That is, the Federal Circuit rejects the merger analysis applied to the declaring code and rejects the reliance on section 102(b) to suggest the SSO of Oracle's code is uncopyrightable.[355] The Federal Circuit correctly resolves some issues in part ostensibly by relying on Ninth Circuit precedent.[356] However, one important aspect of the opinion is that the Federal Circuit is able to distinguish the merger doctrine from the form-function doctrine, a subtle and important point to understand when determining the correct scope of protection for a software copyright.[357]

Early in its opinion, the Federal Circuit makes several helpful analytic observations. First, the Federal Circuit correctly distinguishes the question of copyrightability from the question of scope of protection, particularly in connection with doctrines such as merger and scène à faire.[358] Here, however,

---

350 *See Oracle Am., Inc*., 750 F.3d at 1364; *Lotus v. Borland* precedes *Oracle v. Google*– however, cases are discussed in reverse order of their chronology because *Lotus v. Borland* did not deal specifically with software. Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 818 (1st Cir. 1995).

351 *See Oracle Am., Inc*., 750 F.3d at 1359–60.

352 *See id.*

353 *See id.* at 1365–68.

354 *See id.* at 1376–77; *see supra* notes 330–52 and accompanying text; *See infra* notes 409–13 and accompanying text (regarding the Supreme Court decision in *Google v. Oracle*).

355 *See* Oracle Am., Inc., 750 F.3d at 1361–62.

356 *See id.* at 1365–66.

357 *Compare id.* at 1367–68 (rejecting functionality argument), *with id.* at 1359–60 (analyzing merger).

358 *See id.* at 1358 (relying on 9th circuit law).

the Federal Circuit says it is simply following Ninth Circuit law.[359] The Federal Circuit also early on recognizes the decisional controversy around section 102(b) and points out that this is a source of dispute between the parties.[360]

Section 102(b) expressly states: "In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work."[361] It is a statement regarding the scope of copyright protection. Section 102(a) expressly states:

> Copyright protection subsists, in accordance with this title, in original works of authorship fixed in any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device. . . .[362]

It is a statement regarding what is copyrightable subject matter.

The first statement of section 102 clarifies that original works of authorship fixed in a tangible medium of expression are eligible for copyright protection and the second statement of section 102 clarifies that such protection "for an original work of authorship [does not] extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form. . ."[363] For example, in conjunction with an "idea," an original work of authorship contains ideas, but those are not subject to copyright protection. Likewise, a similar observation, to the extent it may be applicable, might be made regarding other items mentioned in section 102(b).[364] For example, software could potentially relate in some way to a "procedure," a "system," or perhaps a "method of operation." Thus, while a particular work may be copyrightable as an original work of authorship, copyright protection for the particular work does not extend to the "system" or to the "method of operation," which, on the other hand, may be subject to patent protection in appropriate circumstances.[365]

However, some take the view that section 102(b) is a statement regarding the merger doctrine and/or the form-function doctrine.[366] For example, the First Circuit decision, *Lotus v. Borland*, based its ruling that Lotus' menu command

---

359 *See id.*

360 *See id.* at 1355–56.

361 17 U.S.C. § 102(b) (2021).

362 § 102(a).

363 § 102(b).

364 *Id.*

365 *Id.; see, e.g.*, ROBERT C. FABER, LANDIS ON MECHANICS OF PATENT CLAIM DRAFTING (4th ed. 1997).

366 *See supra* notes 115–27, 208–12 and accompanying text.

hierarchy in Lotus 1-2-3 is uncopyrightable on its understanding that section 102(b) is a statement that "methods of operation" are not copyrightable and that Lotus' menu command hierarchy constituted a method of operation.[367]

The Federal Circuit in *Google (2014)* does not shy away from this controversy stating up front that: "Courts routinely cite *Baker* as the source of several principles incorporated into Section 102(b) that relate to this appeal, including that: (1) copyright protection extends only to expression, not to ideas, systems, or processes; and (2) 'those elements of a computer program that are necessarily incidental to its function are . . . unprotectable.'"[368]

Likewise, the *Google (2014)* court clearly maps out the positions regarding section 102(b) and concludes that Google's position on this question is not the correct interpretation.[369] It states:

> Although the parties agree that Oracle's API packages meet the originality requirement under Section 102(a), they disagree as to the proper interpretation and application of Section 102(b). For its part, Google suggests that there is a two-step copyrightability analysis, wherein Section 102(a) grants copyright protection to original works, while Section 102(b) takes it away if the work has a functional component. To the contrary, however, Congress emphasized that Section 102(b) "in no way enlarges or contracts the scope of copyright protection" and that its "purpose is to restate . . . that the basic dichotomy between expression and idea remains unchanged.". . . "Section 102(b) does not extinguish the protection accorded a particular expression of an idea merely because that expression is embodied in a method of operation." . . . Section 102(a) and 102(b) are to be considered collectively so that certain expressions are subject to greater scrutiny . . . In assessing copyrightability, the district court is required to ferret out apparent expressive aspects of a work and then separate protectable expression from "unprotectable ideas, facts, processes, and methods of operation."[370]

However, while the Federal Circuit gets these fundamentals correct, it ultimately is confused about the difference between software and other types of works that,

---

367 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 818 (1st Cir. 1995). *See supra* notes 208–12 and accompanying text.

368 In *Google (2014)*, the Federal Circuit cites to the precise statement in *Altai* that we pointed out earlier is *not correct* and that the Second Circuit itself did not apply in the *Altai* decision. Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1355 (Fed. Cir. 2014); *see* Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 707–08 (2d Cir. 1992). *See also supra* notes 330–52 and accompanying text.

369 *See* § 102(b); *see* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1356 (Fed. Cir. 2014).

370 *Oracle Am., Inc.*, 750 F.3d at 1356–57.

while seeming like software in that they may be considered hybrid-like, are not software and, thus, are not clearly excluded from the form-function doctrine.[371] In particular, the Federal Circuit states:

> When assessing whether the non-literal elements of a computer program constitute protectable expression, the Ninth Circuit has endorsed an "abstraction-filtration-comparison" test formulated by the Second Circuit and expressly adopted by several other circuits . . . This test rejects the notion that anything that performs a function is necessarily uncopyrightable . . . And it also rejects as flawed the *Whelan* assumption that, once any separable idea can be identified in a computer program everything else must be protectable expression, on grounds that more than one idea may be embodied in any particular program.[372]

Thus, the Federal Circuit believes that by adopting the *Altai* approach, the Ninth Circuit has necessarily rejected the approach of *Lotus v. Borland*.[373] The Federal Circuit specifically states, in this regard: "This test rejects the notion that anything that performs a function is necessarily uncopyrightable."[374] However, that is not the complete story.

Both *Altai* and *Lotus* appreciate that AFC is restricted to computer software.[375] That is *not* the same as rejecting the notion that anything that performs a function is necessarily uncopyrightable. *Rather, the form-function doctrine is rejected for software based on Congressional intent.*[376] This seems to be where courts get tripped up the most. Here, even the *Google (2014)* court is confused, as its explanation shows. This confusion becomes even more apparent when the appellate court addresses the SSO of Oracle's code, but ultimately, like in *Altai*, though making some confusing statements, the Federal Circuit in *Google (2014)* still applies the law correctly in the situation before it.[377]

Having rejected section 102(b) and concluded that merger and scène à faire affect the scope of protection rather than copyrightability, the Federal Circuit moves on to resolve the district court ruling about the declaring code.[378]

---

371 *See supra* notes 13, 92–112, 345 and accompanying text.

372 *Oracle Am., Inc.*, 750 F.3d at 1357–58.

373 *See id.* at 1357.

374 *Id.* (citing Mitel. v. Iqtel, 124 F.3d 1366 (10th Cir. 1997)). *Mitel is* not a software case, as such, possibly explaining at least partly the basis for its confusion. Rather, *Mitel* concerns copyright protection for a selection of particular four-digit numbers that activate and manipulate the functions of a Smart-1 call controller.

375 Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 714 (2d Cir. 1992); *see* Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 813 (1st Cir. 1995).

376 *See supra* notes 92–112 and accompanying text.

377 *Oracle Am., Inc.*, 750 F.3d at 1356.

378 *Id.* at 1358–59.

Importantly, the Federal Circuit can correctly separate merger from the form-function doctrine and analyze the district court's position regarding merger.[379] Likewise, here, the Federal Circuit's analysis is spot on. It works through several analogies with the declaring code, first comparing it to the introduction for *A Tale of Two Cities* and then, even more appropriately, to a compilation.[380] Thus, the Federal Circuit concludes there is creativity in the arrangement and selection of the declaring code.[381] Furthermore, since there are a variety of possible arrangements, merger does not apply.[382]

As to scène à faire, the record was not developed below and the lower court held that doctrine inapplicable.[383] Here, the appellate court agrees.[384] However, the appellate court suggests that merger and scène à faire are not as relevant to interoperability where, as here, interoperability was not a concern facing the author of the allegedly *infringed* code.[385] While the appellate court applies appropriate legal reasoning, it may still overstate its position ever so slightly. The appellate court's logic is impeccable; however, it nonetheless fails to consider the atypical,[386] but possible situation. Suffice it to say that such a situation was not before the Federal Circuit. Unfortunately, its statement regarding interoperability, taken out of context, has the potential to be interpreted overly broadly.[387]

The Federal Circuit's confusion about the appropriate legal analysis becomes even more apparent when it tackles the question regarding copyrightability and scope of protection for the SSO of Oracle's overall naming taxonomy.[388] With that said, the issues here are complex and the differences quite subtle, perhaps even more subtle than courts such as *Lotus* and *Altai* appreciated. Although those courts correctly recognized that software, audiovisual displays, and

---

379 *Id.* at 1359.

380 *See also supra* note 317–18 and accompanying text (discussing this helpful analogy in more detail with respect to software).

381 *Oracle Am., Inc.*, 750 F.3d at 1359–62.

382 *Id.*

383 *Id.* at 1363.

384 *See id.*

385 *See id.* at 1368–72.

386 *See supra* note 318–46 and accompanying text (discussing a situation the court may have overlooked).

387 Some may interpret the *Google (2014)* court to suggest that, with respect to merger and scène à faire, interoperability could *never* be relevant to the scope of protection for code and/or that interoperability could *never* be relevant to the scope of protection "after the fact," so to speak, as opposed to interoperability considerations facing a programmer at the time of creation of a program. *See Oracle Am., Inc.*, 750 F.3d at 1370–71. *See also supra* notes 330–60 and accompanying text (discussing this situation in more detail).

388 *Oracle Am., Inc.*, 750 F.3d at 1361–62.

command hierarchies constitute separate categories of copyrightable works.[389]

With that said, perhaps, another perspective might be that these different works are related.[390] For example, code/software may relate to a command structure or to an audiovisual display. Various works that are not technically software, such as screen displays, file structures, and/or menu command hierarchies, for example, might be related to software by being described as hybrid-like in the sense that this article has been using this term. For example, these works potentially include copyrightable expression in a manner so that the expressive elements may not be separable from the utilitarian elements, much like software.[391]

Regardless, the Federal Circuit's discussion seems completely numb to such subtle distinctions and, hence, evidences some amount of confusion, at least as to the salient legal considerations. In particular, the court's stated basis for distinguishing *Lotus*, while not entirely incorrect, could have been much stronger. The appellate court distinguishes *Lotus* on two bases. First, on the facts and, second, as bring rejected under Ninth Circuit law.[392] Thus, regarding the facts, the appellate court states:

> On appeal, Oracle argues that the district court's reliance on *Lotus* is misplaced because it is distinguishable on its facts and is inconsistent with Ninth Circuit law. We agree. First, while the defendant in *Lotus* did not copy any of the underlying code, Google concedes that it copied portions of Oracle's declaring source code verbatim. Second, the *Lotus* court found that the commands at issue there (copy, print, etc.) were not creative, but it is undisputed here that the declaring code and the structure and organization of the API packages are both creative and original. Finally, while the court in *Lotus* found the commands at issue were "essential to operating" the system, it is undisputed that—other than perhaps as to the three core packages—Google did not need to copy the structure, sequence, and organization of the Java API packages to write programs in the Java language.[393]

The distinction made by the Federal Circuit in its 2014 *Googl*e decision that the commands in *Lotus* were not creative, while the structure and organization of

---

389 *See supra* notes 70–77 and accompanying text.

390 The text of the article provides a *great* oversimplification but is offered to at least help in articulating an intuition, which appears implicit in several decisions, that these separate works are related in some way intellectually. Unfortunately, this intuition may have done more to confuse courts than to aid them. In the third section of this article, we attempt to articulate, albeit imperfectly, how these works may be thought of as being related and "hybrid." *See supra* Section III.

391 *Id.*

392 *Oracle Am., Inc.*, 750 F.3d at 1365–66.

