2018

Symbols, Systems, and Software as Intellectual Property: Time for Contu, Part II?

Timothy K. Armstrong
Professor of Law, University of Cincinnati College of Law, timothy.armstrong@uc.edu

SYMBOLS, SYSTEMS, AND SOFTWARE AS INTELLECTUAL PROPERTY: TIME FOR CONTU, PART II?

*Timothy K. Armstrong**

Cite as: Timothy K. Armstrong, *Symbols, Systems, and Software as Intellectual Property: Time for CONTU, Part II?*, 24 MICH. TELECOM. & TECH. L. REV. 131 (2018).
This manuscript may be accessed online at repository.law.umich.edu.

ABSTRACT

The functional nature of computer software underlies two propositions that were, until recently, fairly well settled in intellectual property law: first, that software, like other utilitarian articles, may qualify for patent protection; and second, that the scope of copyright protection for software is comparatively limited. Both propositions have become considerably shakier as a result of recent court decisions. Following Alice Corp. v. CLS Bank Int'l, 134 S. Ct. 2347 (2014), the lower courts have invalidated many software patents as unprotectable subject matter. Meanwhile, Oracle America v. Google Inc., 750 F.3d 1339 (Fed. Cir. 2014) extended far more expansive copyright protection to functional software components than precedent suggested. The result of these developments has been a new period of uncertainty regarding the existence and scope of intellectual property protection for computer software.

The root of the problem lies in Congress's relative inattention to the question of what legal regime (if any) should govern the creation of computer software. Congress extended copyright protection to software largely without grappling with the consequences of applying a body of law designed to promote creative expression to functional, useful code. Meanwhile, Congress has spoken only obliquely to the question whether software warrants patent protection. The turmoil in the courts

* Professor of Law, University of Cincinnati College of Law. B.A. 1989, M.P.Aff. 1993, J.D. 1993, The University of Texas at Austin; LL.M. 2005, Harvard Law School. For helpful comments on earlier drafts of this article, I am grateful to Pamela Samuelson and to the participants in the University of Cincinnati College of Law's Summer 2016 Faculty Scholarship Workshop, the attendees at the College's 2016 Faculty Teach-In, and to the conferees at the 2017 Works-in-Progress Intellectual Property (WIPIP) colloquium at the Boston University School of Law. University of Cincinnati law student Elias Sayre ('18) provided valuable research assistance.

reflects a general lack of legislative guidance. This Article asks whether the time is ripe for remedial legislation and suggests some questions that ought to guide congressional inquiry.

TABLE OF CONTENTS

INTRODUCTION

Computer software is uniquely challenging subject matter for intellectual property law, because software perfectly fits none of the law's governing paradigms. Like literary works—ordinarily the subject matter of copyright law—software is "expressed" in the form of alphanumeric symbols (to wit, the source code written by one or more computer programmers).[1] Unlike, say, a poem or an essay, however, software also has functional characteristics: the symbols expressing its source code are trans-

1. *See, e.g.*, U.S. NATIONAL COMM'N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT OF THE NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS 10 (1979) ("Computer programs are prepared by the careful fixation of words, phrases, numbers, and other symbols in various media."), 15 ("In this respect [a program] is the same as a novel, poem, play, musical score, blueprint, advertisement, or telephone directory.") [hereafter "CONTU FINAL REPORT"]

lated, before or during execution of the program, into a usable form that performs work on a computer.[2] It's partly functional nature gives software, in operation, the character of a machine, system, or process—ordinarily the subject matter of patent law, not copyright.[3] Programs use symbols not only to communicate, but to describe (and, indeed, to implement) a system that actually performs work.[4] Software thus straddles one of intellectual property's most important conceptual divides.[5]

The copyright-patent distinction is not merely semantic, because copyright and patent law provide very different protections to authors and inventors. Copyright law defines six specific uses of protected works that infringe if conducted without authorization of the copyright holder (necessarily leaving uses outside the enumerated six unrestricted), then adds dozens of statutory exceptions defining circumstances in which the law deems even unauthorized uses noninfringing.[6] Patent law, instead of enumerating specific forbidden uses (qualified by numerous exceptions), categorically forbids anyone to "make, use, offer to sell, or sell any patented invention," or to import it into the United States, without the patentee's authorization.[7] The

---

2. On the distinction between human-readable "source code" and the corresponding machine-readable "object code" into which programs must be converted at or before execution, see, *e.g.*, Microsoft Corp. v. AT&T Corp., 550 U.S. 437, 448 n.8 (2007); *Id.* at 459 (Alito, J., concurring).

3. *Compare* 17 U.S.C. § 102(b) (2018) ("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. . .."), *with* 35 U.S.C. § 101 (2018) ("Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title. . .").

4. *See, e.g.*, Pamela Samuelson, et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308, 2320-24 (1994) ("Programs are machines whose medium of construction is text.").

5. *See, e.g.*, Dennis S. Karjala, *A Coherent Theory for the Copyright Protection of Computer Software and Recent Judicial Interpretations*, 66 U. CIN. L. REV. 53, 56–66 (1997) (analyzing the meaning and role of functionality in policing the patent-copyright boundary).

6. *See* 17 U.S.C. §§ 106 (enumerating copyright holders' exclusive rights), 107–122 (describing limitations and exceptions to exclusive rights) (2018). Of particular interest in the context of software are the fair use exception (§ 107), which allows unauthorized copying for certain socially beneficial purposes; the first sale doctrine (§ 109), which permits unauthorized redistribution of a lawfully acquired copy of a work; and a series of computer-specific exceptions (§ 117) allowing software to be backed up and to be used in the ordinary operation of a computer, among other things. As the Supreme Court has made clear, United States law recognizes no copyright rights not expressly granted by statute. *See, e.g.*, Wheaton v. Peters, 33 U.S. (8 Pet.) 591, 661 (1834) ("Congress, by the [Copyright] Act of 1790, instead of sanctioning an existing perpetual right. . . created [it]."); *see also* 17 U.S.C. § 301(a) (2018) (expressly preempting any common-law rules "within the general scope of copyright"). Thus, the rights specifically enumerated in the federal statute constitute the totality of copyright holders' enforceable interests in a work; the statute denies copyright holders the right to limit uses of their works in any respects not expressly granted.

7. 35 U.S.C. § 271(a) (2018). The patent statute does contain a narrow exemption permitting unauthorized use of patented genetic inventions for the purpose of testing drugs for

existence of patent and copyright rights depends on different preconditions for each form of protection, and the rights conferred endure for widely differing lengths of time.[8] Moreover, the protections conferred under copyright and patent law are, with immaterial exceptions, exclusive: what is copyrightable generally cannot be patented, and vice versa.[9] The question of whether software is protected under copyright or patent, or whether it may include some aspects of both forms simultaneously, is therefore highly consequential.

Recent court decisions, however, show that the analytical framework for resolving questions of this type has grown exceptionally shaky. Although the courts unanimously agree that copyright protection extends to software, they differ over how far copyright protection extends and what conduct infringes a program's copyright.[10] On the patent side, things have grown murkier still: although the Federal Circuit embraced software patenting with (arguably excessive) enthusiasm in the 1990s, sparking a decades-long surge in the issuance of software patents by the USPTO, the law has moved sharply in the other direction since the Supreme Court's 2014 decision in *Alice Corp. v. CLS Bank Int'l*,[11] with significant consequences that the lower courts are still resolving.[12] At the time of this writing, the best that can be said is that copyright protection for software exists, but with ill-defined boundaries; and patent protection may be available only for certain types of software but not for others.

---

regulatory approval. *Id.* § 271(e)(1). It also contains an exception grandfathering prior commercial uses of inventions used in "manufacturing or other commercial process[es]" that were later patented by another. *Id.* § 273(a); *see also infra* note 150 and accompanying text (describing origin of this provision).

8. *See, e.g.*, Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 819 (1st Cir. 1995) (Boudin, J., concurring) ("It is no accident that patent protection has preconditions that copyright protection does not—notably, the requirements of novelty and non-obviousness—and that patents are granted for a shorter period than copyrights."). Copyright rights last roughly five times longer than patent rights do. *Compare* 17 U.S.C. § 302(c) (2018) (establishing copyright term of 95 to 120 years for works made for hire created since 1978, a category that includes most computer software), *with* 35 U.S.C. § 154(a)(2) (2018) (establishing 20-year patent term). Given the pace of development in the software industry, however, the differences may be less stark than they initially appear; for practical purposes, both copyrights and patents endure for terms that may comfortably exceed the commercial viability of the products to which they apply.

9. *See, e.g.*, Baker v. Selden, 101 U.S. 99, 102 (1880) (holding that system or method of accounting described in a textbook is not encompassed within the scope of the book's copyright; protection of systems and methods "is the province of letters-patent, not of copyright.")); *but cf.* Mazer v. Stein, 347 U.S. 201, 215–17 & n.33 (1954) (suggesting that design patent protection, as distinct from utility patent protection, may coexist with copyright protection in the same goods).

10. *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1357 (Fed. Cir. 2014) ("Circuit courts have struggled with, and disagree over, the tests to be employed when attempting to draw the line between what is protectable expression and what is not.").

11. Alice Corp. v. CLS Bank Int'l, 134 S. Ct. 2347 (2014).

12. *See generally infra* Part III.

To be sure, uncertainty as to the extent of intellectual property protection for computer software is nothing new. Doctrinal fluidity is the price we pay for applying statutes written long ago in broad, general terms to rapidly changing technologies. Generalist judges have struggled with novel technological subject matter (with varying success) for decades. But although legal uncertainty might have been safely ignored during the digital era's infancy because the law affected few people, those conditions no longer exist. In an era where software has become a ubiquitous component of a vast assortment of consumer products—software now runs our automobiles, household appliances, cell phones, televisions, and more—as well as a significant and still-growing sector of the United States economy, legal uncertainty poses substantial risk to consumers and developers alike.[13]

The party best situated to ameliorate the present legal uncertainty—Congress—has largely absented itself from delineating the boundaries of software as a form of intellectual property. The governing statute extending copyright protection to software was enacted in 1980—eons ago, technologically speaking—and the most recent major revision of patent law (the Leahy-Smith America Invents Act of 2011) made only glancing reference to the protectability of software. The present turmoil in the courts, however, suggests that we have reached an opportune moment for Congress to reassess the state of intellectual property protection for software. Partly in re-

13. On October 22, 2015, U.S. Senators Chuck Grassley (R-IA) and Patrick Leahy (D-VT), the Chair and ranking minority member of the Senate Judiciary Committee, asked the Register of Copyrights to launch a year-long "comprehensive review of the role of copyright" in regulating "how consumers can lawfully use products that rely on software to function." *See Grassley & Leahy Call For Copyright Study*, http://www.grassley.senate.gov/news/news-releases/grassley-leahy-call-copyright-study (Oct. 22, 2015), *archived at* [perma.cc/3HV6-WWZM] [hereafter "Senate Judiciary Committee letter"]. The Senators' letter recognized that "[c]opyrighted software is now essential to the operation of our refrigerators, our cars, our farm equipment, our wireless phones, and virtually any other device you can think of"). *Id.* Furthermore, the role of copyright law in regulating products and activities seemingly unrelated to the creative arts achieved public prominence recently following the revelation that German auto maker Volkswagen had configured the software in its vehicles to alter the results reported during required emissions testing. *See* Complaint at 14–16, United States v. Volkswagen AG, No. 2:16-cv-10006-LJM-MJH (E.D. Mich. Jan. 4, 2016). The fact that the vehicles' software was protected by copyright, and was therefore subject to the prohibitions of the Digital Millennium Copyright Act ("DMCA") against circumventing technological measures that control access to a copyrighted work, was believed to have made the detection of Volkswagen's alleged manipulation more difficult. *See, e.g.*, Mitchell Hartman, *A story of dirty emissions. . .and copyright law*, MARKETPLACE (Sept. 28, 2015, 5:00 AM), http://www.marketplace.org/2015/09/28/tech/story-dirty-emissions-%E2%80%A6-and-copyright-law, *archived at* [perma.cc/82RV-TZAE]. The Volkswagen revelations surely influenced the decision of the Copyright Office to promulgate a DMCA exemption permitting the circumvention of technological measures protecting "[c]omputer programs that are contained in and control the functioning of a motorized land vehicle such as a personal automobile[.]" Exemption to Prohibition on Circumvention of Copyright Protection Systems for Access Control Technologies, 37 C.F.R. § 201.40 (2017). *See generally* Paul Ohm & Blake Reid, *Regulating Software When Everything Has Software*, 84 GEO. WASH. L. REV. 1672 (2016).

sponse to criticism by the Register of Copyrights that the copyright statute is increasingly outdated,[14] the House has held multiple hearings aimed at producing comprehensive copyright reform legislation.[15] With the Senate also poised for possible action in the area of software,[16] the time may be ripe to reassess and synthesize the last four decades of legal and technological development to craft rules that are more stable, certain, and better tailored to the unique context of software.

Because the subject lies outside the expertise of most members of Congress, it would be desirable for legislators to seek independent guidance before undertaking substantial revisions to the law. One possible analogue for such guidance may be found in the 1970s-era Commission on New Technological Uses of Copyrighted Works ("CONTU"), whose investigation prompted the enactment of the first copyright statute covering computer software. Although many scholars have critiqued the substance of the legal regime that resulted from CONTU's efforts, and although some portions of the Commission's report have undoubtedly aged better than others, CONTU remains noteworthy for the ambitious scope of its analysis. Although Congress should revisit many of the foundational assumptions on which the original CONTU's analysis rested, the general idea of a disinterested expert commission advising legislators on unfamiliar but economically important subject matter remains appealing. Our collective understandings of the nature of computer software and the role of the law in encouraging technological development have evolved substantially during the nearly four decades that have elapsed since Congress last meaningfully grappled with basic issues about the proper scope of legal protection for software. Using those insights better to inform policy is both a practical and economic imperative.

This Article reviews the development of intellectual property protection for software (primarily, although not solely, governed by changes in copyright and patent law) with a view toward exploring how past debates may color future doctrinal evolution. Part II explores the history of copyright protection for software, focusing first on Congress's choice to extend copyright protection to software works by statute, and then on the changes over time in judicial thinking about how far the statutory protection extends. Part III turns to patent law, where the debate has been driven less by Congress and more by changes in the case law in the Federal Circuit and, later, the Supreme Court. Part IV surveys the prospects for reform and concludes that Congress is the institutional actor best suited to ameliorate the legal uncertainty that has produced such wide swings in judicial doctrine. Finally, Part V con-

14. *See* Maria A. Pallante, *The Next Great Copyright Act*, 36 COLUM. J.L. & ARTS 315 (2013).

15. *See* Committee on the Judiciary, U.S. House of Representatives, U.S. COPYRIGHT REVIEW, https://judiciary.house.gov/issue/us-copyright-law-review/ (last visited Jan 21, 2018), *archived at* [perma.cc/Y5L8-S667] [hereafter U.S. Copyright Review].

16. *See* Senate Judiciary Committee letter, supra note 13.

cludes with an assessment of what recent legal changes may suggest for the future and proposes an agenda for further legislative study.

I. Software Copyright Law

A. *The 1976 Act and the Birth of Software Copyright Law*

Serious effort to update the increasingly moribund Copyright Act of 1909 began in Congress in the early 1960s. Almost immediately, the applicability of the statute to the new field of computer technology was recognized as a significant issue. In 1963, a draft of proposed copyright legislation included a provision that would have expressly given copyright holders the exclusive right "to reproduce [a work] in any form in the programming or operation of an information retrieval system."[17] Despite the proposed clause's reference to "programming," however, the context suggests that Congress was concerned primarily with the use of computers to store and analyze ordinary works of literature and music.[18]

The next revision of the bill, in 1964, deleted the reference to computers or programming—reflecting a deliberate choice, the Register of Copyrights explained, to rely solely on "general language which can be interpreted by the courts to apply to particular usages."[19] One witness hypothesized that the statute's definition of "literary work" already included computer programs by implication, yielding "much broader copyright status" for software "than under the present [1909] Act."[20] The General Electric Company, however, urged Congress not to lump software in with other literary works. Rather, it suggested, legal protection for software "should be specifically delimited in light of the special character and problems of this art."[21]

---

17. H.R. Comm. on the Judiciary, 88th Cong., Copyright Law Revision, Part 3: Preliminary Draft for Revised U.S. Copyright Law and Discussions and Comments on the Draft, at 4 (Comm. Print 1964) (quoting § 5(a) of the proposed legislation).

18. *See id.* at 120–27. One witness, for example, imagined that it would soon become possible to "feed a book into the machine in its entirety and then make it available to the world at large." *Id.* at 122 (statement of George Schiffer); *see also id.* at 374–76 (reproducing suggestion by Reed C. Lawlor that computerized reproduction of works "for use in the analysis, citation and reasonable quotation" of the computer system's users "shall be considered a fair use"). History eventually validated both these predictions. *See* Authors Guild v. Google, Inc., 804 F.3d 202 (2d Cir. 2015). Lawlor recognized, if perhaps most members of Congress at the time did not, that the questions "What computer-associated acts infringe a copyright?" and "Are computer programs copyrightable?" were distinct. Reed C. Lawlor, *Copyright Aspects of Computer Usage*, 11 Bull. Copyright Soc'y U.S.A. 380, 394 (1964).

19. H.R. Comm. on the Judiciary, 89th Cong., Copyright Law Revision, Part 5: 1964 Revision Bill with Discussions and Comments, at 63 (Comm. Print 1965) [hereinafter Copyright Law Revision, Part 5].

20. *Id.* at 62 (statement of Morton David Goldberg); *see also infra* note 24 (noting that some software had been copyrighted even under the 1909 statute).

21. *Copyright Law Revision, Part 5*, *supra* note 19, at 271 (General Electric Position Statement).

In 1965, the Copyright Office made explicit the implication of the 1964 draft, declaring that, in its view, "computer programs fixed on punchcards, magnetic tape, or any other media" fell within the proposed statutory definition of a "literary work."[22] At the same time, the Copyright Office signaled its retreat from the 1963 draft's reference to the legality of uses of expressive works in "information retrieval systems," reasoning that "it would be a mistake for the statute, in trying to deal with such a new and evolving field as that of computer technology, to include an explicit provision that could later turn out to be too broad or too narrow."[23]

The 1965 draft bill drew a favorable reaction from a representative of the then-nascent commercial software industry, who predicted that the availability of copyright protection for programs would lure new investment.[24] Others, however, foresaw the conceptual difficulty of treating software the same as other literary works. The Electronic Industries Association wrote that the basic problem lay in the impossibility of making lawful uses of uncopyrighted material embedded within a program without duplicating the program—a problem that did not affect other types of literary works.[25] Fur-

22. H.R. COMM. ON THE JUDICIARY, 89TH CONG., COPYRIGHT LAW REVISION, PART 6: SUPPLEMENTARY REPORT OF THE REGISTER OF COPYRIGHTS ON THE GENERAL REVISION OF THE U.S. COPYRIGHT LAW: 1965 REVISION BILL, at 5 (Comm. Print.1965).

23. *Id.* at 18.

24. *Copyright Law Revision: Hearings Before Subcomm. No. 3 of the House Comm. on the Judiciary, 89th Cong., 1st Sess., on H.R. 4347, H.R. 5680, H.R. 6831, H.R. 6835*, Serial No. 8, Part 2, at 1146 (statement of John F. Banzhaf III). Banzaf had himself successfully sought copyright protection for a computer program even under the then-existing Copyright Act of 1909. *See* Michael S. Keplinger, *Computer Software—Its Nature and its Protection*, 30 EMORY L.J. 483, 494–95 (1981) (describing conditions under which Copyright Office had deemed software to be copyrightable under 1909 Act).

25. The Association explained:

> The copyright grant is given to authors because the public benefits from his creative activities. The public benefits because it is free to use the ideas contained in a copyrighted work. For example, the public is free to copy a mathematical formula from a copyrighted book. Computer programs are basically mathematical formulas in the form of instructions to a computer.
>
> This objective of the copyright laws cannot be achieved if one who lawfully obtains a copyrighted magnetic tape cannot reduce it to intelligible form to see if it contains a usable mathematical formula. Insofar as copyrighted computer programs on magnetic tape are concerned, the ideas are not ascertainable without a print-out of the work in visual form. If an intelligible copy could not be made, such a medium of expression could be substantially maintained in secrecy. Therefore, one of the basic purposes of the copyright laws will not be fulfilled unless one can make an intelligible copy to ascertain its contents.

Copyright Law Revision: Hearings Before Subcomm. No. 3 of the House Comm. on the Judiciary, 89th Cong., 1st Sess., on H.R. 4347, H.R. 5680, H.R. 6831, H.R. 6835, Serial No. 8, Part 3, at 1898 (statement of Electronics Industries Association). Modern "functional programming" languages such as Haskell make the Association's analogy between computer programs and "mathematical formulas" literally true; all such programs are formally expressed as mathe-

thermore, the draft's omission of any provision specifically addressing the legality of using a computer to analyze copyrighted works also drew criticism from scientific experts.[26]

Thus, not later than the mid-1960s, two disputed issues surrounding copyright law and computer technology had emerged: first, should computer programs receive protection under copyright as a form of literary work, or via some other approach more narrowly focused on the unique characteristics of software? And second, should the use of computer technology to aid in the study and analysis of expressive works fall within the purview of the new statute? The novelty and uncertainty of these questions prompted calls for Congress to study the matter further before legislating.[27] As Professor Benjamin Kaplan recognized, for example, the question was not merely the proper reach of copyright protection for software, but whether copyright should apply at all in view of the functional characteristics of software "works."[28]

So as not to delay the broader reform bill that would become the Copyright Act of 1976,[29] Congress on December 31, 1974 created the National Commission on New Technological Uses of Copyrighted Works ("CONTU").[30] The thirteen-member Commission was directed "to study and

---

matical functions that return a result. *See, e.g.*, Richard Bird, THINKING FUNCTIONALLY WITH HASKELL 1 (2014).

26. *Copyright Law Revision: Hearings Before the Subcomm. on Patents, Trademarks, and Copyrights of the Senate Comm. on the Judiciary, 90th Cong., 1st Sess., on S. 597*, Part 1, at 581–89 (statement of Prof. Anthony G. Oettinger of the Association for Computing Machinery).

27. *Id.* at 193–96 (statement of Prof. Arthur R. Miller). Interestingly, in his statement, Prof. Miller also stated that the functional attributes of computer programs should disentitle them to copyright protection. *Id.* at 196–97. His later service on the CONTU Commission persuaded him to the contrary, however. *See infra* note 48 and accompanying text.

28. *See* Benjamin Kaplan, *An Unhurried View of Copyright: Proposals and Prospects*, 66 COLUM. L. REV. 831, 843 (1966) ("A suspicion, indeed, that we ought to be thinking not copyright but patent or perhaps a third quiddity, arises as we are told that the programs, or some of them, can be translated, so to speak, into physical parts of the computer's machinery or circuitry.").

29. Other scholars have noted the generally unfavorable consequences of the timing of the software copyright debate on policy: software emerged as a contentious issue when negotiations over copyright revision were well advanced, leading Congress to shunt the issue to one side rather than address it forthrightly in the 1976 Act. Furthermore, because legal issues surrounding software arose when the industry was in its relative infancy, they rested upon a nascent understanding that later developments have called into question. *See, e.g.*, Peter S. Menell, *Rise of the API Copyright Dead?: An Updated Epitaph for Copyright Protection of Network and Functional Features of Computer Software*, https://ssrn.com/abstract=2893192, manuscript at 12 ("The software protection controversy emerged at an inopportune time."); *id.* at 12–15 (summarizing CONTU's work); Pamela Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 DUKE L.J. 663, 694 ("It is possible to ask whether Congress was fully informed of the implications of the decision before it was made").

30. *See* Pub. L. No. 93-573, §§ 201–208, 88 Stat. 1873, 1873–75 (1974). The Commission had been proposed as early as 1967, and legislation had cleared the Senate, but the House

compile data" concerning both the questions stated above, and to "make recommendations" to Congress for further legislative action within three years.[31] Congress and the Copyright Office expressed hope that the formation of the CONTU Commission would permit the remainder of the reform legislation to proceed, postponing for a later date the difficult questions that the Commission was tasked with addressing.[32]

As finally enacted on October 19, 1976 (with an effective date of January 1, 1978), the new Copyright Act essentially echoed the position the Copyright Office had taken on the 1965 bill: the legislative history declared Congress's intent that software should be treated as analogous to a "literary work," which the statutory language weakly buttressed by including "tapes, disks, or cards" as examples of media in which a literary work could be "fixed."[33] The only statutory language expressly referencing computers simply preserved the law as it existed under the 1909 Act.[34] The legislative

---

took no action until it began to become clear that uncertainty over computer issues was potentially impeding action on other areas where substantial agreement had already been achieved. *See, e.g.*, H.R. Rep. No. 94-1476, at 48, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5661.

31. *Id.* §§ 201, 206, 88 Stat. at 1873–74, 1875. Many scholars have articulated (often, but not uniformly, critical) evaluations of CONTU's work. Examples include Arthur R. Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?*, 106 HARV. L. REV. 977 (1993); Paul Goldstein, *Infringement of Copyright in Computer Programs*, 47 U. PITT. L. REV. 1119, 1119–22 (1985); Samuelson, *supra* note 29.

32. *See, e.g.*, H.R. Rep. No. 93-1581, *reprinted in* 1974 U.S.C.C.A.N. 6849, 6854–55.

33. *See* Pub. L. No. 94-553, 90 Stat. 2541, 2543 (1976) (definition of "literary works"). The legislative history explained:

> The term "literary works" does not connote any criterion of literary merit or qualitative value: it includes catalogs, directories, and similar factual, reference, or instructional works and compilations of data. It also includes computer data bases, *and computer programs to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves*.

H.R. Rep. No. 94-1476, *supra* note 30, at 37 (emphasis added). The statute's exclusion of "procedure[s], process[es], system[s], [and] method[s] of operation" from the scope of protectable subject matter, the report continued, was meant to clarify 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." *Id.* at 38; *see also* Michael S. Keplinger, *Computer Intellectual Property Claims: Computer Software and Data Base Protection*, 1977 WASH. U. L.Q. 461, 463 (crediting this language with preventing "monopolization of the processes or algorithms embodied in a program" under copyright law); Goldstein, *supra* note 31, at 1125 ("You need only contrast section 102(b)'s statement of what copyright does *not* protect with the same Act's definition of 'computer program,' to appreciate how very thin, indeed, is the infringement protection available to computer software."). Many courts, however, have given section 102(b) far less weight as a limitation on the reach of copyright protection for software than Congress intended. *See* Pamela Samuelson, *Why Copyright Law Excludes Systems and Processes from the Scope of Its Protection*, 85 TEX. L. REV. 1921, 1961–69 (2007).

34. Pub. L. No. 94-553, 90 Stat. at 2565 ("this title does not afford to the owner of copyright in a work any greater or lesser rights with respect to the use of the work in conjunction with automatic systems capable of storing, processing, retrieving, or transferring informa-

history stated that the statute's extension of existing law was meant simply to give the CONTU Commission time to complete its study and to make recommendations for further legislation, while reiterating Congress's view that software itself was a proper subject of copyright protection.[35]

B. *CONTU and its Aftermath*

CONTU transmitted its final report to Congress on July 31, 1978. Over a strong dissent, the majority recommended that Congress amend the Copyright Act "to make it explicit that computer programs, to the extent that they embody an author's original creation, are proper subject matter of copyright."[36] The Commission found that, because "[t]he cost of developing computer programs is far greater than the cost of their duplication," software would likely be underproduced absent sufficient legal protection or other mechanisms to permit software developers to recoup their costs.[37] This reasoning persuaded the Commission "that some form of protection is necessary to encourage the creation and broad distribution of computer programs in a competitive market."[38]

Although the Commission declared itself to be "unanimous in its belief that computer programs are entitled to legal protection," though "the unanimity has not extended to the precise form that protection should take."[39] Multiple legal doctrines might be invoked to protect software, the Commission continued, including "patent and copyright—exclusively federal statutory methods; trade secret law—derived from statutory and judicial state law; and unfair competition—based on elements of common law and federal statute."[40] The CONTU majority chose copyright law as the proper vehicle partly for historical reasons, and partly by simple process of elimination.

---

tion, or in conjunction with any similar device, machine, or process, than those afforded to works under the law, whether title 17 or the common law or statutes of a State, in effect on December 31, 1977") (codified at 17 U.S.C. § 117 (Supp. I 1977)). Congress's choice to preserve the 1909 Act during the period of the Commission's study was deliberate—the product of compromise between computer industry participants (who sought temporary immunity from liability during the Commission's deliberations) and publishers' representatives (who sought to apply the 1976 statute to electronic reproductions immediately). *See* Melville B. Nimmer, *Foreword: Two Copyright Crises*, 15 UCLA L. REV. 931, 932–33 (1968).

35. H.R. Rep. No. 94-1476, *supra* note 30, at 116 ("The Commission on New Technological Uses is, among other things, now engaged in making a thorough study of the emerging patterns in this field and it will, on the basis of its findings, recommend definitive copyright provisions to deal with the situation"; but "[w]ith respect to the copyrightability of computer programs, the ownership of copyrights in them, the terms of protection, and the formal requirements of the remainder of the bill, the new statute would apply."), 1976 U.S.C.C.A.N. at 5731.

36. CONTU FINAL REPORT, *supra* note 1, at 1.

37. *Id.* at 11.

38. *Id.*

39. *Id.* at 12.

40. *Id.* (footnote omitted).