393 *Id.* at 1365.

the API packages were creative, is questionable.[394] However, it is correct that the cases are factually distinguishable. In *Google (2014)*, code was copied, both literally and arguably non-literally.[395] In *Lotus*, however, a command hierarchy, rather than code, was copied.[396] The *Google (2014)* court also treats the finding in *Lotus* that the commands at issue were "essential to operating" the system as a factual distinction.[397]

Thus, the Federal Circuit overlooks that the two cases, while raising highly analogous factual situations in one sense, nonetheless, more formally raise significantly different legal issues because one case involved copying software and the other did not. Notably, however, despite this, the Federal Circuit correctly concludes that the form-function doctrine does not apply to computer programs stating quite clearly:

> [T]he court concluded that, although the SSO is expressive, it is not copyrightable because it is also functional. The problem with the district court's approach is that computer programs are by definition functional—they are all designed to accomplish some task. Indeed, the statutory definition of "computer program" acknowledges that they function "to bring about a certain result." . . . If we were to accept the district court's suggestion that a computer program is uncopyrightable simply because it "carr[ies] out pre-assigned functions," no computer program is protectable. That result contradicts Congress's express intent to provide copyright protection to computer programs, as well as binding Ninth Circuit case law finding computer programs copyrightable, despite their utilitarian or functional purpose.[398]

However, the Federal Circuit fails to appreciate the legal significance of that distinction with respect to *Lotus.* *Google (2014)* is about copying software, not about copying a command hierarchy.[399] More specifically, the form-function doctrine applies to a command structure, in the absence of Congressional action or a clear legal basis for extending the exemption of the form-function doctrine from software to a command structure.[400]

Failing to appreciate that distinction from *Lotus*, the Federal Circuit, instead, concludes that Ninth Circuit law rejected *Lotus*.[401] The Federal Circuit offers

---

394 *Id.* at 1364–65.
395 *Id.* at 1356.
396 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 814 (1st Cir. 1995).
397 *Oracle Am., Inc.*, 750 F.3d at 1365.
398 *Id.* at 1367.
399 *Id.*
400 *See supra* notes 70–77, 208–12, 347–55 and accompanying text.
401 *Oracle Am., Inc.*, 750 F.3d at 1365.

no clear rejection by the Ninth Circuit of *Lotus* itself and, rather, acknowledges that the Ninth circuit really has not spoken much about *Lotus.* [402] However, it concludes that Ninth Circuit law must be inconsistent with *Lotus* since Ninth Circuit law recognizes that the SSO of a program may be copyrightable where it qualifies as expression, citing to *Johnson Controls v. Phoenix Control Systems,*[403] and to its own decision, *Atari Game Corp. v. Nintendo if America, Inc*.[404] However, there is nothing in either case that suggests a rejection of *Lotus* expressly or implicitly.[405]

To be more specific, although the point may be a subtle one, there is nothing about accepting the copyrightability of the SSO of a computer program that necessarily is inconsistent with the holding in *Lotus* regarding the copyrightability of a command hierarchy. The decisions concern two different types of works, granted that the works do have some potential similarities, notably at least in part because both are hybrid or hybrid-like.[406]

Finally, regarding *Google (2014),* in its discussion of interoperability, the Federal Circuit appropriately recites the language from *Apple v. Franklin* previously discussed.[407] Likewise, here, the Federal Circuit is correct that, as in *Apple*, while it is true that copyright law is to strike a balance between competition and protection, such considerations do not necessarily implicate merger, which wholly depends on the expressions at issue. [408]

On April 5, 2021, the United States Supreme Court rendered its decision in *Google v. Oracle*.[409] The Supreme Court reversed the 2018 Federal Circuit decision finding liability, but did so on grounds other than the topic of this article.[410] The Supreme Court decided that the defense of fair use applied as a matter of law to prevent Google from being liable for copyright infringement.[411]

---

[402] *Id.* at 1365 n.11 ("As Oracle points out, the Ninth has cited *Lotus* only one time on a procedural issue.").

[403] *See* Johnson Controls v. Phoenix Control Sys., 886 F.2d 1173, 1176 (9th Cir. 1989) ("Whether the nonliteral components of a program, including the structure, sequence and organization and user interface, are protected depends on whether, on the particular facts of each case, the component in question qualifies as an expression of an idea, or an idea itself.").

[404] *Oracle Am., Inc.*, 750 F.3d at 1365. *See* Atari Games Corp. v. Nintendo of Am., Inc., 975 F.2d 832, 837 (Fed. Cir. 1992). This provides some irony since, the Federal Circuit, in part, *relies on its own interpretation of Ninth Circuit law* in, *Atari, to conclude that Ninth Circuit law has rejected Lotus.*

[405] *See generally Johnson Controls*, 886 F.2d at 1173; *Atari Games Corp*., 975 F.2d at 832.

[406] *See supra* notes 70–77, 208–12, 347–55 and accompanying text.

[407] *See supra* notes 202–12 and accompanying text.

[408] *Oracle Am., Inc*., 750 F.3d at 1359; Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1253 (3rd Cir. 1983). *See supra* notes 202–12 and accompanying text.

[409] Google LLC, v. Oracle Am., Inc., 141 S. Ct. 1183 (2021).

[410] *Id.* at 1187.

[411] *Id.* at 1209.

Therefore, as a legal matter, the 2014 Federal Circuit decision in *Google v. Oracle* stands.

In a situational analogy, the Supreme Court result in *Google v. Oracle* might be compared with another Supreme Court fair use case also presenting a type of high water mark at the time of its decision—*Sony v. Betamax*.[412] In a 5-4 decision, the majority found fair use applicable and the dissent provided a rather scathing treatment pointing out the inconsistencies with prior fair use law in the majority's pronouncement.[413] Here, the same might be said of *Google v. Oracle*. Likewise, here, as before, the dissent's spot-on criticisms will not prevent the Court's pronouncement from becoming the law of the land.

Our interest, however, is not so much the fine contours of fair use law, which is beyond the scope of this article, but instead, to deduce what we are able from the Court's opinion relevant to the topic at hand. That is, how does the Court's ruling and analysis impact our thesis about a hybrid protection regime for software that differs from the traditional copyright protection regime? There can be little doubt that this recent decision goes far toward confirming a hybrid protection regime for software. To be more specific, the Court appears to have carved out a special area for software at least with respect to fair use, not unlike has been done for parody in connection with fair use.[414] To quote the Court:

> The fact that computer programs are primarily functional makes it difficult to apply traditional copyright concepts in that technological world . . . In doing so here, we have not changed the nature of those concepts. We do not overturn or modify our earlier cases involving fair use—cases, for example, that involve "knockoff" products, journalistic writings, and parodies. Rather, we here recognize that application of a copyright doctrine such as fair use has long proved a cooperative effort of Legislatures and courts, and that Congress, in our view, intended that it so continue. As such, we have looked to the principles set forth in the fair use statute, [Section]107, and set forth in our earlier cases, and applied them to this different kind of copyrighted work. [415]

However, to start in a more analytically convenient spot, early in the opinion, the Court states:

> We shall assume, but purely for argument's sake, that the entire Sun Java API falls within the definition of that which can be copyrighted. We shall ask instead whether Google's use of part of that API was a

---

412 *See generally* Sony Corp. of Am. v. Universal City Studios, Inc., 464 U.S. 417 (1984).

413 *Id.*

414 *Google LLC*, 141 S. Ct. at 1208–09.

415 *Id.*

"fair use." Unlike the Federal Circuit, we conclude that it was. [416]

Thus, the Supreme Court chose not to specifically address the issues of copyrightability and scope of protection as explained by the Federal Circuit in its 2014 decision,[417] just analyzed above in detail. Despite being based on fair use, this outcome, at least indirectly (and as shown below, arguably directly), supports the notion of a hybrid protection regime for software that impacts scope of protection and may "spill over" into copyright law defenses.[418]

At a high level, the decision represents a dramatic shift in the balance between protection and competition for software ostensibly due to its utilitarian aspects.[419] In this regard, the decision represents yet another cut back on how copyright law provides protection for software, a trend in the law at least since the *Altai* decision, suggesting two protection regimes (one for traditional copyrightable subject matter and one for software). [420] Here, as shall be explained, the Court concludes that so called "reimplementing" is transformative.[421] However, reimplementing is a quite common situation in litigation involving software. *Whelan* and *Altai* were essentially reimplementing cases in that the facts involve moving software to other, newer technology platforms that resulted in more overall software expression for society; however, these situations allowed for findings of liability where copying occurred.[422]

In one sense, as indicated by the Federal Circuit's 2018 decision, this case might be viewed as nothing more than an ordinary code copying case. [423] The Court itself quotes the Federal Circuit as follows:

> Deciding that question of law, the court held that Google's use of the Sun Java API was not a fair use. It wrote that "[t]here is nothing fair about taking a copyrighted work verbatim and using it for the same purpose and function as the original in a competing platform."[424]

However, in another sense, in this case the Court comes to terms with the notion that the fundamental nature of software may call for a different set of rules under copyright.[425] That is, a major shift in the balance between competition and

---

416 *Id.* at 1197.

417 Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1354 (Fed. Cir. 2014).

418 *See Google LLC*, 141 S. Ct. at 1188 (explaining that the software scope of protection that is *ultimately* provided by law is affected).

419 *See id.* at 1197.

420 *See* Johnson Controls, Inc. v. Phoenix Control Sys., Inc., 886 F.2d 1173, 1176 (9th Cir. 1980).

421 *Id.*

422 *See* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1246 (3d Cir. 1986); *see also* Comput. Assocs. Int'l Inc. v. Altai Inc., 982 F.2d 693, 720–21 (2d Cir. 1992).

423 Oracle Am., Inc. v. Google LLC, 886 F.3d 1179, 1179 (Fed. Cir. 2018).

424 *Google LLC*, 141 S. Ct. at 1195 (emphasis added) .

425 *Id.* at 1202.

protection is appropriate.[426]

To do this, the Court brilliantly uses fair use, which permits considering factors that otherwise may be less legally appropriate for scope of protection under a *prima facie* case, such as form-function issues in the context of software.[427] In general, a highly refined analysis characterizes this treatment under fair use, particularly with respect to the second factor, the nature of the copyrighted work.

Recall that fair use is an equitable doctrine, and each case is decided on its own facts.[428] However, here, the Court issued a decision as a matter of law, giving the decision more significant precedential impact.[429] A key element of the analysis, of course, is the nature of the work, although the conclusion that reimplementing is transformative is also important to the analysis. By way of contrast, outside of fair use, recall that copyright jurisprudence suggests courts should not inquire into levels of creativity.[430] An entirely legitimate question, of course, is for software how well a federal court may be equipped to conduct such an analysis which involves not only questions of creativity, but also questions regarding refined technical analysis.[431]

From a policy perspective, for reasons previously discussed[432] and discussed in more detail immediately below, the direction of the decision appears correct and even consistent with Congressional intent. That is, in the Court's view, market forces are sufficient to incent *close* to the appropriate amount of software for society. This policy shift is particularly evident in the Court's treatment of the fourth factor,[433] the effect of the use upon the potential market for, or value of, the copyrighted work, by including the public benefits of copying in its analysis. The amount of protection needed through legal process is not nearly so great for software as in other areas of copyright law. In this sense, if this supposition is correct, the Court is being true to Congressional intent even while significantly weakening copyright protection for software.

---

426 *See* Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (2d Cir. 1971).

427 *Google LLC*, 141 S. Ct. at 1195.

428 *See, e.g.*, *id.* at 1197.

429 *Id.* at 1209.

430 *See, e.g.*, Burrow-Giles Lithographic Company v. Sarony, 111 U.S. 53 (1884); Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1251–54 (3d Cir. 1983).

431 *Cf. Google LLC*, 141 S. Ct. at 1215–16 (Thomas, J. dissenting).

432 *See supra* notes 4, 156–60 and accompanying text.

433 *See Google LLC*, 141 S. Ct. at 1206–07 (stating regarding the fourth factor: "Further, we must take into account the public benefits the copying will likely produce. Are those benefits, for example, related to copyright's concern for the creative production of new expression? Are they comparatively important, or unimportant, when compared with dollar amounts likely lost?"). *Cf.* MCA, Inc. *v.* Wilson, 677 F. 2d 180, 183 (2d. Cir. 1981) (calling for a balancing of public benefits and losses to the copyright owner under this factor).

Under fair use, with this possible realization in mind, if there are doubts regarding applicability of particular fair use factor, perhaps, for software, such doubts are now to be resolved in favor of concluding the factor is met in favor of finding fair use, including, how that factor may interact with the other factors to bring about a finding of fair use, analogous to placing a finger on the scale. One might suspect this occurs here because the Court is reinstating a jury decision; however, because the Court resolves the issue as a matter of law, this does not completely justify the decisional result. Instead, the result appears to follow from the nature of the work and the policy considerations just discussed. Rather, as stated, this is a shift in the balance of protection and competition. Regarding the four fair use factors, in general, the Court breaks with traditional approaches under fair use.[434]

If the policy justification above is accepted as true, then the road map to reach the conclusion of the Court is reasonably straight-forward. As to the nature of the work, the Court states that it is functional and utilitarian, making it further from the core of copyright.[435] According to the Court, the declaring code is *even further* from that core than the implementing code.[436] The Court apparently sees the implementing code more like typical code.[437] The Court, in contrast, sees the declaring code as bound up with uncopyrightable ideas[438] and so the Court relies heavily on fair use cases of so called "intermediate copying." Intermediate copying refers to a situation where code is copied, but only to be able to discern the underlying uncopyrightable ideas.[439]

In general, the Court sees less protection for software because it is functional and utilitarian, consistent with the notion of a different protection regime being applicable to other types of copyrightable subject matter; however, as to the declaring code, in particular, the Court sees that code as barely more than an

---

434 The Court concludes the purpose and character of the use is transformative; however, the way the Court treats transformative use is unique here because the code itself is not changed. Rather, the copied code is integrated with other code. The Court likewise handles the market effect differently than precedent. Oracle was licensing the software and lost significant revenue from licenses like adobe. *See* Hartiens, *supra* note 333, at 10–11. However, the court, unlike in other works involving fair use, does not consider the lost licensing opportunity a significant issue under this factor. Instead, the court considers whether Oracle sought to enter this market and whether it was successful. *See also Google LLC*, 141 S. Ct. at 1215–20 (the dissent's analysis of the four factors).

435 *Id.* at 1201–02.

436 *Id.*

437 *Id.*

438 *Id.*

439 *See* Sony Comput. Ent., Inc. *v.* Connectix Corp., 203 F. 3d 596, 603–08 (9th Cir. 2000) (applying fair use to intermediate copying necessary to reverse engineer access to unprotected functional elements within a program); Sega Enter. Ltd. v. Accolade, Inc., 977 F. 2d 1510, 1521–27 (9th Cir. 1992) (holding that wholesale copying of copyrighted code as a preliminary step to develop a competing product was a fair use).

idea, leading it to make comparisons with *Feist* and intermediate copying.[440]

The Court, unfortunately, however, fails to clearly discern the difference between form-function considerations and scope of protection considerations.[441] This, of course, is not surprising since courts generally do not seem to appreciate the difference explicitly, although several seem to do so implicitly.[442] Here, for example, the dissent also seems to perceive the difference at least implicitly.[443] However, the majority seems numb to this subtle point.

The Court says the declaring code is bound up with uncopyrightable ideas,[444] but what is the nature of those so called "uncopyrightable ideas?" In general, they appear to relate to use. Specifically, the Court states:

> The declaring code (inseparable from the programmer's method calls) embodies a different kind of creativity. Sun Java's creators, for example, tried to find declaring code names that would prove intuitively easy to remember . . . They wanted to attract programmers who would learn the system, help to develop it further, and prove reluctant to use another . . . Sun's business strategy originally emphasized the importance of using the API to attract programmers. It sought to make the API "open" and "then . . . compete on implementations." . . . The testimony at trial was replete with examples of witnesses drawing this critical line between the user-centered declaratory code and the innovative implementing code . . .
>
> These features mean that, as part of a user interface, the declaring code differs to some degree from the mine run of computer programs. Like other computer programs, it is functional in nature. But unlike many other programs, its use is inherently bound together with uncopyrightable ideas (general task division and organization) and new creative expression (Android's implementing code). Unlike many other programs, its value in significant part derives from the

---

440 *See* Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 111 S. Ct. 1282, 1290 (1991); *see also Sony Comput. Ent., Inc.*, 203 F. 3d at 603–08 (applying fair use to intermediate copying necessary to reverse engineer access to unprotected functional elements within a program); *Sega Enter. Ltd.*, 977 F. 2d at 1521–27 (holding that wholesale copying of copyrighted code as a preliminary step to develop a competing product was a fair use).

441 This distinction has been previously discussed. One (scope of protection) is a balancing that courts perform so that, for a particular type of copyrightable subject matter, copyright law is neither overprotective nor under protective. *See supra* notes 31–39 and accompanying text. However, the form-function distinction relates to whether providing copyright protection would result in providing use rights to the copyright holder. Google LLC, v. Oracle Am., Inc., 141 S. Ct. 1183, 1202 (2021). *See supra* notes 75–93 and accompanying text.

442 *See supra* notes 207–12 and accompanying text.

443 *Google LLC*, 141 S. Ct. at 1213–14 (Thomas, J. dissenting).

444 *Id.* at 1201–02.

> value that those who do not hold copyrights, namely, computer programmers, invest of their own time and effort to learn the API's system. And unlike many other programs, its value lies in its efforts to encourage programmers to learn and to use that system so that they will use (and continue to use) Sun-related implementing programs that Google did not copy.[445]

Is not this aspect simply related to use rights? Of course, the genius in the Court's decision as stated is the ability under the guise of fair use to bring in considerations less legally appropriate for scope of protection.

Whether you agree with the Court's second factor analysis, the analysis of this factor permeates the other fair use factors. This explains the decision and the language from decision that suggests the Court has carved out a special area of fair use law. Treatment for software under fair use is now quite different. A different set of underlying assumptions is applied arising out of the utilitarian nature of software. Software, in general, is assumed to be less creative. Likewise, the Court provides further leeway to parse creativity quite finely, as was done for the declaring code in comparison with the implementing code. One should ask, however: should a court now do a line-by-line analysis of code that may be at issue? One might also ask: is a court generally capable of such a fine technical analysis?

Putting aside this aspect of the Court's decision, the opinion is easily constructed from the following points:

(1)Software is functional and useful;

(2)The declaring code is barely more than an idea;

(3)Re-implementation of software is transformative; and

(4)A shift in the balance between competition and protection is appropriate for software.

With these considerations, the first two factors, "the purpose and character of the use," and "the nature of the copyrighted work," easily weigh in favor of fair use. Furthermore, these considerations then bleed into the analysis of the third and fourth factor, "the amount and substantiality of the portion used in relation to the copyrighted work as a whole" and "the effect of the use upon the potential market for, or value of, the copyrighted work."[446]

---

445 *Id.*

446 *Google LLC*, 141 S. Ct. at 1204–05 ("The [third factor,] 'substantiality' factor will generally weigh in favor of fair use where, as here, the amount of copying was tethered to a valid, and transformative, purpose."); *see* Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 586–87 (1994) (explaining that the three factor "enquiry will harken back to the first of the statutory factors, for . . . the extent of permissible copying varies with the purpose and character of the use"); *Google LLC*, 141 S. Ct. at 1206–07 (stating regarding the fourth factor: "Further, we must take into account the public benefits the copying will likely produce. Are those benefits, for example, related to copyright's concern for the creative

Much more could be said about the Court's analysis of the fair use factors, but that task is left to others.

However, before leaving discussion of this groundbreaking case, a feature is highlighted here that also breaks from traditional treatment and might be a hallmark for treatment of software under copyright law. The Court's analysis is not only quite fine, but also *measured*. That is, the Court does not employ an all or nothing type approach.[447] This approach tends to benefit the society because it does not totally vitiate the incentive to produce a work.

The *Lotus v. Borland* case (also referred to herein as *Lotus)* from the First Circuit came up in a back-handed way in some earlier discussion. It is discussed, now, out of chronological order, primarily because it is not a software copyright case, as a technical matter, but remains important in this area.[448] The *Lotus* court itself states: "In the instant appeal, we are not confronted with alleged nonliteral copying of computer code."[449]

The case is unusual in several respects. First, the question, which is recognized as one of first impression is: "whether a computer command hierarchy constitutes copyrightable subject matter."[450] That is, as stated by the court, "standing on its own (i.e., without other elements of the user interface, such as screen displays, in issue)."[451] The court, therefore, recognizes that they "are navigating in uncharted waters."[452] Furthermore, the case is exclusively focused on the question of copyrightability. As the court states: "Borland concedes that Lotus has a valid copyright in Lotus 1-2-3 as a whole and admits to factual copying of the Lotus menu command hierarchy."[453] The court also notes, regarding software copyright cases: "Because of this different posture, most copyright-infringement cases provide only limited help to us in deciding this appeal. This is true even with respect to those copyright-infringement cases that deal with computers and computer software." [454]

The First Circuit goes on to acknowledge *Altai*, but finds it of no help:

---

production of new expression? Are they comparatively important, or unimportant, when compared with dollar amounts likely lost?"); *Cf.* MCA, Inc. v. Wilson, 677 F. 2d 180, 183 (2d Cir. 1981) (calling for a balancing of public benefits and losses to copyright owner under this factor).

447 *Google LLC*, 141 S. Ct. at 1199 ("We do not believe that an approach close to 'all or nothing' would be faithful to the Copyright Act's overall design."); *see infra* notes 579–82 and accompanying text.

448 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 814–15 (1st Cir. 1995).

449 *Id.* at 814.

450 *Id.* at 813.

451 *Id.*

452 *Id.*

453 *Id.*

454 *Id.*

> While the *Altai* test may provide a useful framework for assessing the alleged nonliteral copying of computer code, we find it to be of little help in assessing whether the literal copying of a menu command hierarchy constitutes copyright infringement. In fact, we think that the *Altai* test in this context may actually be misleading because, in instructing courts to abstract the various levels, it seems to encourage them to find a base level that includes copyrightable subject matter that, if literally copied, would make the copier liable for copyright infringement. While that base (or literal) level would not be at issue in a nonliteral-copying case like *Altai*, it is precisely what is at issue in this appeal. We think that abstracting menu command hierarchies down to their individual word and menu levels and then filtering idea from expression at that stage, as both the *Altai* and the district court tests require, obscures the more fundamental question of whether a menu command hierarchy can be copyrighted at all. The initial inquiry should not be whether individual components of a menu command hierarchy are expressive, but rather whether the menu command hierarchy as a whole can be copyrighted.[455]

Finally, the case is unusual in light of subsequent events following this First Circuit decision. C*ertiorari* was granted by the Supreme Court.[456] After hearing the case, the remaining justices on the Court split 4-4, which legally means that the appellate court decision below stands as ruling precedent.[457]

The First Circuit in *Lotus* takes a different stance regarding section 102(b) than the court in *Google (2014)*.[458] Without providing much legal support, the First Circuit states:

> Borland argues that the Lotus menu command hierarchy is uncopyrightable because it is a system, method of operation, process, or procedure foreclosed from copyright protection by 17 U.S.C. Section 102(b). Section 102(b) states: "In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work." Because we conclude that the Lotus menu command hierarchy is a method of

---

455 *Id.* at 815.

456 Lotus Dev. Corp. v. Borland Int'l, Inc., 515 U.S. 1191 (1995) (4-4 *per curiam decision*) ("Justice Stevens took no part in the consideration or decision of these motions and this petition.")

457 *Id.*

458 *Compare Lotus Dev. Corp*., 49 F.3d at 816 (arguing the Lotus menu command hierarchy is a method of operation without deciding whether it is copyrightable within the meaning of 17 U.S.C. § 102(b)), *with* Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1199 (2021) (arguing that the Copyright Act does not adopt an "all or nothing" approach.).

> operation, we do not consider whether it could also be a system, process, or procedure.[459]

Interestingly, the *Lotus* court clearly appreciates the problems that utility raises in this context. It states:

> The Lotus menu command hierarchy does not merely explain and present Lotus 1-2-3's functional capabilities to the user; it also serves as the method by which the program is operated and controlled . . . The Lotus menu command hierarchy is also different from the Lotus screen displays, for users need not "use" any expressive aspects of the screen displays in order to operate Lotus 1-2-3; because the way the screens look has little bearing on how users control the program, the screen displays are not part of Lotus 1-2-3's "method of operation." The Lotus menu command hierarchy is also different from the underlying computer code, because while code is necessary for the program to work, its precise formulation is not. In other words, to offer the same capabilities as Lotus 1-2-3, Borland did not have to copy Lotus's underlying code (and indeed it did not); to allow users to operate its programs in substantially the same way, however, Borland had to copy the Lotus menu command hierarchy. Thus the Lotus 1-2-3 code is not an uncopyrightable "method of operation."[460]

Even more directly, it states:

> Our holding that "methods of operation" are not limited to mere abstractions is bolstered by *Baker* v. *Selden.* In *Baker,* the Supreme Court explained that "the teachings of science and the rules and methods of useful art have their final end in application and use; and this application and use are what the public derive from the publication of a book which teaches them . . . The description of the art in a book, though entitled to the benefit of copyright, lays no foundation for an exclusive claim to the art itself. The object of the one is explanation; the object of the other is use. The former may be secured by copyright. The latter can only be secured, if it can be secured at all, by letters-patent." . . . Lotus wrote its menu command hierarchy so that people could learn it and use it. Accordingly, it falls squarely within the prohibition on copyright protection established in *Baker* v. *Selden* and codified by Congress in Section 102(b).[461]

Likewise, the First Circuit even points to the "useful article" provisions of the

---

459 *Lotus Dev. Corp.*, 49 F.3d at 816.
460 *Id.* at 815–16.
461 *Id.* at 816–17 (quoting Baker v. Selden, 101 U.S. 99, 104–05 (1878)).

Act with reference to a VCR hypothetical.[462] The reasoning employed by the court is largely correct with the exception that it is unclear why it felt the need to find an answer in section 102(b) rather than merely relying on *Baker v. Selden.* It appears as a distinction without a difference, at least for this case.

The concurrence provides additional insight, recognizing that "[u]tility does not bar copyright, . . . but it alters the calculus."[463] Likewise, the concurrence incisively states:

> While Congress said that computer programs might be subject to copyright protection, it said this in very general terms; and, especially in Section 102(b), Congress adopted a string of exclusions that if taken literally might easily seem to exclude most computer programs from protection. The only detailed prescriptions for computers involve narrow issues (like back-up copies) of no relevance here.[464]

Likewise, he further states:

> Congress has arguably recognized the tension and left it for the courts to resolve through the development of case law. And case law development is *adaptive*: it allows new problems to be solved with help of earlier doctrine, but it does not preclude new doctrines to meet new situations. . . [F]or me the question is not whether Borland should prevail but on what basis. Various avenues might be traveled, but the main choices are between holding that the menu is not protectable by copyright and devising a new doctrine that Borland's use is privileged. No solution is perfect and no intermediate appellate court can make the final choice. [465]

The concurrence makes it clear that he sees section 102(b) as merely a means to an end, perhaps not fully appreciating that reliance on *Baker* should be sufficient.[466]

II. RECOGNIZING AND UNDERSTANDING THE PARADIGM SHIFT

A. Acceptance of AFC Approach

As has been recognized by courts, the center of this controversy over

---

462 *Id.* at 817.