First, essentially agreeing with the Copyright Office's 1965 rationale as reiterated in the legislative history of the 1976 Act, CONTU declared that the historical expansion of copyrightable subject matter "has already encompassed computer programs" as a form of "literary work" protected "from the moment it is fixed."[41]

Next, the Commission compared the possibility of copyrighting software with other possible forms of legal protection. Patent law, for example, presented an alternative that software developers might find "more attractive than copyright" due to its stronger statutory monopoly.[42] But "[t]he acquisition of a patent, however, is time consuming and expensive. . . and the legal hurdles an applicant must overcome are high."[43] Seemingly hostile language in then-existing Supreme Court case law, in the Commission's view, also cast doubt upon the patent-eligibility of software.[44] The alternative of trade secrecy, the Commission found, fitted poorly with works, such as mass-market commercial software, "that contain the secret and are designed to be widely distributed."[45] The lack of national uniformity in both trade secret and unfair competition law, which ultimately rested upon premises that varied from state to state, also made these options less attractive to the Commission.[46]

The CONTU majority then responded to the criticism that extending copyright protection to software works would slow technological development by requiring wasteful duplication of programmers' efforts. Given the Copyright Act's exclusion of "processes and methods" from the scope of protection, and the long-settled principle that copyright in a work does not prevent reuse of the ideas contained in that work by others, the Commission concluded that "copyright protection for programs does not threaten to block the use of ideas or program language previously developed by others when that use is necessary to achieve a certain result."[47] Responding to criticism that protecting software was tantamount to copyrighting a machine or other functional article, the majority declared that copyright simply meant that computer "users may not take the works of others to operate their machines. . . . one is always free to make the machine do the same thing as it

41. *Id.* at 15 (footnote omitted); *cf. supra* notes 22, 33 and accompanying text.

42. CONTU FINAL REPORT, *supra* note 1, at 16–17.

43. *Id.* at 17. The Commission also noted, a bit obliquely, the risk of possible overprotection of software under patent law, because a patent would bar the reuse by other programmers even of methods and ideas embedded in prior software that copyright law, in contrast, would leave available for others to reuse. *Id.* at 20.

44. *Id.* at 17 ("It is still unclear whether a patent may ever be obtained for a computer program.").

45. *Id.* (footnote omitted).

46. *Id.* at 17–18.

47. *Id.* at 20; *see also id.* at 18–19 (citing 17 U.S.C. § 102(b), and Baker v. Selden, 101 U.S. 99 (1879)); *cf. supra* note 33 (noting divergence of opinion over how much constraining force Section 102(b) actually exerts on the reach of software copyright protection).

would if it had the copyrighted [software] placed in it, but only by one's own creative effort rather than by piracy."[48] Nevertheless, the CONTU majority conceded, "[i]t is difficult, either as a matter of legal interpretation or technological determination, to draw the line between the copyrightable element of style and expression in a computer program and the process which underlies it."[49] This difficulty would come to bedevil the courts called upon to construe the scope of copyright protection in software, as discussed below.

Commissioner John Hersey, joined by Commissioner Rhoda Karpatkin, dissented from the majority's recommendation.[50] Hersey agreed with the majority that "the investment of creative effort in the devising of computer programs does warrant certain modes of protection of the resulting devices[.]"[51] Extending copyright protection to software, however, "would mark the first time copyright had ever covered a means of communication, not with the human mind and senses, but with machines."[52] A computer program, Commissioner Hersey wrote, was "a machine-control element, a mechanical device, having no purpose beyond being engaged in a computer to do mechanical work."[53] That the "machine" being controlled was electronic rather than mechanically based, he believed, made no difference to the essentially functional, utilitarian nature of software. Software was not merely a description of steps that would, if implemented, accomplish a certain result; it was rather a device that itself produced that result: "[w]hen activated, it does the work."[54] Commissioner Hersey also believed that copyright protection was unnecessary to provide an adequate incentive for the creation and dissemination of expressive works.[55] Turning next to the legislative record, Commissioner Hersey disputed the CONTU majority's conclusion that software copyright was already a *fait accompli* under the language

48. *Id.* at 21.

49. *Id.* at 22.

50. Commissioner Melville Nimmer also expressed some sympathy with Hersey's views; and, according to Commissioner Karpatkin, they had also been shared by Commissioner William Dix, who died before the completion of the Commission's inquiry. *See id.* at 26–27 (Nimmer), 37–38 (Karpatkin).

51. *Id.* at 27.

52. *Id.* at 28. Copyright, in Hersey's view, was an appropriate form of protection only for works "intelligible to a human being." *Id.* at 29.

53. *Id.* at 28.

54. *Id.*; *see also id.* ("[p]rinted instructions explain *how* to do something; programs are *able* to do it.") (emphasis in original); *cf. supra* note 4 (noting that this characterization is broadly shared).

55. CONTU FINAL REPORT, *supra* note 1, at 30 ("It appears that the existing network of technological, contractual, nondisclosure, trade-secret, common-law misappropriation, and (in a few instances) patent forms of protection. . . will be wholly adequate, as they apparently have been up to now, to the needs of developers."); *see also, e.g.*, Stephen Breyer, *The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs*, 84 HARV. L. REV. 281, 344 (1970) (emphasizing that early software developers believed copyright protection to be immaterial to their work).

of the 1976 Act.[56] Instead, Commissioner Hersey concluded, Congress should expressly exclude "a computer program in the form in which it is capable of being used to control computer operations" from the scope of subject matter eligible for copyright protection.[57]

Congress adopted the CONTU majority's recommendations, essentially unchanged, in 1980.[58] The new statute added to the Copyright Act a definition of "computer program" that reads today just as written in 1980: "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."[59] And, striking out the 1976 Act's placeholder language preserving the force of the 1909 Act, the 1980 law added two exceptions describing particular circumstances in which the use of a copyrighted work in a computer would be deemed noninfringing.[60] The very brief legislative history explained that the statute "embodies the recommendations of the Commission on New Technological Uses of Copyrighted Works with respect to clarifying the law of copyright of computer software."[61]

Remarkably, however, the 1980 amendment nowhere expressly declares software to be copyrightable, although extending such protection was undoubtedly Congress's intent. The amendment's new definition of "computer program" remains syntactically unmoored from the provisions of the Copyright Act defining copyrightable subject matter. Section 102(a) does not include "computer programs" among its enumerated categories of the subject matter eligible for copyright protection, and even Congress's apparent intention to subsume programs within the statutory definition of "literary works" is not reflected in the language of the amendment.[62] Instead, the 1980 amendment seems to have extended copyright protection to software by implication: the statute as amended declares that computer programs contain "statements or instructions," suggesting (without stating) that they fall within the statute's definition of "literary works" in that they are "expressed

56. CONTU FINAL REPORT, *supra* note 1, at 31 ("The legislative history of the new law can give little comfort to any who would suggest that a thoughtful legislative judgment had been made about the propriety of copyright protection for computer programs."); *cf. supra* note 41 and accompanying text.

57. CONTU FINAL REPORT, *supra* note 1, at 37.

58. Pub. L. No. 96-517, § 10, 94 Stat. 3015, 3028–29 (1980).

59. *Id.* at 3028 (codified at 17 U.S.C. § 101).

60. *Id.* at 3028–29 (codified as amended at 17 U.S.C. § 117); *cf. supra* note 34 and accompanying text.

61. H.R. Rep. No. 96-1307, Part 1, at 23, *reprinted in* 1980 U.S.C.C.A.N. 6460, 6482.

62. *See* 17 U.S.C. §§ 101, 102(a) (2012); *see also* Mark A. Lemley, *Convergence in the Law of Software Copyright*, 10 HIGH TECH. L. J. 1, 6 n.25 (1995) ("CONTU did not, however, add computer programs to the list of protectable works, instead considering them to be both literary and audiovisual works."). Although the conception of software as a form of literary work recurs frequently in the legislative history of the 1976 Act, this assumption appears never to have been reflected in the actual statutory language. *See, e.g.*, *supra* notes 20, 22, 33 and accompanying text.

in words, numbers, or other verbal or numerical symbols or indicia."[63] Again, it seems clear that Congress believed that the 1980 amendments either extended copyright protection to software or reconfirmed that such protection already existed.[64] The legislature, however, surely could have anticipated that extending copyright through an implied analogy, rather than addressing the scope and limitations of copyright forthrightly in statutory text, would cause enduring analytical problems for the courts.

C. *The Early Cases*

The CONTU report's forecast of continuing interpretive difficulty swiftly proved to be more prescient than Congress's hope that the 1980 statute would "clarify the law." To be sure, the literal, exact duplication of a copyrighted program in its entirety posed no great analytical difficulties for the courts.[65] Far more challenging, however, was the question whether infringement resulted when a later program was written to mimic the functionality of an earlier program without copying the earlier program's own source or object code. Patent protection, of course, would bar the second comer from implementing the same functionality as the earlier program; but the more difficult question was whether copyright law would yield the same result.

Courts applying the 1980 statute enunciated an assortment of incompatible standards for measuring the reach of copyright in the nonliteral elements of a computer program, the imitation of which by another programmer would constitute infringement.[66] The Third Circuit in *Whelan Associates,*

---

63. 17 U.S.C. § 101 (definitions of "computer program" and "literary works"). Compare this relatively oblique language with the far more straightforward declaration in the TRIPS Agreement, adopted fifteen years later, that "[c]omputer programs, whether in source or object code, shall be protected as literary works[.]" Agreement on Trade-Related Aspects of Intellectual Property Rights, art. 10(1).

The 1980 revisions to Section 117 also describe the reach of software copyright in a curiously inverted fashion, providing that "it is not an infringement for the owner of a copy of a computer program to make. . . another copy" in some circumstances, inviting the negative inference that copying in other circumstances would be infringing. *See* 17 U.S.C. § 117(a); Stephen Kyle Tapp & Daniel E. Wanat, *Computer Software Copyright Issues: Section 117 and Fair Use*, 22 MEM. ST. U. L. REV. 197, 214 (1992) (recognizing this issue); Richard H. Stern, *Another Look at Copyright Protection of Software: Did the 1980 Act Do Anything for Object Code?*, 3 COMPUTER/L.J. 1, 8 (1981) (same).

64. *See, e.g.*, Williams Elecs., Inc. v. Artic Int'l, Inc., 685 F.2d 870, 875 (3d Cir. 1982) (declaring, perhaps optimistically, that "the copyrightability of computer programs is firmly established after the 1980 amendment to the Copyright Act").

65. *See, e.g.*, Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983).

66. *See, e.g.*, Robert W. Gomulkiewicz, *Legal Protection for Software: Still a Work in Progress*, 8 TEX. WESLEYAN L. REV. 445, 447–49 (2002) (describing different courts' methodologies for addressing nonliteral infringement); Lloyd G. Farr, Sega v. Accolade*: Another Generation of Computer Program Copyright Cases Has Growing Pains*, 27 GA. L. REV. 903, 917–25 (1993) (assessing early post-CONTU decisions).

Inc. v. Jaslow Dental Laboratory, Inc. stated that the relevant question was whether a computer program contained any expressive content not "necessary" to enable the program to serve the purpose for which it was written; if so, such content was protected by copyright:

> Just as *Baker v. Selden* focused on the end sought to be achieved by Selden's book, the line between idea and expression may be drawn with reference to the end sought to be achieved by the work in question. In other words, the purpose or function of a utilitarian work would be the work's idea, and everything that is not necessary to that purpose or function would be part of the expression of the idea.[67]

The test, in the court's view, was whether there were "various means of achieving the desired purpose"; if so, "then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea."[68] Applying this test to the program before it, the court found that the only unprotectable element of the plaintiff's software was the particular purpose for which it was written. Because that purpose could have been implemented in a variety of ways using several software techniques, the particular means chosen by the plaintiff was broadly protectable.[69]

Expressly rejecting *Whelan*, however, the Fifth Circuit in *Plains Cotton Cooperative Association v. Goodpasture Computer Service, Inc.*[70]enunciated a far stricter standard for identifying protectable expressive elements within a computer program. The two programs in *Plains Cotton* shared many organizational similarities, the court recognized, but these were "dictated by the externalities of the cotton market" that both programs were written to serve.[71] The court found that "[t]he record supports the inference that market factors play a significant role in determining the sequence and organization of cotton marketing software, and we decline to hold that those patterns cannot constitute 'ideas' in a computer context."[72] *Plains Cotton* thus narrowed the universe of protected elements within a computer program from the broad view of *Whelan*: elements dictated by "market factors" were un-

---

67. Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1236 (3d Cir. 1986) (citation and footnote omitted; emphasis in original).

68. *Id.*

69. *See id.* at 1236 n.28 ("the idea of the Dentalab program was the efficient management of a dental laboratory (which presumably has significantly different requirements from those of other businesses). 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.").

70. Plains Cotton Coop. Ass'n v. Goodpasture Comput. Serv., Inc. 807 F.2d 1256 (5th Cir. 1987); *see also id.* at 1262 (declining to follow *Whelan*).

71. *Id.* at 1262.

72. *Id.*

protectable even if they could have been implemented by the defendant in ways that differed from the plaintiff's.

After *Whelan* and *Plains Cotton*, the courts struggled for a few years to enunciate a single test that would differentiate the uncopyrightable elements of a computer program from those the law protected. Some courts, following *Whelan*, extended fairly broad copyright protection to computer software.[73] Other cases echoed *Plains Cotton*'s observation that software elements drafted to meet market needs, rather than to serve expressive purposes, would necessarily fall outside the coverage of copyright law.[74] Adding to the confusion, the Fifth Circuit itself seemed to retreat from *Plains Cotton* in later cases.[75]

Commentators, too, were divided. Some argued that the *Whelan* test overprotected software by failing to exclude subsidiary "ideas" embedded within the program (but not commanded by its purpose) from the scope of protection.[76] Others celebrated *Whelan* for providing reassurance to the still-emerging commercial software industry that investments in software development would be rewarded with broad legal protections.[77] Professor Arthur Miller, who served on the CONTU Commission, hailed *Whelan* as the wellspring of all subsequent cases, and attempted to show that even cases that had claimed to differ from *Whelan* were actually consistent with it.[78]

D. *From* Altai *to* Oracle

In *Computer Associates International, Inc. v. Altai, Inc.*, the Second Circuit sought to resolve the *Whelan*/*Plains Cotton* debate by returning to first

---

73. *See, e.g.*, Autoskill Inc. v. National Educational Support Sys., Inc., 994 F.2d 1476, 1495 n.23 (10th Cir. 1993) (rejecting argument that 17 U.S.C. § 102(b) limited the scope of copyright in nonliteral elements of computer software), *rev'd in part on other grounds*, TW Telecom Holdings Inc. v. Carolina Internet Ltd., 661 F.3d 495 (10th Cir. 2011); Lotus Dev. Corp. v. Paperback Software Int'l, 740 F. Supp. 37, 54–58 (D. Mass. 1990).

74. *See* Apple Computer, Inc. v. Microsoft Corp., 799 F. Supp. 1006, 1023–25 (N.D. Cal. 1992); Micro Consulting, Inc. v. Zubeldia, 813 F. Supp. 1514, 1528 (W.D. Okl. 1990).

75. *See* Engineering Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335, 1341–42 (5th Cir. 1994); Kepner-Tregoe, Inc. v. Leadership Software, Inc., 12 F.3d 527, 536 & n.20 (5th Cir. 1994).