463 *Id.* at 819 (Boudin, J., concurring).

464 *Id.* at 820 (Boudin, J., concurring). The concurrence, unfortunately, per the last statement of the paragraph above, fails to appreciate the significance of the definition of "computer programs" in 101 in conjunction with section 117. *See supra* Section I.B.

465 *Lotus Dev. Corp*., 49 F.3d at 820–21 (Boudin, J., concurring) (emphasis in original).

466 *Id.* (Boudin, J., concurring).

computer programs derives from its hybrid nature.[467] In fact, the *Altai* court stated as much:

> To be frank, the exact contours of copyright protection for non-literal program structure are not completely clear . . . . This results from the hybrid nature of a computer program, which, while it is literary expression, is also a highly functional, utilitarian component in the larger process of computing.[468]

It is well-recognized today that computer programs, as works, include elements that are both useful and expressive.[469] Returning to "first principles," however, in other areas of copyright law, when the expressive and useful elements of a copyrightable work are entangled in a manner in which it is not possible to satisfactorily disentangle them, copyright protection is generally deemed unavailable.[470] For the case of computer programs, given the clear intent of Congress to provide protection,[471] this previous approach does not apply. The *Altai* court may have stated it most succinctly: "Thus far, many of the decisions in this area reflect the courts attempt to fit the proverbial square peg in a round hole."[472]

In this context, then, the speed with which the abstraction-filtration-comparison approach was adopted by other courts is worthy of comment.[473] The

---

467 *See, e.g.*, Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 712 (2d Cir. 1992).

468 *Id.* (emphasis added).

469 *Id.* at 704 ("The essentially utilitarian nature of a computer program further complicates the task of distilling its idea from its expression."); *see A Manifesto Concerning the Legal Protection of Computer Programs*, *supra* note 6 at 2310; *see also* Samuelson, *supra* note 6, at 297; Menell, *supra* note 6, at 309; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1225; Armstrong, *supra* note 6, at 138; Vasilescu-Palmero, *supra* note 6, at 161; *The Uneasy Case for Software Copyrights Revisited*, *supra* note 6, at 1773–74; *Tailoring Legal Protection for Computer Software*, *supra* note 6, at 1369–70; Ginsburg, *supra* note at 6, at 2561; Dennis, *supra* note 6, at 58; *An Analysis of the Scope of Copyright Protection for Application Programs*, *supra* note 6, at 1072–73; Miller, *supra* note 6, at 983–84; Reichman, *supra* note 6, at 664–65; Weinreb, *supra* note 6 at 1150; Englund, *supra* note 6, at 891–92; Nimmer, *supra* note 6 at 644; NIMMER ON COPYRIGHT § 13.03(F), *supra* note 6; Johnathan Band & Masanobu Katoh, *Interfaces on Trial: Intellectual Property and Interoperability in the Global Software Industry*, 9 HARV. J. OF L. & TECH. 585, 588 (1996); INTERFACES ON TRIAL: INTELLECTUAL PROPERTY AND INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY, *supra* note 6, at 39–40; INTERFACES ON TRIAL 2.0, *supra* note 6. *See generally Special Issue: Software Interface Copyright*, *supra* note 6 (devoting the Harvard Journal of Law and Technology's Spring Issue to the topic of Copyright and Software Interfaces with 9 articles by well-known intellectual property law experts including: Pamela Samuelson, Peter Menell, and Mark Lemley).

470 *See supra* notes 31–93 and accompanying text.

471 *See supra* Section I.B.

472 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 712.

473 *Id.* at 705.

same year that the *Altai* decision was handed down, the Ninth Circuit adopted the approach.[474] Likewise, it was subsequently adopted by the First, Fifth, Sixth, Tenth, Eleventh, and the D.C. circuit, not to mention a host of district courts.[475] In general, circuits that have addressed the issue since *Altai* appear to agree that the *Altai* approach at least for software is more appropriate than a more traditional approach, exhibited, for example, in *Whelan*.[476] Although these courts may not have fully understood the intellectual difference or shift that occurred, they do appear to understand that the complexities of software call for an approach other than the one courts have typically applied to literary and other similar copyrightable works.

A possible message to be gleaned from this quick embracing of a new approach is that previously copyright law provided too much protection for software. The abstraction-filtration-comparison (AFC) approach provides less protection because major portions of a work may be filtered out prior to the substantial similarity comparison.[477] Thus, *Altai* provides a way to arguably remain reasonably consistent with prior copyright law, while reducing the scope of protection afforded computer software. Nonetheless, this approach makes it that much clearer that two separate regimes exist for addressing the scope of protection for copyrightable works.

B. Differences from Traditional Copyright Law

At a high-level, there are two major ways that analysis regarding the non-literal scope of protection for software differs from traditional copyright law. The first, and more significant difference, is that the 1980 amendments to the Act make it clear that even if copying is an essential step in the use (e.g., "utilization") of a program, that does not denigrate

---

474 *See* Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1524 (9th Cir. 1992).

475 *See* Kohus v. Mariol, 328 F.3d 848, 855 (6th Cir. 2003); Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 815 (1st Cir. 1995); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1525 (9th Cir. 1992); Eng'g Dynamics, Inc. v. Structural Software, 26 F.3d 1335, 1342 (5th Cir. 1994); Autoskill v. Nat'l Educ. Support Sys., 994 F.2d 1476, 1492 (10th Cir. 1993); Gates Rubber Co. v. Bando Chemical Industries, Ltd., 9 F.3d 823, 834 (10th Cir. 1993); Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1543–44 (11th Cir. 1996); Sturdza v. U.A.E., 281 F.3d 1287, 1296–97 (D.C. Cir. 2002). Furthermore, apparently the 6th, 10th and D.C. circuits have extended the *Altai* approach beyond software. *See* Godoy-Dalmau, *supra* note 101, at 248. Note also that, as occurred in *Oracle v. Google*, the Federal Circuit follows the law of the circuit in which the district court sits, but appears to approve of the approach. *See* Atari Games Corp. v. Nintendo of Am., Inc., 975 F.2d 832, 839 (Fed. Cir. 1992).

476 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 710; Whelan Assoc. Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1243 (3d Cir. 1986).

477 *Comput. Assocs. Int'l, Inc.*, 982 F.2d at 706.

the scope of protection afforded to software.[478] Thus, if the right to copy and the right to use are intertwined in such a way that they are not capable of being separated, this does not prevent computer software from being copyrightable and does not affect its scope of protection. In this article, we refer to this regime of protection as hybrid intellectual property rights in that the expressive elements and the utilitarian elements are both being protected.

A second major way the non-literal scope of protection for software differs from traditional copyright law relates to application of the filtration step. Under the AFC approach articulated by *Altai*, public domain elements are filtered out of the protected work. What makes this approach so different is that it, in effect, creates a type of irrebuttable presumption that similarities between the accused work and the protected work are not due to copying of the protected work, but rather copying from the public domain. To the extent there may be similarities between the protected work and works that are in the public domain, those similarities are irrefutably assumed to either not be original to the protected work and/or to have been copied by the accused work from the public domain. However, in other areas of copyright, particularly literary works, which is the analogy used most often with software, this type of filtration does not take place and, hence, no such irrebuttable presumption exists. [479]

1. Hybrid Nature

It a significant overstatement to suggest that AFC is entirely consistent with traditional copyright law principles. More specifically, under *Baker v. Selden*, *Mazer v. Stein*, and language of the Act related to useful articles,[480] for example, if traditional copyright principles were applied, it would follow that software is not protectible at all.[481] However, this

---

478 *Cf. supra* Section I.B.

479 *See supra* note 114 and accompanying text.

480 Baker v. Selden, 101 U.S. 99, 102–04 (1880); Mazer v. Stein, 347 U.S. 201, 217–19 (1954); *see* 17 U.S.C. § 101 (definition of pictorial, graphical and structural works'; definition of 'useful article'); 17 U.S.C. § 113. *See supra* note 126–27 and accompanying text.

481 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1367 (Fed. Cir. 2014) ("If we were to accept the district court's suggestion that a computer program is uncopyrightable simply because it 'carr[ies] out pre-assigned,' functions no computer program is protectable.."); *cf.* Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 820 (1st Cir. 1995) ("While Congress said that computer programs might be subject to copyright protection, it said this in very general terms; and, especially in Section 102(b), Congress adopted a string

result would be contrary to Congressional intent that computer programs be copyrightable along with the recognition in section 117 that copying may be an essential step in the utilization of a computer program.[482]

This difference alone makes software unique as a type of copyrightable expression and justifies use of the term hybrid intellectual property rights. More specifically, it is logically and practically impossible to disaggregate the expression in software from the aspects of the software that are useful. However, in traditional copyright law, if this occurs, then the expression is rendered unprotectible.[483] Not necessarily so for software, however.

It is noted that this issue seems to be the one that confuses courts the most and yet it is critical that it be understood to appropriately carry forward Congressional intent regarding the scope of protection to be afforded software. To be clearer, the doctrine of merger is often confused with the notion of functionality and form. The former *is* a limit on the scope of protection for software, whereas the latter *is not* a limit on the scope of protection for software.[484]

For example, the *Altai* court was confused on this point.[485] One source of confusion may be because *Baker v. Seldon* is the foundational case for both the merger doctrine and the form-function doctrine.[486] Thus, courts may, incorrectly, treat the two concepts as interchangeable, as *Altai* did.[487] They are separate doctrines, although in *Baker v. Seldon*, both were implicated by the same set of facts.[488]

Another source of confusion may be the First Circuit case, *Lotus v. Borland*, discussed in more detail *supra,* where the First Circuit held that the menu command hierarchy in Lotus 1-2-3 was uncopyrightable under

---

of exclusions that if taken literally might easily seem to exclude most computer programs from protection.") (Boudin, J., concurring).

482 17 U.S.C. § 117(a)(1); *see Oracle Am., Inc*., 750 F.3d at 1367 ("That result contradicts Congress's express intent to provide copyright protection to computer programs, as well as binding Ninth Circuit case law find computer programs copyrightable, despite their utilitarian or functional purpose."). *See also supra* notes 50–112 and accompanying text.

483 *Mazer*, 347 U.S. at 217–18.

484 *Compare* McKenna, *supra* note 64, at 531–35 (2017) (distinguishing between the idea-expression dichotomy and the useful articles doctrine), *with* Hickey, *supra* note 81, at 696 (treating merger and functionality as essentially the same doctrine). *See also supra* notes 50–112 and accompanying text.

485 Comput. Assocs. Int'l, Inc. v. Altai Inc., 982 F.2d 693, 708–09 (2d Cir. 1992).

486 *Baker*, 101 U.S. at 101–02.

487 *Comput. Assocs. Int'l, Inc.*, 982 F.2d at 708–09. *See supra* notes 330–52 and accompanying text.

488 *See supra* notes 54–87 and accompanying text.

section 102(b) as a "method of operation."[489] The First Circuit itself stated[490] that it was not considering software in that case.[491] Thus, in that court's view, which appears correct, the facts of that case did not fall into the software regime of hybrid intellectual property rights.[492] However, because the rights at issue were software-like, other courts may be confused and see that case as relevant.[493]

For example, although the analysis and holding in the 2014 Federal Circuit decision *Google v. Oracle* suggests a grasp of the difference between the doctrines of merger and form-function, the Federal Circuit nonetheless confusingly states in its decision: "a court must examine the software program to determine whether it contains creative expression *that can be separated* from the underlying function."[494] However, the expression cannot be separated from the underlying function. Furthermore, that has no bearing on the question of copyrightability or the scope of protection for software. This point is a dramatic and important one, which is why it has been emphasized here.

2. Filtration of Public Domain Elements

Another important difference between traditional copyright law and the so-called hybrid regime for software relates to the filtration step, which has been recognized as one of the more problematic aspects of the AFC approach.[495] To be more specific, in some respects, filtration of

---

489 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 815 (1st Cir. 1995).

490 *See supra* notes 459–67 and accompanying text.

491 *Lotus Dev. Corp.*, 49 F.3d at 815.

492 *Id.* at 816.

493 Comput. Assocs. Int'l, Inc. v. Altai Inc., 982 F.2d 693, 702–03 (2d Cir. 1992); *see* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1364–68 (Fed. Cir. 2014); *compare Lotus, Dev. Corp.*, 49 F.3d at 817 (holding on which the court recognized the difference between software and a user interface), *with* Eng'g Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335, 1351 (5th Cir. 1994) (holding in which the Fifth Circuit appeared to apply the AFC approach to input formats, user interfaces and output reports). However, note that some commentators do see *Lotus v. Borland* as relevant to the scope of protection for software, despite that it was addressing a menu command hierarchy rather than software. *See Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1228.

494 *Oracle Am., Inc.*, 750 F.3d at 1370 ("As previously discussed, a court must examine the software program to determine whether it contains creative expression *that can be separated* from the underlying function." (emphasis added)).

495 *See id.* at 1358 ("It is the second step of this analysis where the circuits are in less accord."). This step as a practical matter may be viewed as the most important step in terms of determining the scope of protection. *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 707

public domain elements is at odds with traditional copyright law principles, particularly for literary works, for example.

Accepting that, to determine scope of protection for a software copyright, filtration via the doctrines of merger and scène à faire is legitimate during an analysis of non-literal scope, and recognizing this has occurred in traditional copyright law cases as well[496], it is *not* typical in copyright law to assume that elements of a work that may be substantially similar to public domain elements *necessarily* originated from the public domain, particularly for a literary work, which is the category most frequently analogized to computer software. This fundamental point has been well-stated by the esteemed Judge Learned Hand, who observed in the 1936 case, *Sheldon v. Metro-Goldwyn Corp.*:

> We are to remember that *it makes no difference how far the play was anticipated by works in the public demesne which the plaintiffs did not use*. The defendants appear not to recognize this, for they have filled the record with earlier instances of the same dramatic incidents and devices, as though, like a patent, a copyrighted work must be not only original, but new . . . . Borrowed the work must indeed not be, for a plagiarist is not himself pro tanto an "author"; but if by some magic a man who had never known it were to compose anew Keats's Ode on a Grecian Urn, he would be an "author," and, if he copyrighted it, others might not copy that poem, though they might of course copy Keats's . . . .[497]

To understand a logical basis for filtering public domain elements, here, despite not doing so for literary works, consider, rather than literary works, perhaps, how courts handle works with pre-existing material, such as compilations and derivative works. In particular, for computer programs, some courts have spoken in terms of the structure, sequence, and organization ("SSO") of software, which appears to be analogous to the selection, coordination and arrangement of a compilation (i.e., the expression of a compilation subject to protection).[498] For example, for a compilation, the unprotected elements of the

---

("Strictly speaking, this filtration serves 'the purpose of defining the scope of plaintiff's copyright.'"); *see also* Hebl, *supra* note 292, at 142–46; Ocasio, *supra* note 97, at 314–17.

496 *See, e.g.*, Ferman v. Jenlis, Inc., 224 F. Supp. 3d 791, 803–06 (S.D. Iowa 2016) (applying merger); Walker v. Time Life Books, 784 F.2d 44, 50–51 (2d Cir. 1986) (applying scène à faire).

497 Sheldon v. Metro-Goldwyn Pictures Corp., 81 F.2d 49, 53–54 (2d Cir. 1936) (emphasis added).

498 *See Oracle Am., Inc.*, 750 F.3d at 1359–62; Whelan Assoc., Inc. v. Jaslow Dental Lab. Inc., 797 F.2d 1222, 1245 (3d Cir. 1986); Jon Wilkins, *Protecting Computer Programs as Compilations under Computer Associates v. Altai*, 104 YALE L.J. 435, 456–57 (1994). The compilation analogy has also been applied to architectural works. *See* Oravec v. Sunny Isles Luxury Ventures, L.C., 527 F.3d 1218, 1230 (11th Cir. 2008); Intervest Constr., Inc.,

compilation, which are pre-existing, are omitted or filtered in connection with a substantial similarity analysis.[499]

As another example, this filtering might be considered not unlike, the approach the Ninth Circuit employed in *Apple Computer v. Microsoft Corp.*[500] (hereinafter, *Apple v. Microsoft)* In that case, Microsoft had licensed some aspects of its GUI from Apple, but Apple believed that Microsoft's product exceeded the scope of the license. Because of the existing license agreement, to find infringement the appellate court required "virtual identity" to prove copyright infringement, a more restrictive test than the usual substantial similarity approach.[501] In other words, the existence of the license resulted in a recognition that certain aspects of similarity between features of the copyrighted work and features of the accused work would not be probative of copyright infringement.

However, a difference for a compilation, a derivative work or in the *Apple v. Microsoft* case, is that the elements were specifically known to not be probative of copyright infringement.[502] In such situations, it is factually clear that the pre-existing expression, if not omitted, would incorrectly contribute to a conclusion of substantially similarity between the two works, despite the pre-existing material having no bearing on whether copyright infringement actually took place.[503] One clear example, as suggested, is the pre-existing material of a derivative work.[504]

---

v. Canterbury Estate Homes, Inc., 554 F.3d 914, 919 (11th Cir. 2008). Architectural works are similar to software in that functionality may intersect with copyrightable expression. *See* Jonathan Seil Kim, note, *"Filtering" Copyright Infringement Analysis in Architectural Works*, 2018 U. ILL. L. REV. 281, 295–96 (2018); Lauren Jean Bradberry, *Putting the House Back Together Again: The Scope of Copyright Protection for Architectural Works*, 76 LA. L. REV. 268, 294–98 (2015).

499 *See* 17 U.S.C. § 103 ("The subject matter of copyright as specified by section 102 includes compilations and derivative works, but protection for a work employing preexisting material in which copyright subsists does not extend to any part of the work in which such material has been used unlawfully . . . . The copyright in a compilation or derivative work extends only to the material contributed by the author of such work, as distinguished from the preexisting material employed in the work, and does not imply any exclusive right in the preexisting material. The copyright in such work is independent of, and does not affect or enlarge the scope, duration, ownership, or subsistence of, any copyright protection in the preexisting material."). *See, e.g.*, Beryl R. Jones, *Copyright*: *Factual Compilations and the Second Circuit*, 52 BROOK. L. REV. 679, 682–84, 700–01 (1986); U.S. COPYRIGHT OFFICE, CIRCULAR NO. 14: COPYRIGHT IN DERIVATIVE WORKS AND COMPILATIONS (July 2020).

500 *Apple Comput., Inc. v. Microsoft Corp.*, 35 F.3d 1435, 1447 (9th Cir. 1995).

501 *Id.* at 1439.

502 *Id.* at 1447.

503 *Id.* at 1439.

504 *See* NIMMER ON COPYRIGHT § 2A.05(A)(2)(b), *supra* note 94 ("Originality operates as a threshold requirement for copyrightability and thus may, on occasion, deny protection to the work in question altogether; yet, in addition, the same doctrine also enters the picture

Here, however, it appears that this analytical step may be akin to creating an irrebuttable presumption that such elements of the protected work were copied from the public domain.[505] It is assumed because there may be similar elements in the public domain that those parts of the work are not original and, thus, are not available for copyright protection (and/or it is assumed that the accused work copied from the public domain rather than from the protected work).[506] This approach logically treats the software as if it were a type of compilation of well-known programming approaches. [507] However, an important question is whether such an approach is necessarily justified, in general, for software.

The question should be, as discussed by Learned Hand, how probative is the similarity that exists between the accused work and the protected work? As stated by Judge Hand in *Sheldon v. Metro-Goldwyn Pictures Corp.*,: "If the defendant has had access to other material which would have served him as well, his disclaimer becomes more plausible."[508] However, as a factual matter, in any given situation involving software, we do not know for sure whether the plaintiff's expression is original or if the defendant actually did have access to public domain materials that it copied. Perhaps, rather than creating an irrebuttable presumption, a burden shifting mechanism should be employed.

It is, of course, true that in the software field, certain well-known programming techniques may be proliferated throughout the community of

---

during the infringement analysis as part of the substantial similarity requirement, to eliminate non-original parts of a work from the comparison, when the work in question, considered as a whole, is deemed entitled to at least some protection.").

505 *See id.*

506 *See id.* ("Originality operates as a threshold requirement for copyrightability and thus may, on occasion, deny protection to the work in question altogether; yet, in addition, the same doctrine also enters the picture during the infringement analysis as part of the substantial similarity requirement, to eliminate non-original parts of a work from the comparison, when the work in question, considered as a whole, is deemed entitled to at least some protection.").

507 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1360 (Fed. Cir. 2014); Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1239 (3d Cir. 1986); Wilkins, *supra* note 499. The compilation analogy has also been applied to architectural works. *See* Oravec v. Sunny Isles Luxury Ventures, L.C., 527 F.3d 1218, 1230 (11th Cir. 2008); Intervest Constr., Inc. v. Canterbury Estate Homes, Inc., 554 F.3d 914, 920 (11th Cir. 2008). Architectural works are similar to software in that functionality may intersect with copyrightable expression. *See* Kim, *supra* note 499, at 286–87; Bradberry, *supra* note 499, at 268–69 (2015).

508 Sheldon v. Metro-Goldwyn Pictures Corp., 81 F.2d 49, 54 (2d Cir. 1936); *see also Whelan Assocs., Inc.*, 797 F.2d at 1232 n.23 ("Although not an issue in this case . . . it is important to note that even the showing of substantial similarity is not dispositive, for it is still open to the alleged infringer to prove that his work is an original creation . . . or that the similarities between the works was not on account of copying but because both parties drew from common sources that were part of the public domain. The cause of the substantial similarity—legitimate or not is a question of fact.").

software programmers.[509] But, in any given situation, it amounts to speculation whether this explains similarity between the works at issue. Perhaps, therefore, the burden should be placed on the defendant to demonstrate through proof that this explains the similarity and/or the burden should be placed on the plaintiff to show, for expression which may be similar to public domain expression, such expression is original to the plaintiff. In this way, rather than having an irrebuttable presumption that has the potential to significantly weaken a plaintiff's copyright in its software, it would, instead, be treated as a point subject to dispute and evidence between the parties via burden shifting.[510]

To take this point just a bit further, creating a type of irrebuttable presumption might logically be viewed as moving the originality requirement of copyright law closer to a novelty requirement (of the type that exists in patent law, for example). To illustrate this point, assume, for the sake of argument, that any time the public domain contains expression similar to expression in a copyrightable work, it is treated as if that expression is not original. In the logical extreme then, to be copyrightable, the work at issue would need to contain only novel expression. Such an approach, however, does not seem like an appropriate one, at least within the domain of copyright law.[511]

---

509 Various observers have noted that good programmers are "lazy" in that they do not recreate what already exists. *See* STEVEN WEBER, THE SUCCESS OF OPEN SOURCE 75 (2004) ("A good programmer is 'lazy like a fox.' Because it is so hard and time consuming to write good code, the lazy fox is always searching for efficiencies. . . . The last thing a programmer, particularly a volunteer programmer, wants to do is build from scratch a solution to a problem that someone else has already solved or come close to solving."); ERIC RAYMOND, THE CATHEDRAL & THE BAZAAR 24 (1999); LARRY WALL & RANDAL SCHWARZ, PROGRAMMING PERL (Tim O'Reilly ed., O'Reilly and Associates 1991).

510 For example, initially, the defendant could have the burden to show that some of the expression is present in the public domain. If the defendant met that burden, then the burden could shift to the plaintiff to show that those features are original to the plaintiff despite any similar public domain features. Likewise, if the plaintiff successfully meets its burden, then the burden could shift back to the defendant to show that the similarity in its accused work came from copying the public domain rather than copying from the plaintiff.

511 Some commentators have taken note about the potential overlap between protection provided via a software copyright and protection afforded through patent protection. *See* Samuelson, *supra* note 6 at 243; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1215; NIMMER ON COPYRIGHT§ 2A.07, *supra* note 16. Many key differences, however, exist between the scope of protection provided by a patent versus a copyright, which, of course, necessarily includes software. For a patent – *the scope of protection is defined by the claims, which are prepared by the applicant. See, e.g.*, Vitronics v. Conceptronic, Inc., 90 F.3d 1576, 1281–82 (Fed. Cir. 1996). For a copyright, *the scope of protection is determined by substantial similarity of the expression of a copyrightable work to an accused work. See, e.g.*, Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1544–45 (11th Cir., 1995). It would be extremely difficult for the scope of protection available for a computer program via copyright, including with respect to non-literal protection, to approach the scope of protection able to be garnered via patent protection. Thus, it is believed that concerns about overlap may be

Thus, in this regard, it may be that the *Altai* court, and the courts that have followed it, have gone just a little too far in limiting the scope of protection for software.[512] Instead, a potentially workable approach would permit a burden shifting mechanism to identify expression that might be original despite similar expression available from the public domain so that proof, rather than an assumption, ultimately resolves this issue in a case.

3. Merger and Scène à Faire

Likewise, while accepting that merger and scène à faire have been applied in traditional copyright law case; how those doctrines have been applied to software is appropriate for discussion here. As a first legal point, there is some disagreement among courts about whether those doctrines should be employed in connection with a copyrightability analysis or an infringement analysis.[513]

---

overstated and/or misplaced. A patent typically covers making, using, selling, offering for sale or importing tangible apparatuses (or processes), for example, that fall within the scope of the patent claims. S*ee* 35 U.S.C. § 271(a). A copyright covers copying (used in the broad, non-literal sense) – albeit, even considering non-literal scope, this at most relates to 'use' only to the extent that copying of a program indirectly includes use – and this applies only where there are a myriad of ways to accomplish the same task (e.g., absent merger). *See, e.g.*, *supra* notes 50–87 and accompanying text. We also observe that the intellectual property clause of the Constitution, Article 1, Section 8, Clause 8, gives Congress the power "to promote the Progress of Science and Useful arts, by securing, for limited Times, to Authors and Inventors, the exclusive Right to their respective Writings and Discoveries," and on its face has few clear limits imposed regarding this particular Congressional power other than that the protection be "for limited times" and be with respect to "writings" and "discoveries." Several Supreme Court cases have sought to construe various terms of this provision, such as the term "writings." U.S. CONST. art. 1 § 8, cl. 8; *See, e.g.*, Mazer v. Stein, 347 U.S. 201, 208–12 (1954); Feist Publ'ns, Inc. v. Rural Tel. Service Co., 499 U.S. 340, 363–64 (1991). Granting that there may be limits on the exercise of this Congressional power, either by logical construction and/or court decisions, it should be noted that what Congress may be unable to do under one of its powers, it may still be able to do under another power, such as under the interstate commerce power, for example. *Cf.* Trade-Mark Cases, 100 U.S. 82, 97 (1879). In this latter case, the Supreme Court held a federal trademark statute unconstitutional; however, afterwards, Congress re-enacted another federal trademark statute and made it clear it was relying on the interstate commerce power and the statute since then appears to have been accepted as constitutional. *See Mazer*, 347 U.S. at 206 n.5.