76. *See, e.g.*, Samuelson, *supra* note 33, at 1967; Lemley, *supra* note 62, at 10–12; Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 STAN. L. REV. 1045, 1084–85 (1989); J. Dianne Brinson, *Copyrighted Software: Separating the Protected Expression from Unprotected Ideas, A Starting Point*, 29 B.C. L. REV. 803, 850–55 (1988).

77. *See* Anthony L. Clapes, *et al.*, *Silicon Epics and Binary Bards: Determining the Proper Scope of Copyright Protection for Computer Programs*, 34 UCLA L. REV. 1493 (1987).

78. *See* Miller, *supra* note 31, at 997–1013. Professor Miller's article recognized forthrightly that his CONTU service had led him to rethink his early opposition to copyrighting software, although his quotations from his own prior testimony to Congress selectively emphasized his concerns concerning educational use of copyrighted works rather than copyrighting software as such. *Compare id.* at 981–82 n.10 *with* Miller, *supra* note 27.

principles.[79] The "question of whether and to what extent the 'non-literal' aspects of a computer program, that is, those aspects that are not reduced to written code, are protected by copyright law," in the court's view, could be answered by referring to "long-standing doctrines of copyright law" based on Congress's apparent command to treat software as a form of protected "literary work."[80] Treating software as a form of "literary work" implicated copyright's idea/expression dichotomy and the conventional limits on copyright protection for useful articles or functional works, as well as the copyright statute's express exclusion of "process[es], procedure[s], system[s], or method[s] of operation" from the scope of protected subject matter.[81] The court found *Whelan* to provide limited guidance on those issues because it interpreted the unprotected "idea" of a computer program too narrowly. In the court's words:

> The leading commentator in the field has stated that "[t]he crucial flaw in [*Whelan*'s] reasoning is that it assumes that only one 'idea,' in copyright law terms, underlies any computer program, and that once a separable idea can be identified, everything else must be expression." 3 Nimmer § 13.03(F), at 13-62.34. This criticism focuses not upon the program's ultimate purpose but upon the reality of its structural design. As we have already noted, a computer program's ultimate function or purpose is the composite result of interacting subroutines. *Since each subroutine is itself a program, and thus, may be said to have its own "idea,"* Whelan*'s general formulation that a program's overall purpose equates with the program's idea is descriptively inadequate*.[82]

Instead, the *Altai* court enunciated "a three-step procedure, based on the abstractions test utilized by the district court, in order to determine whether the non-literal elements of two or more computer programs are substantially similar."[83] The court's three steps were:

1. **Abstraction.** First, the court must "dissect the allegedly copied program's structure and isolate each level of abstraction contained within it. This process begins with the code and ends with an articulation of the program's ultimate function. Along the way, it is necessary essentially to retrace and map each of

79. Comput. Ass'n Int'l v. Altai, 982 F.2d 693, 697-01 (2d Cir. 1992).
80. *Id.* at 702.
81. *Id.* at 702–07.
82. *Id.* at 705 (emphasis added).
83. *Id.* at 706.

the designer's steps—in the opposite order in which they were taken during the program's creation."[84]

2. **Filtration.** Next, the *Altai* court directed, courts must undertake a "'successive filtering method'" to search "each level of abstraction" identified in the preceding step for subject matter lying outside the scope of the author's copyright.[85] The filtration process excluded several types of material from the scope of the program's copyright. First, the court said, "elements dictated by efficiency" must be filtered out. Those elements lay on the "idea" side of copyright's idea/expression dichotomy and were also subject to the copyright's doctrine of merger, which denies protection to expression that is necessary to permit an idea to be communicated. Quoting the CONTU report, the court explained that:

> [C]opyrighted language may be copied without infringing when there is but a limited number of ways to express a given idea. . .. In the computer context, this means that *when specific instructions, even though previously copyrighted, are the only and essential means of accomplishing a given task, their later use by another will not amount to infringement*.[86]

Even *Whelan*, the *Altai* court recognized, had effectively endorsed application of the merger doctrine in principle, despite its reliance on the possibility of expressive variation in programming code.[87] Second, *Altai* directed courts to "filter out"

---

84. *Id.* at 707. As an illustration of this approach, the court approvingly quoted from a student author's description of the different levels of abstraction that coexisted within software works:

> At the lowest level of abstraction, a computer program may be thought of in its entirety as a set of individual instructions organized into a hierarchy of modules. At a higher level of abstraction, the instructions in the lowest-level modules may be replaced conceptually by the functions of those modules. At progressively higher levels of abstraction, the functions of higher-level modules conceptually replace the implementations of those modules in terms of lower-level modules and instructions, until finally, one is left with nothing but the ultimate function of the program. . .. A program has structure at every level of abstraction at which it is viewed. At low levels of abstraction, a program's structure may be quite complex; at the highest level it is trivial.

Id. (quoting Steven R. Englund, Note, *Idea, Process, or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Computer Programs*, 88 MICH. L. REV. 866, 897–98 (1990)).

85. *Altai*, 982 F.2d at 701.

86. *Id.* at 709 (emphasis added).

87. *See id.* at 708 ("While, hypothetically, there might be a myriad of ways in which a programmer may effectuate certain functions within a program, —i.e., express the idea em-

> "elements dictated by external factors"—a term that *Altai* used more broadly than *Plains Cotton*'s reference to uncopyrightable program components driven by "market factors."[88] In the context of software, this meant that infringement could not be based upon similarities between two programs that were attributable to "(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."[89] Finally, the court directed, "elements taken from the public domain" must be "filtered out" of the plaintiff's work before comparing it with the defendant's work. Public domain material "is free for the taking and cannot be appropriated by a single author even though it is included in a copyrighted work."[90]
>
> 3. **Comparison.** After "filtering out" unprotected content at each level of abstraction contained within the work, the *Altai* court concluded,
>
> > there may remain a core of protectable expression. In terms of a work's copyright value, this is the golden nugget. At this point, the court's substantial similarity inquiry focuses on *whether the defendant copied any aspect of this protected expression*, as well as an assessment of the copied portion's relative importance with respect to the plaintiff's overall program.[91]

The *Altai* court's three-step analysis drew mostly, but not uniformly, favorable reviews from copyright experts. The Nimmer treatise, which ar-

---

bodied in a given subroutine—efficiency concerns may so narrow the practical range of choice as to make only one or two forms of expression workable options.") (citation omitted).

88. *See id.* at 709–10; *cf. supra* note 72 and accompanying text.

89. *Id.*

90. *Id.* at 711. As an example, the court cited "elements of a computer program that have entered the public domain by virtue of freely accessible program exchanges and the like." *Id.* One might quarrel with the court's perception of an equivalency between a program being "freely accessible" and the program's status as a public domain work; under the 1976 Act, copyright protection attaches the moment a work is fixed in a tangible medium of expression, and the fact that the work may thereafter become widely available is irrelevant to its public-domain status. Free and open-source software works, to be sure, are distributed under licenses that grant broad reuse rights (similar to public-domain works), but even open-source software remains protected by copyright, and reuse of the software beyond the extent permitted by the license may be infringing. *See* Jacobsen v. Katzer, 535 F.3d 1373 (Fed. Cir. 2008); *see also infra* notes 222–224 and accompanying text (discussing free and open-source software).

91. *Altai*, 982 F.2d at 710 (citation omitted) (emphasis added).

ticulated its own version of the analysis from which *Altai* drew, championed the court's reasoning, noting that "[o]ther courts have subsequently followed *Altai*'s lead, so that this filtration test may now be regarded as the dominant, albeit not universal, standard."[92] The Patry treatise, on the other hand, criticized the "formidable" burdens the *Altai* analysis requires of nonspecialist district judges, while also emphasizing tensions between prior case law on the one hand, and *Altai*'s use of both copyright's merger doctrine and amorphous concepts such as "efficiency" on the other.[93] The *Altai* analysis has been enduringly influential, however, both in the courts and in the academic analysis of software copyright issues.[94]

The First Circuit added its own criticism of *Altai* in *Lotus Development Corp. v. Borland International, Inc.*[95] The question in *Lotus* was whether the names and hierarchical structure of menu commands in a spreadsheet program were themselves protectable under copyright, a question the *Lotus* court answered in the negative.[96] The court rejected Lotus's request to conduct an *Altai* abstraction-filtration-comparison analysis of its menu command names, for essentially two reasons. First, the court declared, because *Lotus*, unlike *Altai*, was a case involving allegations of literal infringement of the copyrighted work, the *Altai* analysis (which allowed a finding of infringement even in the absence of literal copying) was inapposite.[97] Second, and more importantly, the *Lotus* court warned, *Altai* itself presupposed the existence of *some* level of copyrightable expression within the work—a presumption that may lead courts astray when the question before them was whether the plaintiff's work was copyrightable at all. In the court's words:

> In fact, we think that the *Altai* test in this context may actually be misleading because, in instructing courts to abstract the various

92. 4 NIMMER ON COPYRIGHT § 13.03[F] (2015) (internal quotations omitted) (footnotes omitted).

93. 2 PATRY ON COPYRIGHT § 3:87 (2015); *see also* William F. Patry, *Copyright and Computer Programs: It's All in the Definition*, 14 CARDOZO ARTS & ENT. L.J. 1, 55 (1996) (concluding that *Altai*'s analysis "is overly complicated, and has been of practical assistance to only one type of person, the expert witness, whose services are essential for the detailed analysis of the programmer's creative process that the court's test requires.").

94. *See, e.g.*, Pamela Samuelson, *A Fresh Look at Tests for Nonliteral Copyright Infringement*, 107 NW. U. L. REV. 1821, 1843 (2013) (footnote omitted) (showing that "[s]everal courts have, in fact, adopted an *Altai*-like filtration test in thin copyright nonsoftware cases."); Lemley, *supra* note 62, at 14–17.

95. Lotus Development Corp. v. Borland International, Inc., 49 F.3d 806 (1st Cir. 1995).

96. *See id.* at 815–19 (reasoning that Lotus's menu command hierarchy was a "method of operation" of Lotus's program and therefore expressly excluded from copyright protection under 17 U.S.C. § 102(b)).

97. *See id.* at 815 ("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.").

> 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*.[98]

Lotus's critique of *Altai* thus parallels *Altai's* own criticism of *Whelan* in an important respect. In each instance, the later court said, the earlier court's analysis leads to potential overprotection of software under copyright law by confining its analysis in a way that disregards some of the law's built-in limiting principles (the idea-expression dichotomy in *Whelan*, according to *Altai*; and the statutory exception for "methods of operation" in *Altai*, according to *Lotus*).

Regardless of these criticisms of *Altai*, as Professor Samuelson has observed, "[a]pplication of [*Altai*'s] abstraction-filtration-comparison test generally results in programs having thin copyright protection."[99] "After *Altai*, courts became more openly skeptical about claims of copyright protection for the 'look and feel' of programs, as such claims typically sought to protect the now unprotected utilitarian aspects of programs."[100] The comparatively limited scope of copyright protection available post-*Altai* coincided with the Federal Circuit's rising acceptance of the patent alternative, which both that court and the CONTU report had previously rejected.[101]

More recently, however, another court has given software far more expansive copyright protection than the *Altai* analysis might suggest is appropriate. *Oracle America, Inc. v. Google Inc.*[102] involved a claim of copyright infringement in the "declaring code" for thirty-seven software packages written by Oracle's predecessor-in-interest for use by programmers developing software in the Java programming language. Some of the packages provided "core" functionality that programmers were required to adopt "in

98. *Id.* (emphasis added).

99. Pamela Samuelson, *The Uneasy Case for Software Copyrights Revisited*, 79 GEO. WASH. L. REV. 1746, 1770 (2011).

100. *Id.* at 1771 (footnote omitted).

101. *See id.* at n.213 & 1775.

102. Oracle America, Inc. & Google Inc., 750 F.3d 1339 (Fed. Cir. 2014) (hereinafter *Oracle II*).

order to make any worthwhile use of the [Java] language."[103] The challenged "declaring code" for each of the packages supplied with Java "introduces the method body and specifies very precisely the inputs, name, and other functionality" provided by those packages.[104] Each of the Java packages at issue also included lengthier "implementing code" that actually "gives the computer the step-by-step instructions for carrying out the declared function,"[105] but the implementing code was not in issue in the parties' dispute; Google wrote its own code to implement the functionality of the Java packages, but gave its counterpart the same method and variable names defined in Oracle's own Java "declaring code."[106] A jury found that Google had infringed Oracle's copyright in the Java API declaring code, but the district judge partially granted Google's motion for judgment as a matter of law on the grounds that Oracle's software was not protected by copyright.[107]

The Court of Appeals reversed. It agreed with the parties that the Second Circuit's *Altai* opinion supplied the governing analytical framework.[108] The Federal Circuit's version of the *Altai* inquiry, however, seems quite different from *Altai*'s own. In particular, *Oracle* treated the "filtration" analysis as relevant only to the question of infringement, not as to the reach of copyright protection in the plaintiff's program.[109] (The court's opinion does not

---

103. *Id.* at 1349 (quoting Oracle America, Inc. v. Google Inc., 872 F. Supp. 2d 974, 982 (N.D. Cal. 2012) (hereinafter *Oracle I*) *aff'd in part and rev'd in part,* 750 F.3d 1339 (Fed. Cir. 2014)).

104. *Id.*

105. *Id.* at 1349.

106. The court stated:

> Google copied the declaring source code from the 37 Java API packages verbatim, inserting that code into parts of its Android software. In doing so, Google copied the elaborately organized taxonomy of all the names of methods, classes, interfaces, and packages—the overall system of organized names—covering 37 packages, with over six hundred classes, with over six thousand methods. . .. It is undisputed, however, that Google wrote its own implementing code. . ..

Id. at 1350–51 (quoting *Oracle II*, 872 F. Supp. 2d at 999.).

107. *Oracle II*, 872 F. Supp. 2d at 974. The parties had agreed to have the question of copyrightability *vel non* decided by the district judge; the jury, accordingly, was instructed "to assume that the structure, sequence, and organization of the 37 API packages was copyrightable." *Oracle I*, 750 F.3d at 1351. This procedural division of responsibility seems in some respects to have clouded the court's analysis on appeal, for some of the issues the parties had apparently consented to have the district judge resolve were instead declared on appeal to present jury issues.

108. The Federal Circuit applied Ninth Circuit law to the copyright issues in the parties' cross-appeals. *Oracle II*, 750 F.3d at 1353. It then noted that the Ninth Circuit had embraced the *Altai* abstraction-filtration-comparison analysis in software copyright cases. *Id.* at 1357 (citing, *inter alia*, Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1525 (9th Cir. 1992)).