512 Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 721 (2d Cir. 1992).

513 *See* NIMMER ON COPYRIGHT § 2A.05(A)(2)(b), *supra* note 94 ("This approach treats the merger principle as one relating to the boundaries of permissible copying, rather than solely as a rule of copyrightability"); NIMMER ON COPYRIGHT § 2A.10(B)(4), *supra* note 93 (describing the view that merger is a defense to infringement as "[t]he better view."); Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1358 (Fed. Cir. 2014) (stating that in the 9th Circuit, "[w]hile questions regarding originality are considered questions of copyrightability, concepts of merger and scène à faire are affirmative defenses to claims of infringement."). *But see* Ocasio, *supra* note 97, at 311–12; *The Story of Baker v. Selden: Sharpening the Distinction Between Authorship and Invention*, *supra* note 97. *See also*

However, the better view seems to be that the appropriate place for these doctrines relates to analysis of substantial similarity, at least in connection with software.[514] Thus, the question of protectability is a separate question from scope of protection.

One source of confusion may relate to some courts' treatment of copyright law doctrines, such as the merger doctrine and the doctrine known as scène à faire, in connection with copyrightability.[515] That is, merger might apply to an entire work in question, such as a short phrase or title,[516] for example, which in some instances may be held to be uncopyrightable because the idea and the expression merge, so to speak. In contrast, for a piece of software, unless it is unusually short and compact, at best, some elements of the code might be subject to merger, such as a simple sorting routine, which may be sufficiently short that the expression and the idea merge. However, typically, one would expect this expression to constitute a feature or element of a much larger and complex set of expressions. Rather than rendering the overall work uncopyrightable, it may simply be one expressive element to omit in the analysis of substantial similarity that takes place for purposes of making an infringement determination.[517]

Likewise, the merger doctrine typically does not have the broad sweep suggested in *Altai*.[518] Rather, although the *Altai* court talks about efficiency regarding code or software, all of the court's merger examples *dealt with screen displays*[519] rather than software. This is especially interesting since the court

---

supra notes 31–86 and accompanying text.

514 NIMMER ON COPYRIGHT §2A.05(A)(2)(b), *supra* note 94 ("This approach treats the merger principle as one relating to the boundaries of permissible copying, rather than solely as a rule of copyrightability."); NIMMER ON COPYRIGHT §2A.10(B)(4), *supra* note 93 (describing the view that merger is a defense to infringement as "[t]he better view."); *Oracle Am., Inc*., 750 F.3d at 1358 (stating that in the 9th Circuit, "[w]hile questions regarding originality are considered questions of copyrightability, concepts of merger and scène à faire are affirmative defenses to claims of infringement."). *But see* Ocasio, *supra* note 97, at 321; *The Story of Baker v. Selden: Sharpening the Distinction Between Authorship and Invention*, *supra* note 97. *See also supra* notes 31–87 and accompanying text.

515 *Comput. Assocs. Int'l, Inc.*, 982 F.2d at 703, 706.

516 *See, e.g*., CIRCULAR 33, *supra* note 91; Ferman v. Jenlis, Inc., 224 F. Supp. 3d 791, 807 (S.D. Iowa 2016) (stating there is no protection for "No Trespassing" sign showing surveillance camera).

517 *See* NIMMER ON COPYRIGHT §2A.10(B)(2), *supra* note 93 (stating, in connection with merger, that "a given routine or component of the software may properly fall within the scope of merger"); NIMMER ON COPYRIGHT § 13.03(F)(2)(b), *supra* note 6 (discussing computer searching and sorting algorithms as potentially good examples of merger).

518 *See Comput. Assocs. Int'l, Inc.*, 982 F.2d at 708 ("Efficiency is an industry-wide goal. Since, as we have already noted, there may be only a limited number of efficient implementations for any given program task, it is quite possible that multiple programmers, working independently, will design the identical method employed in the allegedly infringed work. Of course, if this is the case, there is no copyright infringement.").

519 *See id.* at 708–09.

itself goes out of its way to distinguish screen displays from software. The court states early:

> As a caveat, we note that our decision here does not control infringement actions regarding categorically distinct works, such as certain types of screen displays. These items represent products of computer program, rather than the programs themselves, and fall under the copyright rubric of audiovisual works.[520]

Thus, the *Altai* court offers no real examples from prior decisions of situations in which merger has been applied to software.[521] One may also be inclined to criticize the *Altai* approach as encouraging the creation of less efficient, proprietary software except that it seems that merger should not typically arise much in connection with software, if properly applied.

Again, these doctrines relate to the scope of protection to be afforded the work in question. In other words, with merger, for example, was the author faced with a situation where there were only a limited number of ways of expressing an idea, at least as a practical matter? Perhaps, so. This seems to be the point made by the *Altai* court regarding efficiency.[522] That is, the court specifically explains that while there may be other ways to perform the task, but if a few ways are much more efficient than the others, then as a practical matter, there are only a limited number of ways, such as an extremely compact sorting process (mentioned above as a possible illustration) to perform the task.[523] If so, then merger applies, because to do otherwise would permit an author greater protection than copyright intends an author to have with respect to the expression at issue. Furthermore, if most programmers ultimately gravitate, so to speak, to these more efficient expressions, then similarity of such expressions may not be probative to show that the accused code had been copied from protected code. However, in general, it is expected that this will not be a frequent occurrence since there are more often many ways to code a particular task and it is expected that merger may be applicable to only one or a small set of expressive features to be filtered out of a larger and more complex overall expressive program. Perhaps, with respect to merger and software, courts need to engage in a special inquiry and make particularized findings. [524]

---

520 *Id.* at 703.

521 *Id.* at 708.

522 *Id.*

523 *See id.*

524 It is noted that although the text speaks as if this is straight-forward, in actual practice making this determination, especially for a court, may be quite a challenge. A court would need to be able to separate situations in which many alternate ways exist to accomplish a task, where those alternate ways are essentially interchangeable, from situations in which, though there may be many ways to accomplish a task, a small number of those ways are necessarily preferred as being more efficient. *See supra* notes 353–57 and accompanying

As a further illustration, consider Google's position regarding merger as to the declaring code in *Oracle v. Google* in comparison with the following hypothetical from Nimmer:

> As an illustration, consider an art textbook that seeks to teach its readers the methods of sketching flowers; it contains detailed pictorial illustrations and examples of the techniques to be employed in drawing and sketching objects. Assume that the techniques described in the book are such that the drawings and illustrations are essential to working them, but that the book as a whole contains sufficient original expression (*e.g.*, description of the techniques illustrated) so that denying it protection as a whole would be inappropriate. If someone were to purchase the book and copy its illustrations and examples—without any modification whatsoever—in her own sketching of flowers, the means/ends distinction tells us that this copying ought to be classified as non-infringing.[525]

Nimmer, referring to *Baker v. Seldon*, makes a distinction between copying for explanation versus copying for use.[526] Google, in effect, desires the Federal Circuit to see copying the declaring code as copying for use and, thus, immunized. In this hypothetical from Nimmer, it would be unlikely that one could argue that merger had taken place as to the drawings and illustrations since they most likely may be expressed in various ways. Likewise, in *Google (2014),* there are various ways to express the declaring code, but Google would like its copying to be immunized, arguing that the merger doctrine applies by analogy with *Baker v. Selden.*[527] However, Nimmer states regarding his hypothetical, "This example serves as an illustration of how the distinction no longer should be deemed applicable under governing law."[528] To state this another way, due to how the law has evolved since *Baker v. Seldon*, the form-function doctrine does not apply to the hypothetical; likewise, it does not apply to software.

Engaging in a similar analysis with scène à faire, one should ask whether there were considerations that the author faced, such as a technical need or desire to interoperate with other *known* programs?[529] The point being, that if the author

---

text.

525 NIMMER ON COPYRIGHT § 2A.06(B)(1)(a), *supra* note 115.

526 NIMMER ON COPYRIGHT § 2A.03, *supra* note 1; *see* NIMMER ON COPYRIGHT § 2A.04, *supra* note 81; NIMMER ON COPYRIGHT § 2A.06, *supra* note 115.

527 Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1213–14 (2020).

528 NIMMER ON COPYRIGHT §2A.06(B)(1)(b), *supra* note 115.

529 It is important to realize this is quite different from a desire for interoperability with the author's work by a copyist. *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1370 (Fed. Cir. 2014) ("Because copyrightability is focused on the choices available to the plaintiff at the time the computer program was created, the relevant compatibility inquiry asks whether the plaintiff's choices were dictated by a need to ensure that its program

did face those concerns, then perhaps those aspects of the work, to the extent there is similarity with the accused work, are not probative of copying. That is, for scène à faire expression, though expression as opposed to simply an idea, such expression is so widely available that perhaps it did not originate, so to speak, from the protected work. Thus, the similarity that is attributed to a scène à faire expression may not be probative of copying and may be omitted in an analysis of substantial similarity. However, this point, of course, is subject to proof.[530]

When properly applied, the case for use of the scène à faire doctrine to limit the scope of protection for software may be more likely to occur than merger.[531] That is, external factors seem to have greater potential than merger to explain similarity between a protected work and an accused work at a non-literal level.[532] For example, in *Altai*, the suggestion was made that external factors, such as compatibility with certain hardware or even widely adopted industry custom, might justify application of this doctrine.[533]

In this respect, by way of comparison, the 2014 Federal Circuit decision in *Oracle v. Google* may seem confusing regarding how it treated interoperability.[534] The Federal Circuit was correct because, at trial, the scène à faire doctrine was not sufficiently developed.[535] In that case, Google must not have had a strong basis to suggest that external factors did affect the accused product. So scène à faire was not fully addressed on appeal.[536] Consequently,

---

worked with existing third-party programs.").

530 *See id.* at 1363 ("The trial court rejected Google's reliance on the scène à faire doctrine. It did so in a footnote, finding that Google had failed to present evidence to support the claim that either the grouping of methods within the classes or the code chosen for them 'would be so expected and customary as to be permissible under the scène à faire doctrine.'"). *See also* NIMMER ON COPYRIGHT § 13.03(F)(3), *supra* note 6. ("The further question arises as to who bears the burden of proof: Does it lie on plaintiff to prove as part of its *prima facie* case that the elements which it claims to be original fall outside the merger and scène à faire doctrine? Or, conversely, must defendant demonstrate the applicability of those doctrines as affirmative defenses? Although plaintiff's failure to present proof about those issues could defeat a plaintiff's application for a preliminary injunction, it would seem that defendant must go forward at trial with appropriate evidence as to those doctrines.").

531 Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F. 2d 693, 709–10 (2d Cir. 1992).

532 *Id.* at 710.

533 *Id.* at 709–10 ("[A] programmer's freedom of design choice is often circumscribed by extrinsic considerations such as (1) the mechanical specifications of the computer on which a particular program is intended to run; (2) compatibility requirements of other programs with which a program is designed to operate in conjunction; (3) computer manufacturers' design standards; (4) demands of the industry being serviced; and (5) widely accepted programming practices within the computer industry.").

534 *Compare Oracle Am., Inc*., 750 F.3d at 1363, 1370, *with Computer Assocs. Int'l Inc*., 982 F.2d at 710.

535 *Oracle Am., Inc.*, 750 F.3d at 1363–64.

536 *See id*. at 1363 ("The trial court rejected Google's reliance on the scène à faire doctrine. It did so in a footnote, finding that Google had failed to present evidence to

when addressing interoperability, the Federal Circuit in *Google (2014)* was relatively dismissive, suggesting that interoperability was an issue more properly relegated to a fair use analysis.[537] However, in what situation, if any, should interoperability be a factor affecting scope of protection?

Google was arguing that copying by it was appropriate in order for its product to be interoperable with Java.[538] However, the appellate court in *Google (2014)* was correct in concluding *that* desire by Google should *not* be a consideration in connection with scope of protection.[539] However, if, instead, Google had argued that some similarity between Google and Oracle's programs derived from the fact that the Google's software was interoperable with the same *other* software that was interoperable with Oracle's software, that, for example, could be relevant to the scope of protection to be afforded to Oracle's software. In that situation, Google's position would be that the similarity between Google's code and Oracle's code at a non-literal level is not due to copying from Oracle (again, the similarity in this instance would not be probative of copying), but rather simply because both programs were interoperable *with some other code*. Thus, in the appropriate case, interoperability could be a consideration to potentially limit the scope of protection under an analysis of substantial similarity.

Consider, for example, a typical API call of the form NAME (PARAMETER1, PARAMETER2 . . .), one example, being, as mentioned by the Federal Circuit in *Google (2014)*, "max(x,y)."[540] Of course, one position, which may be correct, is that merger applies. However, since merger and scène à faire are relatively close cousins, it might also be the case that scène à faire may be applicable. For example, consider a sufficiently complex code statement or set of statements that have the potential to be expressed a variety of enough possible ways so that merger would not apply. For example, suppose the API calls are a set of statements to well-known databases for converting GPS coordinates to other three-dimensional coordinates used with particular

---

support the claim that either the grouping of methods within the classes or the code chosen for them 'would be so expected and customary as to be permissible under the scène à faire doctrine.'").