109. *See id.* at 1358. *Altai* itself did not appear to require evaluation of the defendant's own work, as distinct from the plaintiff's, as part of the filtration analysis, although the court's opinion is less than perfectly clear on this point. *See supra* notes 85–90 and accompanying text. Some other courts, too, require software copyright holders to identify with particularity

indicate whether the jury, which was tasked with resolving the infringement issue, was instructed that it must consider the doctrines, such as merger and *scènes-à-faire*, that the appeals court declared to be relevant only at that stage of the analysis.) Indeed, the crucial element of the *Oracle* court's analysis derives not from *Altai*, but from the Third Circuit's *Whelan* decision, which *Altai* expressly rejected: as *Whelan* did, the *Oracle* court declares software broadly copyrightable because of the possibility that a programmer could have employed alternative forms of expressing the ideas embedded therein.[110]

Oracle discussed many of the same copyright principles that the *Altai* court had relied on. In virtually every case, however, the *Oracle* court concluded that those principles either did not apply or were relevant only to the question of infringement rather than to the reach of copyright protection. The doctrine of merger, for example, which both *Altai* and the CONTU Report had cited as a basis for withholding copyright protection from "specific [software] instructions. . . [that] are the only and essential means of accomplishing a given task," was irrelevant because Oracle could have chosen many different names for the methods and variables defined in its Java API packages.[111] Similarly, copyright's rule against extending protection to individual words and short phrases, the court reasoned, could not limit the scope of copyright in Oracle's declaring code because its size—7,000 lines in total—was far longer than works previously denied protection on this basis.[112] The doctrine of *scènes-à-faire*, which *Altai* had used to justify withholding copyright protection from several different aspects of computer programs, failed to protect Google essentially because the question had not been adequately briefed, and in addition because of the possibility that Oracle might have expressed its declaring code in many different ways.[113]

---

those portions of their programs that are protectable (and not excluded from protection by doctrines such as merger or *scènes-à-faire*) prior to engaging in any comparison with the alleged infringing work. *See, e.g.*, Paycom Payroll, LLC v. Richison, 758 F.3d 1198, 1205–08 (10th Cir. 2014) (even analysis of literal copying must begin by "abstracting" and filtering out those portions of the copied code that are unprotectable); Automated Solutions Corp. v. Paragon Data Sys., 756 F.3d 504, 518–21 (6th Cir. 2014).

110. *Compare supra* notes 67–69 and accompanying text *with Oracle II*, 750 F.3d at 1361 ("[t]he evidence showed that Oracle had 'unlimited options as to the selection and arrangement of the 7000 lines Google copied' ") (quoting Brief for Appellant at 50, *Oracle II*, 750 F.3d 1339 (Nos. 2013–1021, 2013–1022); *id*. at 1368 (finding Oracle's declaring code copyrightable because "the declaring code could have been written and organized in any number of ways"); *cf. supra* notes 86–87 and accompanying text (summarizing *Altai* court's critique of *Whelan* on this point).

111. *See Oracle II*, 750 F.3d at 1360-61.

112. *See generally id.* at 1362–63. The court at this point in its opinion appears to treat Oracle's Java declaring code as a single work of 7,000 lines, rather than 7,000 declarations each expressed in one line of code. *See id.* This characterization would have been highly relevant to the quantum of damages if Oracle had prevailed in the lawsuit. *See* 17 U.S.C. § 504(c)(1) (2012).

113. *See Oracle II*, 750 F.3d at 1363–64; *cf. supra* note 89 and accompanying text.

Although the district court had also relied on the First Circuit's decision in *Lotus*, the appellate court found *Lotus* to be incompatible with *Altai*—although it did so without addressing the reasons *Lotus* gave for not using the abstraction-filtration-comparison analysis to evaluate a menu command hierarchy.[114] Finally, the appeals court declared "irrelevant to copyrightability" Google's arguments that copying of the Java API declaring code was essential to create a product that was interoperable with the large installed Java code base.[115]

Oracle remains a fairly recent decision that has yet to be broadly applied by other courts or treated in depth by commentators.[116] Following a new trial on remand, a jury found Google's use to be noninfringing under the fair use doctrine,[117] prompting a new appeal by Oracle that remains pending at the time of this writing.[118] The previous appellate opinion's impact on the law and on software development, accordingly, remains uncertain.[119] It is surely

---

114. *See Oracle II*, 750 F.3d at 1365–66; *cf. supra* note 98 and accompanying text. Indeed, the *Oracle* case involved the very scenario that *Lotus* warned would lead courts to overprotect software under copyright law, making the *Oracle* opinion's decision not to even describe (much less to distinguish) the *Lotus* opinion's reasoning on this point doubly puzzling.

115. *Oracle II*, 750 F.3d at 1368. *But see* Clark D. Asay, *Copyright's Technological Interdependencies*, 18 STAN. TECH. L. REV. 189, 232–33 (2015) (noting that prior case law appears to treat interoperability as bearing on the copyrightability of software and labeling this a "key, unresolved issue" stemming from the *Oracle* decision).

116. A thorough descriptive summary of the decision is available in *Survey of Additional IP and Technology Law Developments*, 30 BERKELEY TECH. L.J. 1317, 1351–55 (2015). Predominantly critical evaluations of the appeals court's analysis are available in: Pamela Samuelson, *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, 31 BERKELEY TECH. L.J. 1215, 1255-57 (2017) (criticizing *Oracle* for, inter alia, misunderstanding Google's argument against copyrightability of the Java API as an argument against copyrightability of all software code and failing to defer to lower court's factual determinations); Jonathan Band, *The Protectability of Application Program Interfaces: Oracle America v. Google*, 59 J. OF THE JAPANESE GROUP OF THE INT'L ASS'N FOR THE PROTECTION OF INTELL. PROP. 2 (Oct. 2014). Candor requires me to acknowledge my joining Professor Samuelson's amicus brief in support of Google's unsuccessful petition for a writ of certiorari from the Federal Circuit's decision. *See* Google Inc. v. Oracle Am., Inc., 135 S. Ct. 2887 (2015) (denying cert. to *Oracle II*, 750 F.3d 1339).

117. *See* Joel Rosenblatt, *Google Beats Oracle $9 Billion Copyright Suit Over Use of Java to Build Android*, 92 PAT. TRADEMARK & COPYRIGHT J. (BNA) No. 2264, at 339 (June 3, 2016).

118. *See* Joe Mullin, *It's official: Oracle will appeal its "fair use" loss against Google*, ARSTECHNICA (Oct. 27, 2016), https://arstechnica.com/tech-policy/2016/10/its-official-oracle-will-appeal-its-fair-use-loss-against-google/, archived at [perma.cc/5LB8-HFAL].

119. *See, e.g.*, Clark D. Asay, *Oracle v. Google and Software's Copyright Anticommons*, 66 EMORY L.J. 265, 304 (2016), (arguing that impact of court's decision will make collaborative innovation and interoperability among heterogeneous software products more difficult by giving one company control over the "technical vocabulary" necessary to develop compatible products); JONATHAN BAND, INTERFACES ON TRIAL 3.0: ORACLE AMERICA V. GOOGLE AND BEYOND 31-36 (2016) (arguing that Federal Circuit misread Ninth Circuit's case law on interoperability, the First Circuit's *Lotus v. Borland* decision, and the Federal Circuit's own DMCA precedents); Tyler J. Demasky, *Recent Development, Oracle v. Google: Setting a Standard or*

not too early to conclude, however, that *Oracle*'s resuscitation of the *Whelan* analysis—in which the scope of software copyrightability turns not upon functional considerations or market needs served by the program but upon whether a programmer could have used different software code to implement those same functional characteristics—will, if followed by other courts, give software much more expansive copyright protection than the still-dominant *Altai* analysis requires. Almost forty years after the CONTU Report, the courts have made little progress in delimiting the precise scope of copyright protection that applies to such functional "literary" works.

II. Software Patent Law

A. *Early Cases on Software Patenting*

The governing U.S. Patent Act of 1952 emerged in a largely pre-software world, and the possibility that patent protection might be extended to computer software caused Congress none of the intellectual consternation that would accompany the software copyright debate a generation later. Section 101 of the patent statute provides, today as ever, that "[w]hoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title."[120] Those "conditions and requirements" demand that the invention be useful (meaning, successfully accomplishing its intended purpose), novel (that is, previously unknown), nonobvious (that is, not merely a trivial or inconsequential improvement over the prior art), and that it be adequately disclosed in a valid application for a patent.[121]

A well-known trilogy of early Supreme Court cases under the 1952 Act suggested that patent law presented substantial obstacles to the legal protection of computer software; indeed, the earliest of these cases led CONTU to reject software patentability.[122] In *Gottschalk v. Benson*, the Court ruled that

---

Handicapping an Industry?, 18 N.C. J.L. & Tech. Online 1, 28–32 (2016) (arguing that norms and settled practices of software industry validate the type of API reuse in which Google engaged); Peter S. Menell, *API Copyrightability Bleak House: Unraveling the* Oracle v. Google *Jurisdictional Mess*, 31 Berkeley Tech. L.J. 1515, 1562 (2016) (arguing that Federal Circuit misinterpreted both § 102(b) and the Ninth Circuit precedents it purported to apply); Daria Vasilescu-Palermo, *APIs and Copyright Protection: The Potential Impact on Software Compatibility in the Programming Industry*, 16 J. Marshall Rev. Intell. Prop. L. 153, 168 (2016) ("Holding that APIs are entitled to copyright protection may affect the interoperability of computer programs and hinder innovation in the field of computer programming"); Lothar Determann & David Nimmer, *Software Copyright's* Oracle *from the Cloud*, 30 Berkeley Tech. L.J. 161, 170 (2015) (if future cases follow *Oracle*'s analysis, "defendants will have to prove fair use, merger, or scènes à faire in order to vindicate copying of interfaces, lock-out codes, and other gateways to interoperability").

120. 35 U.S.C. § 101 (2012).

121. 35 U.S.C. §§ 101–103, 111–112 (2012).

122. *See supra* notes 43–44 and accompanying text.

a process for converting binary-coded decimal ("BCD") numerals into a "pure" binary form representing the same numerical value, despite its usefulness in digital computing, could not be patented, on the grounds that permitting the patent would essentially monopolize the idea of numerical conversion.[123] In *Parker v. Flook*, the Court extended *Benson* to deny patent protection to a new way of computing "alarm limits" that were constantly updated during a catalytic conversion process, on the grounds that, despite the inclusion of "post-solution activity" (the actual use of the result of the computation in the control of the ongoing catalytic conversion process), the claimed invention amounted to no more than a new mathematical formula, the unpatentability of which was clear under existing law.[124] In contrast, in *Diamond v. Diehr*,[125] the Court upheld a patent covering an improved rubber-curing process that incorporated a computer-driven temperature control mechanism. The *Diehr* Court distinguished *Benson* and *Flook* on the grounds that the claimed invention involved more than the mere computation of a number and that the patent would not effectively prevent others from using the same mathematical formula.[126]

Summarizing the impact of the Supreme Court's *Benson–Flook–Diehr* trilogy on patent eligibility, another court wrote:

> The goal is to answer the question "What did applicants invent?" If the claimed invention is a mathematical algorithm, it is improper subject matter for patent protection, whereas if the claimed invention is an application of the algorithm, § 101 will not bar the grant of a patent.[127]

---

123. Gottschalk v. Benson, 409 U.S. 63, 71–72 (1972) ("It is conceded that one may not patent an idea. But in practical effect that would be the result if the formula for converting BCD numerals to pure binary numerals were patented in this case. The mathematical formula involved here has no substantial practical application except in connection with a digital computer, which means that if the judgment below is affirmed, the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself.").

124. Parker v. Flook, 437 U.S. 584, 590 (1978). As in *Benson*, the Court feared that a contrary ruling would essentially confer a patent monopoly over an abstract idea, noting that:

> The notion that post-solution activity, no matter how conventional or obvious in itself, can transform an unpatentable principle into a patentable process exalts form over substance. A competent draftsman could attach some form of post-solution activity to almost any mathematical formula; the Pythagorean theorem would not have been patentable, or partially patentable, because a patent application contained a final step indicating that the formula, when solved, could be usefully applied to existing surveying techniques.

125. Diamond v. Diehr, 450 U.S. 175 (1981).

126. *See id.* at 185–88.

127. *In re* Abele, 684 F.2d 902, 907 (C.C.P.A. 1982), abrogated by *In re* Bilski, 545 F.3d 943 (Fed. Cir. 2008).

This distinction, CONTU believed, implied "that it would be difficult for any applicant to secure a patent in a program, since novel and useful mathematical formulas may not be patented and since useful 'post-solution applications' of them meet the same fate."[128] Post-*Diehr* case law appeared to substantiate this concern, as attempts to patent inventions incorporating software components foundered on the lack of any identifiable *physical* effect or output.[129]

B. Alappat*,* State Street*, and the Software Patent Boom*

A series of Federal Circuit decisions in the mid-1990s, however, took a very different view and greatly expanded the scope of patent protection for software-related inventions. Just as the reach of software copyright began to wane in the wake of decisions such as *Altai*, patent protection expanded to fill the gap.

In *In re Alappat*,[130] the court of appeals reversed the USPTO's conclusion that a computer-implemented invention was patent-ineligible subject matter. The invention was a rasterizer that converted waveform data to pixel values that could be displayed on a computer screen.[131] Such rasterizers were known in the prior art, and the claimed invention was implemented entirely using "device[s] known in the electronics arts before Alappat made his invention."[132] The claimed invention's improvement over the prior art, in the USPTO's view, lay in its use of a novel "mathematical algorithm for computing pixel information."[133] The USPTO reasoned that a mathematical formula for computing pixel values was ineligible subject matter for patenting under the reasoning of authorities such as *Gottschalk v. Benson*.[134]

In a fractured *en banc* ruling that produced seven separate opinions (with three concurrences and three dissents), the Federal Circuit declared the invention to be patent-eligible. The majority ruling reasoned that the invention amounted to the creation of a new and patent-eligible "machine," within the meaning of Section 101, "because a general purpose computer in effect becomes a special purpose computer once it is programmed to perform par-

128. *See generally* CONTU FINAL REPORT, *supra* note 1, at 17.

129. *See, e.g.*, *In re* Schrader, 22 F.3d 290, 293–94 (Fed. Cir. 1994); *In re* Grams, 888 F.2d 835, 840 (Fed. Cir. 1989). The relative paucity of case law in this era is partly a reflection of the fact that, for a variety of practical and philosophical considerations, many producers of software did not even seek patent protection at all. Gomulkiewicz, *supra* note 66, at 449–50.

130. *In re* Alappat, 33 F.3d 1526 (Fed. Cir. 1994) (en banc), *abrogated by In re* Bilski, 545 F.3d 943 (Fed. Cir. 2008).

131. *Id.* at 1538–39 (referencing pertinent claim language).

132. *Id.* at 1539 (emphasis omitted).

133. *Id.* (internal quotations omitted); *see also id.* at 1539–40 ("[W]hen the claim is viewed without the steps of this mathematical algorithm, no other elements or steps are found.") (internal quotations omitted.)