537 *See Oracle Am., Inc*., 750 F.3d at 1371–72 ("Google maintains on appeal that its use of the 'Java class and method names and declarations was "the only and essential means" of achieving a degree of interoperability with existing programs written in the [Java language].' . . . Although this competitive objective might be relevant to the fair use inquiry, we conclude that it is irrelevant to the copyrightability of Oracle's declaring code and organization of the API packages.").

538 *Id.* at 1353.

539 *See id.* at 1370 ("Because copyrightability is focused on the choices available to the plaintiff at the time the computer program was created, the relevant compatibility inquiry asks whether the plaintiff's choices were dictated by a need to ensure that its program worked with existing third-party programs.").

540 *Id.* at 1349.

cellphone functions that have become commonly used and expected in the industry. In this instance, the presence of such calls in some code, including perhaps, the organization and structure of the code around those calls may otherwise appear similar, but may not be probative of copying. It may be the case, for example, that such calls have been relatively standard for compatibility across devices that execute code, perhaps through licensing across the industry, through open source made freely available, through a standards body making its specification freely available, etc. However, such interoperability considerations should have *by necessity* limited the creativity of the author of the code that is alleged to *have been infringed as well as* the creativity of the author of the code that is alleged *to have infringed*. Whereas, if those calls were, instead, taken from the code alleged *to have been infringed* due to its popularity, for example, so that the code that is alleged *to infringe achieves interoperability*, but the author of the infringed code was not otherwise constrained in this manner, that is an entirely different matter. [541] Below, the terminology "interoperable after the fact," is meant to capture this notion that, at the time of its creation, the need to achieve interoperability was not a limit on the creativity of the work that is alleged to have been copied.

Is it correct that interoperability after the fact would *never* be a consideration for scope of protection? Although an unlikely scenario, to say never may be a bit extreme. The 2014 *Google* court may have been overly dismissive on the topic of interoperability "by not considering that the merger doctrine coupled with the idea/expression dichotomy as well as considerations about the balance between protection and competition might, in a particular situation, may work together to limit the scope of protection in connection with "interoperability after the fact."[542]

To make this more concrete, consider a situation in which one or a few simple API calls may be used to achieve interoperability. That is, suppose a party is writing code and desires to write the code to operate on multiple platforms. Suppose one platform is, like Java, a popular platform and, in particular, the popular platform has a few simple known API calls to perform some known functions, but the source code implemented by those calls is not known. [543] So, imagine a few simple API calls having a specific, well-defined structure (e.g., name, list of parameters, and function to be performed). Suppose the author of code desires to use these small number of APIs so that the code will operate on

---

[541] *See id.* at 1372 ("Finally, to the extent Google suggests that it was entitled to copy the Java API packages because they had become the effective industry standard, we are unpersuaded.").

[542] *Id.* at 1368.

[543] *But see id.* at 1351 (distinguishing from the 37 API packages at issue in *Google (2014)*, in which the declaring code was copied verbatim).

the popular platform as well as on a separate platform being created. The author of the code has no rights with respect to the popular platform in this hypothetical. However, code being written by the author that is intended to be executable on both platforms may include the same names for the calls as well as the same parameters and, when executed, those calls may perform the same functions. This is to achieve interoperability for the source code, meaning here that the code with those API calls will execute on both platforms flawlessly – the popular platform and the new platform.[544] In this case, however, the so-called declaring and the implementing code for the APIs on the new platform are written from scratch. Only specific API calls, that respectively comprise for each common call, a name, parameters, and a particular function, which are limited in number, in this example, are copied.

As this hypothetical demonstrates, regarding scope of protection the devil is truly in the details. That is, given the right set of facts and circumstances, it might be imaginable that the scope of protection for popular API code might not extend so far so that merely using a set of API calls having the same call name and parameter names to provide the same function in a limited number of instances, and nothing else, amounts to copyright infringement.[545]

III. HYBRID IP RIGHTS FOR SOFTWARE (AND OTHER EXPRESSIVE WORKS?)

A. Relating Source Code, APIs, and GUIs

In this section we explore the nature of hybrid IP rights with respect to three related examples: software, APIs, and GUIs. We will attempt to clarify the legal distinctions, developed in the prior sections of this article, and discuss how they seem to differ for related, but different types, of expressive content. Then, having drawn appropriate distinctions based on legal analysis, we will consider how policy considerations might suggest a departure from the present legal approach that is employed in light of statutory pronouncements and case law.

An important conclusion from the previous sections of this article is that the Federal Circuit in *Google (2014)* correctly recognized that software functionality (e.g., form-function) does not generally limit the scope of non-literal copyright protection for software;[546] however, other traditional copyright

---

544 We assume, other than these calls, the rest of the code is just generic code, such as C++, for example.

545 *See Oracle Am., Inc.*, 750 F.3d at 1363 (using a fair use argument may also be available in light of the Supreme Court decision).

546 *See supra* notes 330–72 and accompanying text.

law doctrines, such as merger or scène à faire, may, without reference to functionality, limit such scope.[547] This ruling is left undisturbed by the Supreme Court, which reversed on other grounds (i.e., fair use).[548]

Some commentators or copyright law theorists, however, appear to disagree.[549] It may be that these commentators would prefer not to have a two-regime approach to protection; however, that approach goes a long way towards reconciling points that have confused courts. More important than the prior consideration, however, the will of Congress is clear that computer programs are to be copyrightable. Likewise, the fundamental nature of software is utilitarian. *These two principles together, it is believed, lead to the legal structure explored in this article described as the hybrid protection regime.*

A second important conclusion from the previous sections is that by and large the *Lotus* court also reached the correct outcome.[550] It is this juxtaposition, however, that in large part, it is believed, has some courts confused. *That is, confusion has resulted from not distinguishing types of copyrightable works and, consequently, not distinguishing the appropriate legal doctrines that respectively govern.*

With that said, we are reluctant to be too harsh in this observation because courts are exercising an intuition that, were this not a statutory area of law, and, instead, governed by common law considerations, might be entirely appropriate. Along these lines, it may be appropriate to view an API or a GUI, for example, as hybrid or hybrid-like.[551]

To explore this notion in more depth, consider the situation in *Lotus*.[552] As explained, that decision dealt with the copyrightability of a menu command hierarchy. We know already that a menu command hierarchy is not software, that is, it is *not* source code to be executed by a computer. However, is it an API? Is it a GUI? It appears to be neither. Not being software, the reasoning of the *Lotus* court appears correct as to copyrightability, whether viewed under the

---

547 *See id.*

548 *See* Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1186 (2021).

549 *See* Menell, *supra* note 6 at 309; *see also Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1296.

550 *See supra* notes 459–67 and accompanying text.

551 The term "hybrid-like" is meant to suggest forms of expression that may not be "hybrid" *per se*, but which do share similarities. For example, whereas with software, it is necessary to copy it to use it, this may not necessarily apply to some of these examples of expression, such as a GUI. Likewise, although some of these examples may be both expressive and useful, it may be that in some cases at least, some expressive elements might be separable from some useful elements, again, such as a GUI, for example. However, some works may be hybrid-like because they have elements that are expressive and useful that cannot be separated and that they are computer generated in some way, such as an API, a GUI, a file structure, a menu command hierarchy, etc. *See supra* notes 11–22 and accompanying text; *see supra* Section II.B.1.

552 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 809–10 (1st Cir. 1995).

form-function doctrine or under section 102(b) interpreted to be a statutory codification of that doctrine.[553] That is, the aspects of the menu command hierarchy that appear to be expressive are necessarily incidental to its use.[554]

However, the next point is tricky, important, and potentiality also somewhat controversial. While the menu command hierarchy is neither code nor necessarily a UI or GUI, it may, depending on the underlying software, relate to the SSO of that code, not unlike had been successfully argued in *Google (2014)* to the Federal Circuit. While the distinction between source code and a UI or GUI has at times been confusing for courts, an important and subtle point is that the non-literal scope of some source code may very well, in some factual situations, include some expressive aspects that relate to an interface, an API, a file structure or a hierarchical organization of commands, for example.[555]

On appeal, in *Lotus*, there was no question that the issue raised related to the menu command hierarchy; however, one might ask the question, had *Lotus*, instead, argued that the menu command hierarchy reflected the SSO of its source code,[556] might the legal result be like the result in *Google (2014)* or at least a much more difficult question for the First Circuit to resolve?[557] That is, the form-function doctrine is not applicable to software, and the SSO of the source code is within the non-literal scope of the source code.[558] Perhaps, arguably, copying the menu command hierarchy amounted to non-literal infringement of the source code. This, of course, is both a factual and a legal question that could only be answered with appropriate investigation. Nonetheless, just considering the possibility raises the point that this area raises subtle factual and legal questions that have the potential to lead to decisions that may superficially seem too not be fully consistent, but ultimately are consistent legally.

More specifically, this hypothetical causes us to consider whether the notion of hybrid rights should, on a policy basis, be extended beyond simply source

---

553 *Id.* at 812.

554 *See supra* notes 461–68 and accompanying text.

555 *See* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1244 (3d Cir. 1986) ("Insofar as everything that a computer does, including its screen outputs, is related to the program that operates it, there is necessarily a causal relationship between the program and the screen outputs. The screen outputs must bear *some* relation to the underlying programs, and therefore they have some probative value. The evidence about the screen outputs therefore passes the low admissibility threshold of Fed. R. Evid. 401." (emphasis in original)).

556 Recognizing, here, that this is a supposition for the purposes of discussion and may not be factually accurate.

557 *See Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1252–53 (stating that the *Lotus* and *Google* cases raise remarkably similar questions). This statement is a supposition for the purposes of discussion and may not be factually accurate.

558 *See supra* notes 64–114 and accompanying text.

code to other forms of expression that are similar to software, such as command structures, APIs and/or GUIs, while at the same time recognizing that this is ultimately an issue that should be decided by Congress, rather than courts.

In contrast to levels of abstraction, for example, but at least as theoretical, one way to think of the relationship among these works is in terms of the content, so to speak, forming a corpus or body of possibilities of realizable expression.[559] At one extreme might be expression comprising the source code, which, in terms of realizable expression constitutes a code implementation realization. At another extreme might be expression comprising the GUI, which is what is produced by executing the source code and which constitutes an audiovisual realization. Between these two extremes, depending on the content itself, one may consider there to be various other potential realizable expressions. Courts have observed that these realizations are related, which may have added to the confusion. For example, in the case of Lotus 1-2-3, an example of realizable content is the menu command hierarchy, which is neither the source code implementation realization nor the GUI audiovisual realization, but is related to both, almost as an intersection or type of combination of sorts and can be thought of as at a place or location somewhere between the two extremes. Likewise, at another place or location between these two extremes, depending on the content, may be an API structure, which may be another example of a realization of expression included within this body of content. At still another place or location between these two extremes, closer in a sense, to the source code, may be the SSO of the source code. Likewise, depending on the content, it may be, for example, that the SSO and the API structure are close expressive realizations or it may be, for example, that a menu command hierarchy and the SSO are close expressive realizations.

In a given case, if, for example, an API and source code are sufficiently close, as in substantially similar, then perhaps, copying the API may also amount to a non-literal infringement of the source code, as took place in *Google (2014)*.[560] Similarly, perhaps, in *Lotus*, it might potentially have been asserted that copying the menu command hierarchy amounted to non-literal infringement of the source code.[561]

Thought of in this manner, this may assist to articulate an intuition that seems implicit in some court rulings that relate decisions like *Lotus* to decisions regarding non-literal infringement of software.[562] While the intuition has merit, the legal considerations dictate that these be treated as separate copyrightable

---

559 It is recognized this model is highly abstract, and not perfect; nonetheless, it may help to capture a relationship that is otherwise difficult to precisely pin down.

560 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1379 (Fed. Cir. 2014).

561 Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 814–15 (1st Cir. 1995).

562 *Id.* at 819.

works, which may also lead to different outcomes, in light of a controlling statute, as opposed to being guided by common law decision making.

Decisions that attempt to follow *Lotus*, but concern software, may, therefore, be mistaken, because software is treated differently than a menu command hierarchy; likewise, however, courts that reject *Lotus*, in the manner that *Google (2014)* stated, for example, may also be mistaken, because as to a GUI, an API, or a menu command hierarchy, for example, there is no need to reject *Lotus.* The *Lotus* ruling is not fundamentally inconsistent with the ruling in *Google (2014)* based on the statutorily mandated principles involved.[563]

B. Policy Considerations

One point that should be clear from the preceding sections is that analyzing software and the non-literal scope of protection is a complex endeavor. For example, consider that in a given case, a court must (1) consider multiple levels of abstraction; (2) at each level, identify elements to be filtered out on multiple technical bases; and (3) compare what remains with the accused work to make a determination regarding substantial similarity.

Do the hybrid considerations make these questions more problematic? Potentially so since the utilitarian nature of software ultimately means that the content at issue is technical in nature. Legal issues, such as merger, scène à faire, etc., may be much more challenging questions, particularly when the parties present conflicting positions with conflicting evidence. Is it relatively easy for a court otherwise having little expertise with respect to the technology at issue to successfully identify multiple levels of abstraction with respect to the expression contained within source code, in general? Is it relatively easy for a court otherwise having little expertise with respect to the technology at issue to successfully evaluate whether expressive examples within some source code may be coded in multiple other ways?[564] Similarly, in the same situation, how easy is it for a court to determine that external considerations necessarily led to expressive features of the source code? How easy is it for a court to separate expressive features that are able to be coded only a few ways from other expressive features that should receive protection from copying, or how easy is

---

563 *Id.*; *see also Oracle Am., Inc.*, 750 F.3d at 1381.