134. *See generally id.* at 1542 (discussing USPTO's reasoning); *cf. Gottschalk* 409 U.S. at 71–72.

ticular functions pursuant to instructions from program software."[135] The *Benson* rule against extending patentability to what was, in substance, merely a mathematical algorithm presented no obstacle in the majority's view, for even though the invention "transform[ed] one set of data to another through what may be viewed as a series of mathematical calculations," the patent language "is not 'so abstract and sweeping' that it would 'wholly preempt' the use of any apparatus employing the combination of mathematical calculations recited therein."[136] The rasterizer claimed in the patent was "not a disembodied mathematical concept which may be characterized as an 'abstract idea,' but rather a specific machine to produce a useful, concrete, and tangible result."[137]

Four years later, the court in *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*[138]extended *Alappat* still further, effectively declaring that the "mathematical algorithm" exception to patentability could not be invoked to limit patents for software. The invention in *State Street* was a computerized system for mutual fund price accounting and administration. The type of calculations at issue could be performed by hand, but the addition of a computer made it possible for share prices to be determined much more rapidly after the close of each trading day.[139] The use of a computer to perform such calculations was not, in the court's view, subject to the "mathematical algorithm" exception to patent-eligibility that doomed the patents in *Benson* and *Flook*. Instead, citing *Alappat*, the court declared that:

> [T]he transformation of data, representing discrete dollar amounts, by a machine through a series of mathematical calculations into a final share price, constitutes a practical application of a mathematical algorithm, formula, or calculation, because it produces "a useful, concrete and tangible result"—a final share price momentarily fixed for recording and reporting purposes and even accepted and relied upon by regulatory authorities and in subsequent trades.[140]

The combination of *Alappat* and *State Street*, along with a burgeoning software sector in the U.S. economy in the 1990s, produced a sizable increase in the number of patents covering software inventions issued by the USPTO. The number of issued patents within the computing field grew from fewer than 5,000 in 1995, to nearly 12,000 in 1998, to over 16,000 in 2004,

---

135. *Alappat*, 33 F.3d at 1545.
136. *Id.* at 1544 (quoting *Benson*).
137. *Id.*
138. State St. Bank & Tr. Co. v. Signature Fin. Grp., 149 F.3d 1368, 1369 (Fed. Cir. 1998).
139. *See id.* at 1371.
140. *Id.* at 1373.

25,000 in 2009, and nearly 55,000 in 2014.[141] The Federal Circuit's reasoning that software was patent-eligible because it produced a "useful, concrete, and tangible result" met with criticism in the United States,[142] and many foreign jurisdictions continued to insist that, as a mere exercise in applied mathematics, software standing alone was ineligible for patent protection.[143] In the United States, however, patent-law issues began to loom very large for participants in the software industry, with even established and reputable developers cited for infringement of comparatively obscure inventions.[144]

C. *The Leahy-Smith America Invents Act of 2011*

As Congress began debating significant patent reform legislation in the middle of the first decade of the 2000s, the massive growth in software patenting occasionally became a subject of legislative concern. Patents covering business methods enabled by use of a computer, the sort of invention at issue in the *State Street* case, came in for particular scrutiny. Representative Joseph Crowley (D-NY) complained about "nuisance patents" that allowed a patentee to "sue the Red Cross for soliciting charitable contributions on the Internet, claiming that his patent covers this entire field."[145] Senator Charles Schumer (D-NY) criticized a patent over "double clicking" on a computer.[146] Academics, technologists, and popular commentators, too, questioned whether the incentive of patent protection was appropriate or necessary for software development.[147] Part of the problem, some suggested, was that

---

141. *See* Patent Counts By Class By Year, CY 1977–2015, http://www.uspto.gov/web/offices/ac/ido/oeip/taf/cbcby.htm (accessed Apr. 7, 2016) [hereafter USPTO PATENT COUNTS]. The referenced totals include all patents issued in Classes 700 through 726.

142. *See, e.g.*, Laboratory Corp. of Am. Holdings v. Metabolite Labs., Inc., 548 U.S. 124, 136–37 (2006) (Breyer, J., dissenting from dismissal of certiorari) (*State Street* "does say that a process is patentable if it produces a 'useful, concrete and tangible result.' But this Court has never made such a statement and, if taken literally, the statement would cover instances where this Court has held the contrary.") (citing, inter alia, *Benson* and *Flook*).

143. For example, legislation to ban the issuance of patents over software was proposed in 2013 in Germany and New Zealand. *See* Joe Mullin, *In historic vote, New Zealand bans software patents*, ARS TECHNICA, (Aug. 28, 2013), http://arstechnica.com/tech-policy/2013/08/in-historic-vote-new-zealand-bans-software-patents/; AA Thornton & Co., *German Proposal to Limit Software Patenting*, (July 8, 2013), http://www.lexology.com/library/detail.aspx?g=201b37de-d4f2-4b23-88a2-ddf53e5e3ab9.

144. *See, e.g.*, i4i LP v. Microsoft Corp., 598 F.3d 831 (Fed. Cir. 2010) (Defendant's flagship Microsoft Word product infringed plaintiff's patent on editing XML document markup language), *aff'd*, 131 S. Ct. 2238 (2011).

145. 157 CONG. REC. H4496 (daily ed. June 23, 2011).

146. 157 CONG. REC. S5436–37 (daily ed. Sept. 8, 2011). The apparent reference was to U.S. Patent No. 6,727,830, issued April 27, 2004 to Microsoft Corporation.

147. *See, e.g.*, RICHARD M. STALLMAN, THE DANGER OF SOFTWARE PATENTS, IN FREE SOFTWARE, FREE SOCIETY: SELECTED ESSAYS OF RICHARD M. STALLMAN 95 (Joshua Gay, ed., 2002); JAMES BOYLE, SHAMANS, SOFTWARE, AND SPLEENS: LAW AND THE CONSTRUCTION OF THE INFORMATION SOCIETY 132–35 (1996); Jay Dratler, Jr., *Does Lord Darcy Yet Live?: The Case Against Software and Business-Method Patents*, 43 SANTA CLARA L. REV. 823, 853–71 (2003).

software had been deemed to be unpatentable subject matter for so long that there was essentially no public record of the prior art against which the USPTO could evaluate any individual patent application, resulting in the issuance of an excessive number of substantively weak patents.[148]

With sustained Congressional attention returning to the subject of intellectual property for the first time in many years, the time might have appeared ripe for the legislature to revisit the CONTU Report's assumptions about the proper form of intellectual property protection for computer software. The opportunity, however, passed largely unnoticed. The final version of the resulting patent reform legislation, enacted as the Leahy-Smith America Invents Act of 2011 ("AIA"),[149] included several provisions that dealt obliquely with software patenting, but none addressing the issue directly. The bill expanded the prior commercial use defense, which had originally been enacted to protect accused infringers of business-method patents, to cover other types of patents, including software.[150] A new limitation on the patenting of tax strategies was expressly declared in the bill not to invalidate patents on tax preparation software.[151] The legislation included a new transitional procedure for reviewing business method patents,[152] but several legislators stated that the new review process would apply only to patents claiming business methods as such, not software patents.[153] Confusingly, despite Congress's apparent belief that the post-grant review procedure would not apply to patents claiming software as such, the Obama Administration cited the new review process as a reason for rejecting an online citizen peti-

148. *See* ADAM B. JAFFEE & JOSH LERNER, INNOVATION AND ITS DISCONTENTS 200–02 (2004); BOYLE, *supra* note 147, at 134 ("Partly because the Patent Office has no expertise in software matters, it is widely thought to grant too many patents and to cover processes that had previously been considered to be part of the public domain.").

149. Leahy-Smith America Invents Act, Pub. L. No. 112-29, 125 Stat. 284 (2011) (codified in scattered sections of 35 U.S.C.) [hereafter AIA].

150. AIA, *supra note* 149, *§ 5*, 125 Stat. at 297–99 (amending 35 U.S.C. § 273); *see also* 157 CONG. REC. S5430 (daily ed. Sept. 8, 2011) (statement of Sen. Kyl) (explaining the intent of the new provision).

151. AIA, *supra* note 149, § 14(c), 125 Stat. at 327–28 (codified at note following 35 U.S.C. § 102); *see also* H.R. REP. NO. 112-98, Part 1, at 51–52, 79 ("a software program that is novel and non-obvious as software would not be affected by this section even though the software is used for a tax purpose"); *reprinted in* 2011 U.S.C.C.A.N. 67, 82, 104; 157 CONG. REC. S5432 (statement of Sen. Kyl) ("patents can still be issued for software that helps taxpayers prepare their tax returns, but that provision is intended to be narrowly construed and is not intended to authorize patents for business methods or financial management software"); Enfish, LLC v. Microsoft Corp., 56 F. Supp. 3d 1167, 1172 (C.D. Cal. 2014) ("This section [of the AIA] implicitly affirms software as [patent-]eligible subject matter.").

152. AIA, *supra* note 149, § 18, 125 Stat. at 329–31 (codified at note following 35 U.S.C. § 321).

153. *See, e.g.*, 157 CONG. REC. S5431 (daily ed. Sept. 8, 2011) (statement of Sen. Kyl) ("technological inventions are excluded from the scope of the program. . .. an actual software invention is a technological invention, and is not subject to review under section 18"); *id.* at S5433 (statement of Sen. Durbin); *id.* at S5441 (statement of Sen. Leahy).

tion calling for an end to software patenting.[154] The result of all the legislative attention paid to software patenting in the run-up to the AIA was, ultimately, very little substantive change in the law.

D. Bilski *and* Alice*: The Boom Goes Bust*

The substantial changes to judicial doctrine on the subject of software patenting in this decade present a substantial contrast to the relative inaction of Congress. The Supreme Court has addressed the patent eligibility of software-related inventions in two cases since 2010, with consequences that are still being felt in the lower courts.

The Federal Circuit's *en banc* decision in *In re Bilski* undertook a searching review of the evolution of the law on the patent-eligibility of software inventions.[155] *Bilski* involved a patent covering the use of hedging techniques in commodities trading, not computer software.[156] Nevertheless, the court's analysis did much to undermine the key precedents upon which software had been determined to be broadly patent-eligible. The court of appeals drew from the Supreme Court's *Benson–Flook–Diehr* trilogy the principle that "[a] claimed process is surely patent-eligible under § 101 if: (1) it is tied to a particular machine or apparatus, or (2) it transforms a particular article into a different state or thing."[157] The court then confronted its own post-*Diehr* case law. Specifically disapproving the reasoning of *Alappat* and *State Street*, the court of appeals stated:

> [W]hile looking for "a useful, concrete and tangible result" may in many instances provide useful indications of whether a claim is drawn to a fundamental principle or a practical application of such a principle, *that inquiry is insufficient to determine whether a claim is patent-eligible under § 101*. . .. Therefore, we also conclude that the "useful, concrete and tangible result" inquiry is inadequate and reaffirm that the machine-or-transformation test outlined by the Supreme Court is the proper test to apply.[158]

---

154. *See* Quentin Palfrey, *Promoting Innovation and Competitive Markets through Quality Patents*, PETITION THE WHITE HOUSE ON THE ISSUES THAT MATTER TO YOU, (Sept. 23, 2011), https://petitions.whitehouse.gov/petition/direct-patent-office-cease-issuing-software-patents ("the new transitional post-grant review program will help the USPTO take a closer look at certain business method patents, including a number of software patents"), *archived at* [perma.cc/TJ4N-5DLV].

155. *In re* Bilski, 545 F.3d 943 (Fed. Cir. 2008), *aff'd sub nom.* Bilski v. Kappos, 561 U.S. 593 (2010).

156. *See Bilski* at 949 (quoting patent claim language).

157. *Id.* at 954; *see also id.* at 951–55 (surveying development of Supreme Court's views on patent eligibility).

158. *Id.* at 959–60 (emphasis added); *cf. supra* notes 137, 140 and accompanying text.

The Supreme Court affirmed.[159] Although the majority agreed that Bilski's invention was unpatentable, it ruled that the Federal Circuit's exclusive focus on whether the claimed process was "tied to a particular machine or apparatus" or "transforms a particular article" was too narrow.[160] Nevertheless, the majority agreed with the Federal Circuit in rejecting the suggestion of cases such as *State Street* that a "useful, concrete and tangible result" sufficed to render a computational process patent-eligible.[161]

Justice Stevens, writing for himself and three others, would have gone further still. Both English and American legal history, in the concurring opinion's view, supported a conclusion that "a series of steps for conducting business is not a 'process' under § 101."[162] Congress's creation of the "prior commercial use" defense in 1999, Justice Stevens argued, reflected its "surprise and perhaps even dismay" at the Federal Circuit's decision in *State Street*.[163] And Justice Breyer wrote separately to "highlight the substantial *agreement* among many Members of the Court" on several issues, including specifically the Court's unanimous rejection of the "useful, concrete, and tangible result" test from *State Street*.[164]

The potential implications of *Bilski* for software patenting became clear four years later with the Supreme Court's decision in *Alice Corp. Pty. Ltd. v. CLS Bank International*.[165] *Alice* involved "several patents that disclose schemes to manage certain forms of financial risk" that were "[a]ll. . . implemented using a computer; the system and media claims expressly recite a computer, and the parties have stipulated that the method claims require a computer as well."[166] Agreeing with the court of appeals that the patent's claims were directed to ineligible subject matter, the Supreme Court reasoned that its case law required the courts to "distinguish between patents that claim the building blocks of human ingenuity and those that integrate the building blocks into something more, thereby transforming them into a patent-eligible invention[.]"[167] Its precedents in *Benson* and *Flook*, the Court

159. 561 U.S. 593.

160. *See id.* at 603 ("Adopting the machine-or-transformation test as the sole test for what constitutes a 'process' (as opposed to just an important and useful clue) violates these statutory interpretation principles."), 604 ("The machine-or- transformation test is not the sole test for deciding whether an invention is a patent-eligible 'process.' ").

161. *Id.* at 612 ("nothing in today's opinion should be read as endorsing interpretations of § 101 that the Court of Appeals for the Federal Circuit has used in the past. *See, e.g.*, *State Street*, 149 F.3d, at 1373. . .."); *cf. supra* note 140 and accompanying text.

162. *Bilski*, 561 U.S. at 643 (Stevens, J., concurring); *see also id.* at 626–44.

163. *See id.* at 645. As noted previously, Congress enlarged this exception in the 2011 AIA to remove any suggestion that it was limited to patents over business methods. *See supra* note 150 and accompanying text.

164. *Bilski*, 561 U.S. at 657, 659–60 (Breyer, J., concurring).

165. Alice Corp. Pty. Ltd. v. CLS Bank International, 134 S. Ct. 2347 (2014).

166. *Id.* at 2352 (footnote omitted), 2353.

167. *Id.* at 1254 (internal quotations omitted) (citing Mayo Collaborative Servs. v. Prometheus Labs., Inc., 132 S. Ct. 1289, 1303 (2012)).

continued, set forth "the longstanding rule that an idea of itself is not patentable."[168] And *Bilski* established that "basic concept[s]" and "fundamental economic practice[s]" constituted just such "a patent-ineligible abstract idea, just like the algorithms at issue in *Benson* and *Flook*."[169] In *Alice*, the Court identified another unpatentable "abstract idea" underlying the claims made in the challenged patent: "the concept of intermediated settlement, *i.e.*, the use of a third party to mitigate settlement risk."[170]

Unlike the business method at issue in *Bilski*, the patent claims in *Alice* all required the use of specialized computer software. Reviewing its *Benson–Flook–Diehr* trilogy, however, the Court in *Alice* reasoned that "[t]he introduction of a computer into the claims does not alter the analysis[.]"[171] To the contrary, the Court continued, "if a patent's recitation of a computer amounts to a mere instruction to implement an abstract idea on a computer, that addition cannot impart patent eligibility."[172] "[E]ach step" of the software-implemented process claimed in the patent "does no more than require a generic computer to perform generic computer functions."[173] This differed, in the Court's view, from other types of computer-related inventions that might meet the threshold requirement of patent eligibility:

> Viewed as a whole, petitioner's method claims simply recite the concept of intermediated settlement as performed by a generic computer. *The method claims do not, for example, purport to improve the functioning of the computer itself. Nor do they effect an improvement in any other technology or technical field.* Instead, the claims at issue amount to nothing significantly more than an instruction to apply the abstract idea of intermediated settlement using some unspecified, generic computer. Under our precedents, that is not enough to transform an abstract idea into a patent-eligible invention.[174]

In the wake of the Federal Circuit's decisions in *Alappat* and *State Street*, the USPTO began to grant many more patents covering software inventions.[175] Following *Alice*, however, the reverse appears to be occurring. In 2015, the first full year following *Alice* for which data are available, the number of U.S. software patents issued fell by 7.12%—the first year-over-

168. *Id.* at 2355 (internal quotations and citation omitted).
169. *Id.* at 2356 (internal quotations and citation omitted).
170. *Id.*
171. *Id.* at 2357; *see also id.* at 2357–58.
172. *Id.* at 2358 (internal quotations and citations omitted).
173. *Id.* at 2359.
174. *Id.* at 2359–60 (internal quotations and citations omitted, emphasis added). Justice Sotomayor, joined by two others, would have decided the case based upon the general unpatentability of claims for methods of transacting business without addressing the computer software issue. *See id.* at 2360–61 (Sotomayor, J., concurring).
175. *See* USPTO PATENT COUNTS, *supra* note 141 and accompanying text.

year decline in the number of such patents granted since 2007.[176] In one subject-matter field (category 705, *Data Processing: Financial, Business Practice, Management, or Cost/Price Determination*), the number of issued patents fell by more than half in a single year.[177] A recent study found that *Alice* had also led to the first decline in five years in the number of new patent infringement lawsuits filed, with a 13% decrease in 2014.[178]

Lower courts, too, have been reacting to the Court's ruling in *Alice*. Although some computer-related patents withstood judicial scrutiny after *Alice*,[179] a vastly greater number were struck down as patent-ineligible subject matter.[180] With manifest understatement, the Federal Circuit noted recently

176. *See id.*

177. *See id.*

178. *See* PricewaterhouseCoopers, 2015 Patent Litigation Study: A Change in Patentee Fortunes 3 & fig. 1, https://www.pwc.com/us/en/forensic-services/publications/assets/2015-pwc-patent-litigation-study.pdf, *archived at* [perma.cc/K2LM-EXRM]. New case filings rebounded somewhat in 2015, although did they not return to pre-*Alice* levels; and case filings in 2016 fell again, to the lowest level since 2011. *See* Brian Howard, Announcing the Patent Litigation Year in Review 2015, https://lexmachina.com/14318/ (Mar. 16, 2016), *archived at* [perma.cc/4L87-CW5M]; Brian Howard, *2016 Fourth Quarter Litigation Update*, https://lexmachina.com/q4-litigation-update/ (Jan. 12, 2017), *archived at* [perma.cc/4FJC-CM2E].

179. *See, e.g.*, DDR Holdings, LLC v. Hotels.com, LP, 773 F.3d 1245, 1256–59 (Fed. Cir. 2014); Motio, Inc. v. BSP Software LLC, 154 F. Supp. 3d 434, 437–41 (E.D. Tex. 2016); California Inst. of Tech. v. Hughes Comm'cns Inc., 59 F. Supp. 3d 974, 993–1000 (C.D. Cal. 2014); Fairfield Indus., Inc. v. Wireless Seismic, Inc., No. 4:14-CV-2972, 2014 WL 7342525, **5–7 (S.D. Tex. Dec. 23, 2014).

180. *See, e.g.*, Intellectual Ventures I LLC v. Capital One Bank (USA), N.A., 792 F.3d 1363, 1367–71 (Fed. Cir. 2015); Internet Patents Corp. v. Active Network, Inc., 790 F.3d 1343, 1347–49 (Fed. Cir. 2015); OIP Technologies, Inc. v. Amazon.com, Inc., 788 F.3d 1359, 1362–64 (Fed. Cir. 2015); Content Extraction & Transmission LLC v. Wells Fargo Bank, N.A., 776 F.3d 1343, 1347–49 (Fed. Cir. 2014); Ultramercial, Inc. v. Hulu, LLC, 772 F.3d 709, 713–17 (Fed. Cir. 2014); buySAFE, Inc. v. Google, Inc., 765 F.3d 1350 (Fed. Cir. 2014); Digitech Image Techs., LLC v. Electronics for Imaging, Inc., 758 F.3d 1344 (Fed. Cir. 2014); Planet Bingo, LLC v. VGKS LLC, 576 F. App'x 1005 (Fed. Cir. 2014); OpenTV, Inc. v. Apple Inc., No. 5:15-cv-02008-EJD, 2016 WL 344845, **4–9 (N.D. Cal. Jan. 28, 2016); American Needle, Inc. v. Café Press Inc., No. 15-cv-3968, 2016 WL 232438, **2–3 (N.D. Ill. Jan. 19, 2016); Motivation Innovations, LLC v. Petsmart, Inc., 156 F. Supp. 3d 558, 568–69 (D. Del. 2016); Open Text S.A. v. Box, Inc., 78 F. Supp. 3d 1043, 1046–50 (N.D. Cal. 2015); Vehicle Intelligence & Safety LLC v. Mercedes-Benz USA, LLC, 78 F. Supp. 3d 884, 888–94 (N.D. Ill. 2015); Bascom Research, LLC v. LinkedIn, Inc., 77 F. Supp. 3d 940, 947–54 (N.D. Cal. 2015); Collarity, Inc. v. Google Inc., No. 11-1103-MPT, 2015 WL 7597413, **4–11 (D. Del. Nov. 25, 2015); Listingbook, LLC v. Market Leader, Inc., No. 1:13-cv-583, 144 F. Supp. 3d 777, 786–92 (M.D.N.C. 2015); Telebuyer, LLC v. Amazon.com, Inc., No. No. 2:13-cv-1677-BJR, 2015 WL 4493045, **9–12 (W.D. Wash. July 23, 2015); Jericho Sys. Corp. v. Axiomatics, Inc., No. 3:14-CV-2281-K, 2015 WL 2165931, **3–7 (N.D. Tex. May 7, 2015); Advanced Auctions LLC v. eBay Inc., No. 13CV1612 BEN (JLB), 2015 WL 1415265, **3–6 (S.D. Cal. Mar. 27, 2015); Tenon & Groove, LLC v. Plusgrade S.E.C., No. 12-1118-GMS-SRF, 2015 WL 82531, **7–8 (D. Del. Jan. 6, 2015), *magistrate's recommendations adopted*, 2015 WL 1133213 (D. Del. Mar. 11, 2015); Hewlett Packard Co. v. ServiceNow, Inc., No. 14-cv-00570-BLF, 2015 WL 1133244, **5–11 (N.D. Cal. Mar. 10, 2015); IpLearn, LLC v. K12 Inc., 76 F. Supp. 3d 525, 531–35 (D. Del. 2014); KomBea Corp. v. Noguar L.C., 73 F. Supp. 3d 1348, 1351–57 (D. Utah 2014); Cogent Medicine, Inc. v. Elsevier Inc., 70 F. Supp. 3d

that *Alice* gave "renewed vigor" to "§ 101 defense[s] previously lacking in merit" under prior law.[181]

In mid-2016, with software patents in free fall, the Federal Circuit signaled a desire to limit some of the impact of the *Alice* analysis. Judge Hughes's opinion for the panel in *Enfish, LLC v. Microsoft Corp.*[182]cautioned that "[w]e do not read *Alice* to broadly hold that all improvements in computer-related technology are inherently abstract[.]"[183] Although the Supreme Court had suggested that inventions that "purport to improve the functioning of the computer itself" would be more likely patent-eligible, the *Enfish* court countered that "[s]oftware can make non-abstract improvements to computer technology just as hardware improvements can[.]"[184] Just five days later, however, in another opinion by Judge Hughes, the Court of Appeals struck down another software patent that merely claimed the use of "conventional or generic technology in a nascent but well-known environment."[185] The division between these two virtually simultaneous panel opinions concerning what the *Alice* analysis actually requires has continued to color subsequent cases, with multiple decisions both upholding[186] and invalidating[187] particular software patents.

1058, 1063–66 (N.D. Cal. 2014); MyMedicalRecords, Inc. v. Walgreen Co., Nos. 2:13-CV-00631 ODW (SHx) *et al.*, 2014 WL 7339201, **2–5 (C.D. Cal. Dec. 23, 2014); Open Text S.A. v. Alfresco Software Ltd., No. 13-cv-04843-JD, 2014 WL 4684429 (N.D. Cal. Sept. 19, 2014); Every Penny Counts, Inc. v. Wells Fargo Bank, N.A., No. 8:11-cv-2826-T-23TBM, 2014 WL 4540319 (M.D. Fla. Sept. 11, 2014).

181. Mortgage Grader, Inc. v. First Choice Loan Servs. Inc., 811 F.3d 1314, 1322 (Fed. Cir. 2016); *see also* Jasper L. Tran, *Two Years After* Alice v. CLS Bank, 98 J. PAT. & TRADEMARK OFF. SOC'Y 354 (2016) (summarizing post-*Alice* patent invalidation rates of the Federal Circuit, district courts, and the USPTO's Patent Trial and Appeal Board).

182. Enfish, LLC v. Microsoft Corp, 822 F.3d 1327 (Fed. Cir. 2016).

183. *Id.* at 1335.

184. *Id.*; *cf. Alice Corp.*, 134 S. Ct. at 2347 and accompanying text.

185. *In re* TLI Commc'ns LLC Patent Litig., 823 F.3d 607, 612 (Fed. Cir. 2016).

186. *See, e.g.*, Visual Memory LLC v. NVIDIA Corp., 867 F.3d 1253, 1257–62 (Fed. Cir. 2017); Amdocs (Israel) Ltd. v. Openet Telecom, Inc., 841 F.3d 1288, 1300–01 (Fed. Cir. 2016) (upholding patents over system for billing of network users based on bandwidth they consumed where system "entails an unconventional technological solution. . . to a technological problem" and "operate[s] in an unconventional manner to achieve an improvement in computer functionality"); McRO, Inc. v. Bandai Namco Games Am. Inc., 837 F.3d 1299, 1316 (Fed. Cir. 2016) (upholding patent on method of synchronizing lip and facial movements of video-game characters where claimed invention was "limited to a specific [animation] process. . . and does not preempt approaches that use rules of a different structure or different techniques"); BASCOM Global Internet Servs., Inc. v. AT&T Mobility LLC, 827 F.3d 1341, 1350 (Fed. Cir. 2016) (upholding patent that "recite[d] a specific, discrete implementation of the abstract idea of filtering content"); TNS Media Research LLC v. TIVO Research and Analytics, Inc., 223 F. Supp. 3d 168, 179 (S.D.N.Y. 2016) (specifically noting post-*Enfish* trend toward slightly broader acceptance of software patent validity).

187. *See, e.g.*, Easyweb Innovations. LLC v. Twitter, Inc., 689 F. App'x 969 (Fed. Cir. 2017) (invalidating five patents); Recognicorp, LLC v. Nintendo Co., Ltd., 855 F.3d 1322, 1326–28 (Fed. Cir. 2017); Intellectual Ventures I LLC v. Capital One Fin. Corp., 850 F.3d 1332, 1340–42 (Fed. Cir. 2017); Intellectual Ventures I LLC v. Erie Indem. Co., 850 F.3d

The *Alice* decision itself has come in for some criticism, with one court comparing *Alice*'s methodology to Justice Stewart's famous "I know it when I see it" standard for identifying obscenity.[188] And another criticized *Alice* for "fail[ing] to answer this: when, if ever, do computer patents survive § 101?"[189] Many open issues remain, such as how to identify an "abstract idea" ineligible for patent protection, or an "inventive concept" that may rescue the "abstract idea" and render it patent-eligible.[190] Although its ultimate importance may not yet be known, *Alice* has already had a demonstrable impact on both administrative and judicial practice in software patenting.

1315, 1327–32 (Fed. Cir. 2017) (invalidating two patents); Smartflash LLC v. Apple Inc., 680 F. App'x 977 (Fed. Cir. 2017) (invalidating three patents); FairWarning IP, LLC v. Iatric Sys., Inc., 839 F.3d 1089, 1094 (Fed. Cir. 2016) (invalidating patent on method for detecting misuse of confidential health data where patent's "claims merely implement an old practice in a new environment"); Intellectual Ventures I LLC v. Symantec Corp., 838 F3d 1307, 1314–16 (Fed. Cir. 2016) (invalidating e-mail filtering patent that added no inventive concept to the abstract idea of searching digital information to see whether it contained specified content); Affinity Labs of Tex., LLC v. DirecTV, LLC, 838 F.3d 1253, 1262 (Fed. Cir. 2016) (invalidating patent that merely "recites the use of generic features of cellular telephones" to implement "the general concept of out-of-region delivery of broadcast content"); LendingTree, LLC v. Zillow, Inc., 656 F. App'x 991, 996 (Fed. Cir. 2016) (invalidating patent that merely claimed implementation of a "fundamental economic practice" performed electronically); Front Row Techs., LLC v. NBA Media Ventures, LLC, 204 F. Supp. 3d 1190, 1268, 1279 (D.N.M. 2016) (invalidating patent claims "directed to two abstract ideas: (i) sending video of an event to handheld devices over wireless networks; and (ii) authorizing handheld devices to receive streaming video based on a user's location" where electronic devices that were alleged to supply the necessary "inventive concept" "are generic, and the claims use them in conventional ways"); Sound View Innovations, LLC v. Facebook, Inc., 204 F. Supp. 3d 655, 661–64 (D. Del. 2016) (invalidating patent directed to implementing abstract idea of "targeted advertising" using "generic computer components").

188. Eclipse IP LLC v. McKinley Equipment Corp., No. SACV 14-154-GW(AJWx), 2014 WL 4407592, at *3 (C.D. Cal. Sept. 4, 2014) (citing Jacobellis v. Ohio, 378 U.S. 184, 197 (1964) (Stewart, J. concurring)); *See also id.* ("Thus, so far, the two-part [*Alice*] test for identifying an abstract idea appears to be of limited utility, while comparisons to previously adjudicated patents—or more precisely, to past cases' characterizations of those patents—have done the heavy lifting.") (citation and footnote omitted); *Amdocs*, *supra* 186, at 817–18 (expressing similar views).