564 *See* NIMMER ON COPYRIGHT § 13.03(F)(1)(a), *supra* note 6 ("Unfortunately, because computer programs tend to be incomprehensible to a lay judge or jury, evaluating the similarity between two computer programs is often exceedingly difficult. Such difficulties are particularly applicable when the allegations of infringement go beyond mere literal copying of the program code to claims that the organization and structure of plaintiff's program have been copied, thereby forcing the trier of fact to understand the design, structure, and function of both programs."). *See also supra* note 326 and accompanying text.

it for a court to separate expressive features that are necessarily coded in a particular manner as a result of external considerations from other expressive features that should receive protection from copying?

For example, it was previously noted by the Second Circuit, in the *Altai* case, that the lower court had inadvertently filtered the wrong work.[565] It is fair to litigants to have such complex technical and legal questions put before a court of general federal law? Instead, perhaps, as in patents, a special federal appellate court should hear such cases. A related question, for similar reasons, however, is: how easy is it for an appellate court to review such complex determinations that are to be made by a lower court? Perhaps even a special appellate court might not be sufficient. For example, despite the ruling in patent law that claim construction is relegated to the judge, not the jury, the reversal rate on appeal of claim construction determinations remains quite high.[566] This, of course, does not promote certainty in the law.

Another important question is whether and/or to what extent should economics factor into the question of non-literal scope of protection? For example, how clear it is that protection is needed for software through copyright law? To have this question resolved becomes that much more important in light of the Supreme Court's ruling in *Google v. Oracle*, which suggests a different balance between protection and competition for software content.[567] Perhaps, at the extreme, the market functions sufficiently well that the desired amount of software would be generated without protection under copyright law. If so, what of Congressional intent regarding the 1980 amendments? As noted, many times, *supra.*, courts do consider the balance between competition and protection; however, after *Feist*, protecting investment to produce the software, which sounds like protecting effort, is probably not an appropriate consideration, in

---

565 *See supra* note 165 and accompanying text. It was noted by the appellate court in *Altai* that the district court filtered out unprotectable elements from OSCAR 3.5, rather than from ADAPTER. This may show the challenge presented to courts by the complexity of this type of analysis in that the lower court got confused as to which program is to be filtered.

566 *See, e.g.*, Gretchen Ann Bender, *Uncertainty and Unpredictability in Patent Litigation: The Lime is Ripe for a Consistent Claim Construction Methodology*, 8 J. INTELL. PROP. L. 175, 194 (2001); Christian A. Chu, *Empirical Analysis of the Federal Circuit's Claim Construction Trends*, 16 BERKLEY TECH. L.J. 1075, 1075 (2001); Kimberly A. Moore, *Are District Court Judges Equipped to Resolve Patent Cases?* 15 HARV. J.L. & TECH. 1, 2 (2001); Kimberley A. Moore, *Markman Eight Years Later: Is Claim Construction More Predictable?* 9 LEWIS & CLARK L. REV. 231, 232 (2005); Michael Saunders, *A Survey of Post-Phillips Claim Construction Cases*, 22 BERKLEY TECH. L.J. 215, 215–18 (2007); Andrew Zidel, *Patent Claim Construction in the Trial Courts: A Study Showing the Need for Clear Guidance from the Federal Circuit*, 33 SETON HALL L. REV. 711, 713, 727, 737, 739 (2003).

567 Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1194 (2021).

comparison with protecting creativity.[568] *Feist* appears to alter the calculus that courts should use when balancing competition and protection. In this article, we have spoken a lot about incentives for generating copyrightable works. However, have those considerations changed? Should a court only focus on incentives related to creativity rather than market incentives?

For example, compare the approach of the *Altai* court, which came after *Feist*, with the *Apple* court and the *Whelan* court, both of which preceded *Feist*.[569] The *Apple* court stated: "The CONTU Final Report recognized that *'the cost of developing computer programs is far greater than the cost of their duplication.'* . . . Apple introduced substantial evidence of the considerable time and money it had invested in the development of the computer programs in suit."[570]

Along similar lines, the *Whelan* court stated:

> As we stated above, . . . among the more significant costs in computer programming are those attributable to developing the structure and logic of the program. The rule proposed here, which allows copyright protection beyond the literal computer code, would provide the proper incentive for programmers by protecting their most valuable efforts, while not giving them a stranglehold over the development of new computer devices that accomplish the same end.[571]

Yet, the *Altai* court stated:

> The immediate effect of our copyright law is to secure a fair return for an 'author's' creative labor. *But the ultimate aim is, by this incentive, to stimulate artistic creativity for the general public good* . . . When technological change has rendered its literal terms ambiguous, the Copyright Act must be construed in light of this basic purpose.[572]

The *Altai* court raises a subtle distinction compared with the *Whelan* and *Apple* courts.[573] While copyright law may help an author to secure a fair return, the ultimate aim is to stimulate artistic creativity, arguably a different consideration than simply protecting investment. However, software is at root utilitarian. Is it being intellectually honest to consider copyright law as stimulating artistic creativity when it comes to software? Consider, for example, the battle between

---

568 Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 346–47 (1991).

569 Apple Comput., Inc. v. Franklin Comput. Corp., 714 F.2d 1240, 1247 (3d Cir. 1983); *Feist Publ'ns, Inc*., 499 U.S. at 340–41; Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1237 (3d Cir. 1986); Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 711 (2d 1992).

570 *Apple Comput., Inc*., 714 F.2d at 1254 (emphasis added).

571 *Whelan Assocs., Inc*., 797 F.2d at 1237 (emphasis added).

572 *Comput. Assocs. Int'l, Inc.*, 982 F.2d at 711 (emphasis added).

573 *Id.* at 693.

Oracle and Google. Is that battle about artistic creativity or it is about technology and markets?

Furthermore, the non-literal protection regime for software, although primarily employing analytic legal tools consonant with copyright law doctrines, nonetheless, may have aspects that appear similar to patent law considerations. For example, copyright protection for software may at least indirectly encompass use of the software, since you must copy software in order to use it. Use is usually governed by patent rights and generally considered excluded from copyright law. Hence, in patent law, any overlap between form and function is really not an important an issue for patentable subject matter. For other areas of copyright law other than software, this overlap typically does affect copyrightability and/or scope of protection. Likewise, as mentioned previously, filtering out the public domain elements of a software work in the logical extreme leads to novelty.[574] However, novelty is a concept that applies in patent law, not in copyright law. The term hybrid may be appropriate here, in addition to the reason provided in the introduction, because the rights involved are potentially a type of hybrid of patent considerations and copyright considerations.[575]

---

574 *See supra* note 317 and accompanying text.

575 Some commentators have taken note about the potential overlap between protection provided via a software copyright and protection afforded through patent protection. *See Staking the Boundaries of Software Copyrights in the Shadow of Patents*, *supra* note 6, at 286; *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, *supra* note 6, at 1284; NIMMER ON COPYRIGHT § 2A.07, *supra* note 16. Many key differences, however, exist between the scope of protection provided by a patent versus a copyright, which, of course, necessarily includes software. For a patent – *the scope of protection is defined by the claims, which are prepared by the applicant*. *See, e.g.*, Vitronics Corp. v. Conceptronic, Inc., 90 F.3d 1576, 1582 (Fed. Cir. 1996). For a copyright, *the scope of protection is determined by substantial similarity of the expression of a copyrightable work to an accused work. See, e.g.*, Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1545 (11th Cir. 1996). It would be extremely difficult for the scope of protection available for a computer program via copyright, including with respect to non-literal protection, to approach the scope of protection able to be garnered via patent protection. Thus, it is believed that concerns about overlap may be overstated and/or misplaced. A patent typically covers making, using, selling, offering for sale or importing tangible apparatuses (or processes), for example, that fall within the scope of the patent claims. *See* 35 U.S.C.§ 271(a) (2012). A copyright covers copying (used in the broad, non-literal sense) – albeit, even considering non-literal scope, this at most relates to 'use' only to the extent that copying of a program indirectly includes use – and this applies only where there are a myriad of ways to accomplish the same task (e.g., absent merger). *See, e.g.*, *supra* note 50–86 and accompanying text. We also observe that the Constitution's Intellectual Property Clause gives Congress the power "to promote the Progress of Science and Useful arts, by securing, for limited Times, to Authors and Inventors, the exclusive Right to their respective Writings and Discoveries," and on its face has few clear limits imposed regarding this particular Congressional power other than that the protection be "for limited times" and be with respect to "writings" and "discoveries." Several Supreme Court cases have sought to construe various terms of this provision, such as the term "writings." U.S CONST. art. I, § 8,

Answers to the questions above are beyond the scope of this article; however, a different question, more legal in nature, is to what extent should the hybrid approach for software also apply to expression that, while technically not software, might be related to software and may potentially be hybrid-like in nature, such as GUIs, APIs, file structures, command structures, or the like? For example, should a court, at the beginning of a case, consider whether the case involves hybrid rights to determine what set of substantive legal considerations should govern? As a practical matter, to treat GUIs, APIs, etc., like software, requires Congressional action. In this regard, there have been several calls for a CONTU II.[576]

Nonetheless, it might be interesting to consider, how the *Lotus* case might have been decided if it had been treated as hybrid-like IP. Under current law for a copyrightable work, like a book or a movie, for example, if the work is original, fixed in a tangible medium and creative, then it is subject to protection. In that instance, the scope of protection is largely governed by the standard of substantial similarity.[577] This amounts to a full scope of protection. On the other hand, as in *Lotus*, which involved a menu command hierarchy, despite potentially being original and creative, no protection was ultimately provided. Outside of the unique situation carved out for software, other potentially hybrid-like situations based on statutory considerations should generally receive no copyright protection.[578] This may be viewed as an "all or nothing" approach.

In the unique case of software, which we have called hybrid IP, however, an AFC analysis governs. Such an approach is not full protection, because some expression is filtered from the copyrightable work. However, likewise, it provides at least some protection because there is literal protection and potentially some non-literal protection. The protection may be thinner, so to speak, than full protection, due to the hybrid nature that we have explored, but it provides an alternative to the "all or nothing" approach. [579] From a policy

---

cl. 8. *See, e.g.*, Mazer v. Stein, 347 U.S. 201, 207 (1954); Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 346–47 (1991). Granting that there may be limits on the exercise of this Congressional power, either by logical construction and/or court decisions, it should be noted that what Congress may be unable to do under one of its powers, it may still be able to do under another power, such as under the interstate commerce power, for example. *Cf.* Trade-Mark Cases, 100 U.S. 82, 97 (1879). In this latter case, the Supreme Court held a federal trademark statute unconstitutional; however, afterwards, Congress re-enacted another federal trademark statute and made it clear it was relying on the interstate commerce power and the statute since then appears to have been accepted as constitutional. *See Mazer*, 347 U.S. at 206 n.5.

576 *See* Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d at 712; Armstrong, *supra* note 6, at 135.

577 *See supra* notes 63–86 and accompanying text.

578 *See* Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 813 (1st Cir. 1995).

579 *Cf.* 4 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 3.04 (2021)

perspective, then, in a system in which encouraging more creative works is the goal, perhaps a thinner approach is preferable to an "all or nothing" approach.[580]

So, consider, what would have happened using such an approach with *Lotus*. Well, of course, as we know, the copying was literal and conceded to have been done, so there would have been infringement. [581] However, if protection had been provided, would it have prevented developments like Excel from coming into existence? Probably not because the menu command hierarchy of Excel is considerably different from Lotus 1-2-3. Furthermore, it may be that the scope of protection with appropriate filtering would not provide protection much beyond literal copying of the exact menu command hierarchy. Finally, again, now, there is also a credible possibility of fair use. Consistent with what appears to be the direction of the Court, perhaps given the objective of copyright law to encourage the creation of more works for the benefit of everyone, this would be a workable approach with better results for software *and* for those areas outside of software that share some of its hybrid-like characteristics.

IV. CONCLUSION

As has been discussed in some detail, copyright law employs two regimes of protection, one referred to here as a traditional approach and one referred to here as a hybrid approach. The hybrid approach governs protection of software because software is hybrid, being both expressive and useful.

As a result of its hybrid nature, the scope of protection that courts provide software is different, particularly the scope of non-literal protection. Regarding both literal and non-literal protection, the form-function doctrine of copyright law does not apply. This is primarily an interpretation that follows from Congressional intent and the fact that to use software, it must be copied. Another difference regarding the scope of protection for software is that public domain elements are filtered out. This approach does not generally apply in traditional areas where copyright protection is provided.

Unfortunately, due to the complexities in this area, courts sometimes confuse a number of concepts. The issues confused most often include distinguishing between types of works, such as distinguishing software from related expression, such as a GUI, an API, or a menu command hierarchy. Courts also confuse the merger doctrine with the form-function doctrine. Thus, as a result of these last two points, courts sometimes confuse the legal considerations that govern these

---

(discussing "thin" copyrights in connection with compilations and derivative works).

580 *See* Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183, 1199 (2021) (stating "We do not believe that an approach close to 'all or nothing' would be faithful to the Copyright Act's overall design.").

581 *See Lotus Dev. Corp*., 49 F.3d at 814.

different types of works.