189. California Inst. of Tech. v. Hughes Comm'cns Inc., 59 F. Supp. 3d 974, 984 (C.D. Cal. 2014).

190. *See, e.g.*, Versata Dev. Group, Inc. v. SAP Am., Inc., 793 F.3d 1306, 1331 (Fed. Cir. 2015) (recognizing the "problem inherent in the search for a definition of an 'abstract idea' that is not itself abstract"); John Clizer, Note, *Exploring the Abstract: Patent Eligibility Post* Alice Corp. v. CLS Bank, 80 Mo. L. Rev. 537, 552–56 (2015). The Federal Circuit has disavowed any attempt to supply a governing definition. *See Amdocs*, *supra* 186, at 1294 (declaring it "difficult to fashion a workable definition [of 'abstract idea'] to be applied to as-yet-unknown cases with as-yet-unknown inventions"). The court instead invites litigants to reason by analogy to earlier cases in which the patent-eligibility of similar inventions was resolved. *See id.*

III. Institutional Responses to Legal Uncertainty

Both copyright and patent law have changed significantly in the last two decades, and are continuing to evolve in ways that may be highly consequential for the development and legal protection of computer software. The case developments sketched out above describe two opposing pendulum swings: in copyright, from relatively strong protection under early decisions such as *Whelan*, to relatively weak protection under *Altai*, to the apparent revival of the *Whelan* analysis under *Oracle*; and in patent, from relatively strong protection under *Alappat* and *State Street*, to the more recent decisions in *Bilski* and (especially) *Alice*, which appear to be causing a substantial pullback in the extent of patent protection for software.

All this has happened, however, with virtually no legislative action. Indeed, turmoil in the pertinent legal doctrines reflects nothing so much as the lack of meaningful guidance in the language of the governing statutes. Time and again, Congress has spoken either indirectly or not at all on the issues that have divided the courts.[191] Ameliorating the problems caused by legislative inaction would seem to represent a problem particularly suited to legislative response. Congress and the courts, however, both have multiple sources of authority from which they might draw to help clarify the reach of intellectual property protection for software. A sufficiently rigorous judicial reexamination of the copyright and patent law principles that recent precedents have so thoroughly unsettled might even obviate the need for remedial legislation; and if legislation is ultimately necessary, Congress has several paths from which to choose. It is appropriate, accordingly, to survey some of the possible institutional responses to the problems described above.

A. *Statutory Revision or Administrative Delegation*

Perhaps the most direct alternative would be for Congress to simply answer the questions raised by recent case law concerning the availability and scope of patent and copyright protection for software. At present, however, there is little evidence that Congress possesses the expertise necessary to legislate wisely in this area. Congress counts few technological experts among its members.[192] Nor has Congress undertaken notable steps to develop policy expertise in this area. Of the twenty hearings on copyright reform that the House Judiciary Committee held between 2013 and 2015, for

191. *See, e.g.*, *supra* notes 62–64 (noting that Congress expressed its desire to extend copyright protection to software only in the legislative history, not the text, of the 1980 amendments to the Copyright Act), 149–154 (noting that the 2011 AIA statute only obliquely considered the possibility of patenting software) and accompanying text.

192. *See* Congressional Research Service, *Membership of the 114th Congress: A Profile*, https://fas.org/sgp/crs/misc/R43869.pdf, at 4 (2016) (noting that the present Congress includes "five software company executives in the House and two in the Senate" as well as "one physicist, one microbiologist, one chemist, and eight engineers (all in the House, with the exception of one Senator who is an engineer)").

example, not one devoted itself to the question of software copyright.[193] The very paucity of legislation dealing specifically with either copyright or patent protection for software shows that Congress itself may possess limited institutional competence to legislate on the question without external guidance.[194]

As an alternative to legislating directly, Congress might choose—as it frequently has in other areas involving complex technical subject matter—to delegate power to an outside agency to devise and administer appropriate legal rules.[195] Indeed, there is some precedent for such a delegation within the copyright statute itself; in 1998, Congress delegated to the Librarian of Congress the authority to create exceptions to liability under the new anticircumvention provisions of the Digital Millennium Copyright Act.[196] A prominent rationale for delegating substantive rulemaking authority to administrative agencies—to wit, that specialist agencies' technical expertise will lead to rules better adapted to the specific characteristics of the activities they regulate—would appear to be fully applicable in the complex field of computer technology.[197] Proposals to increase the involvement of specialized administrative bodies in intellectual property law recur periodically in the academic literature.[198]

---

193. *See* U.S. Copyright Review, *supra* note 15.

194. Congress's historical practice of outsourcing the drafting of intellectual property legislation to affected industry groups only reinforces doubts about whether statutory revision is a task Congress is equipped to undertake on its own—while also clouding the question whether legislation resulting from such a process would necessarily advance the broader public interest. *See* Sepehr Shahshahani, *The Nirvana Fallacy in Fair Use Reform*, 16 MINN. J.L. SCI. & TECH. 273, 291–95 (2015); Stuart Minor Benjamin & Arti K. Rai, *Fixing Innovation Policy: A Structural Perspective*, 77 GEO. WASH. U. L. REV. 1, 42 (2008) ("A major problem for legislative battles involving innovation is that future industries and innovators do not have a seat at the lobbying table, as they either do not exist or exist only in nascent form."). My emphasis herein on the question of legislative competence is not meant to signal unconcern with the effect of statutory revision on the public interest; indeed, the proposal articulated below for a legislative commission modeled on the original CONTU would provide the opportunity for public-interest advocates to have more direct influence on legislative outcomes than they presently enjoy.

195. *See, e.g.*, Anandashankar Mazumdar, *Disagreement on Autonomy, Small Claims But Largely Agree on IT Improvement*, 93 PAT. COPYRIGHT & TRADEMARK J. (BNA) 3009, 3010 (2017) (summarizing proposal by publishers' trade association to give Copyright Office substantive rulemaking authority on grounds that agency may react more swiftly than Congress to technological change).

196. *See* 17 U.S.C. § 1201(a)(1)(B)–(E) (2012).

197. *See, e.g.*, Chevron USA Inc. v. Natural Resources Def. Council, Inc., 467 U.S. 837, 865–66 (1984) (reasoning that agencies' superior substantive expertise justifies judicial deference to their construction of complex regulatory terms); Benjamin & Rai, *supra*, note 194, at 33–34 (summarizing factors tending to give administrative agencies greater substantive expertise in technical fields than Congress or the courts are able to develop).

198. *See, e.g.*, Jason Mazzone, *Administering Fair Use*, 51 WM. & MARY L. REV. 395, 413 (2009) (arguing that administrative agencies play a "surprising[ly]" "limited" role in intellectual property law, which "is well-suited to agency governance"); Michael W. Carroll, *Fixing Fair Use*, 85 N.C. L. REV. 1087, 1123–27 (2007) (arguing for creation of administrative

There are reasons to doubt the viability of the administrative alternative here, however. First, and perhaps most obviously, the existence of multiple proposals to expand the authority of administrative agencies in intellectual property serves to highlight the novelty of this approach. Administrative governance remains largely untried in regulating intellectual property,[199] and Congress may be less inclined to depart from past practice here than if it were contemplating regulation of a previously unregulated field. Second, Congress's duty to confine agencies' delegated powers within "intelligible principles" fixed by the legislature[200] means that even the choice of administrative regulation cannot entirely free Congress from the duty to grapple with some of the challenging questions surrounding intellectual property protection for software that have arisen in the absence of legislative guidance to date.

Recent history provides a third reason to question the efficacy of the administrative alternative.[201] In 2006 and 2010, the Librarian of Congress used the authority delegated to it under the 1998 DMCA statute to promulgate a DMCA exemption permitting purchasers of handheld wireless devices (such as cellular telephones) to "unlock" the devices so they could be used on other providers' networks.[202] In 2012, however, the Librarian effectively refused to renew the exemption, based largely upon arguments it had considered and persuasively rejected in the earlier rulemakings.[203] Consumers frustrated by the agency's abrupt and poorly explained *volte-face* complained to the Obama Administration, which agreed that "consumers should be able to unlock their cell phones without risking criminal or other penalties."[204] Con-

---

tribunal within U.S. Copyright Office to adjudicate issues of fair use); Joseph P. Liu, *Regulatory Copyright*, 83 N.C. L. REV. 87, 148–60 (2004) (recommending legislative delegation of both rulemaking and adjudicative functions to the Copyright Office or another administrative agency); Dan L. Burk & Julie E. Cohen, *Fair Use Infrastructure for Rights Management Systems*, 15 HARV. J.L. & TECH. 41, 66–67 (2001) (suggesting that Library of Congress serve as escrow agent to hold decryption keys that users would need to engage in fair uses of encrypted digital works). For a less sanguine view, see Benjamin & Rai, *supra* note 194, at 47–51 (criticizing agencies as protecting market incumbents at the expense of innovation).

199. *See, e.g.*, Merck & Co., Inc. v. Kessler, 80 F.3d 1543, 1549–50 (Fed. Cir. 1996) (noting that the USPTO has no substantive rulemaking power in patent law).

200. Mistretta v. United States, 488 U.S. 361, 372 (1989).

201. A further retelling of these events is available in Timothy K. Armstrong, *The DMCA and the Cell Phone Unlocking Debate* (Cincinnati Law Research Paper Series, Working Paper No. 14-04), https://ssrn.com/abstract=2401013.

202. *See* Exemption to Prohibition on Circumvention of Copyright Protection Systems for Access Control Technologies, 75 Fed. Reg. 43,825-01, 43,830–31 (July 27, 2010); Exemption to Prohibition on Circumvention of Copyright Protection Systems for Access Control Technologies, 71 Fed. Reg. 68,472, 68,476 (Nov. 27, 2006).

203. *See* Exemption to Prohibition on Circumvention of Copyright Protection Systems for Access Control Technologies, 75 Fed. Reg. 65,260, 65,264–66 (Oct. 26, 2012).

204. *Make Unlocking Cell Phones Legal*, THE WHITE HOUSE (Jan. 24, 2013), https://petitions.whitehouse.gov/petition/make-unlocking-cell-phones-legal/1g9KhZG7 [http://web.archive.org/web/20150128211840/https://petitions.whitehouse.gov/petition/make-unlocking-

gress agreed and amended the DMCA to make the cell phone unlocking exemption permanent.[205] The result of the delegation of rulemaking authority, in this instance at least, was that the agency bungled an issue involving advanced technologies so badly that Congress had to step in anyway to fix the problem. To be sure, this example hardly demonstrates that administrative regulation of technology is doomed to failure. It does, however, provide a cautionary illustration for those skeptical that administrative expertise will necessarily yield results best attuned to the needs of technology users.

B. *Bringing Outside Expertise to Bear*

If Congress itself lacks the expertise either to legislate directly or to supply intelligible standards to guide administrative policymaking, then the solution must at some point involve looking outside Congress for guidance. Several models for developing such policy guidance exist. For example, in areas ordinarily governed by state law, the Uniform Law Commission supplies model legislation intended to have general applicability, such as the ubiquitous Uniform Commercial Code (UCC), the generally successful Uniform Trade Secrets Act (UTSA), and the largely rejected Uniform Computer Information Transactions Act (UCITA).[206] Alternatively, the series of *Restatements of the Law* published by the American Law Institute (ALI) provide guidance and commentary on widely accepted principles of both state and federal law; indeed, the drafting of the ALI's first-ever *Restatement of Copyright Law* is underway at the time of this writing.[207]

Neither of these alternatives quite fits the current need to clarify both copyright and patent law as they apply to computer software, however. Copyright and patent law are overwhelmingly governed by federal, not state, law, and are for that reason ill-suited to the model-act drafting process.[208]

---

cell-phones-legal/1g9KhZG7]. A somewhat chastened Librarian of Congress responded by declaring that "[t]he rulemaking is a technical, legal proceeding" that "was not intended to be a substitute for deliberations of broader public policy." Press Release, LIBRARY OF CONG., *Statement from the Library of Congress Regarding White House Statement Today in Response to a Petition on Section 1201 Rulemaking* (Mar. 4, 2013), http://www.loc.gov/today/pr/2013/13-041.html [perma.cc/7TS4-F6JN].

205. *See* Unlocking Consumer Choice and Wireless Competition Act, Pub. L. No. 113-144, 128 Stat. 1751 (2014).

206. *See generally, Acts,* UNIF. LAW COMM'N, http://www.uniformlaws.org/Acts.aspx (last visited Mar. 1, 2017).

207. *See Copyright,* AM. LAW INSTI., https://www.ali.org/projects/show/copyright/ (last visited Mar. 1, 2017). For a generally favorable view of the possible benefit of a Restatement in the field of copyright law, see Ann Bartow, *A Restatement of Copyright Law as More Independent and Stable Treatise*, 79 BROOK. L. REV. 457 (2014). For a more skeptical reaction, see Michael Risch, *Weighing Treatises v. Restatements—Copyright Edition*, WRITTEN DESCRIPTION (Jan. 27, 2015), http://writtendescription.blogspot.com/2015/01/weighing-treatises-v-restatements.html [perma.cc/K577-W6TR].

208. *See* 17 U.S.C. § 301(a) (2012) (federal law now preempts "all legal or equitable rights that are equivalent to any of the exclusive rights within the general scope of copy-

ALI *Restatements*, meanwhile, "aim at clear formulations. . . and reflect the law as it presently stands or might appropriately be stated by a court."[209] One may question whether a "clear formulation" of the law of software copyright "as it presently stands" is even possible in view of the fluid nature of current doctrine.[210] And, of course, the ALI has yet to take up the subject of patent law, rendering any guidance it might provide on software copyright law necessarily incomplete.

International bodies such as WIPO or the WTO might provide alternative fora for debating and resolving software issues. This alternative, however, presents difficulties of its own. International views are, if anything, even less settled than United States law on the desirability and nature of intellectual property protection for software.[211] Congress and the present Administration may balk at even the appearance of following international entities' leadership. And, of course, the history of U.S. reliance on international fora to resolve contentious intellectual property issues is somewhat checkered; the story of how the 1996 WIPO process was used effectively to short-circuit Congressional opposition to the legislation that became the 1998 DMCA remains salient today.[212]

On over 100 occasions in the last thirty years, Congress has created temporary advisory commissions tasked with developing policy expertise in a wide variety of subject areas and reporting back to the legislature.[213] Such commissions are well-understood and accepted parts of the legislative pro-

---

right. . .. under the common law or statutes of any State"); 28 U.S.C. § 1338(a) (2012) (conferring exclusive federal court jurisdiction over cases "arising under any Act of Congress relating to patents. . . or copyrights"). Small pockets of state regulatory authority remain; for example, state (rather than federal) law governs the existence and scope of copyright rights in sound recordings fixed in a tangible medium before February 15, 1972. *See* 17 U.S.C. § 301(c) (2012); Flo & Eddie, Inc. v. Sirius XM Radio, Inc., 821 F.3d 265, 269–71 (2d Cir. 2016) (certifying question of state copyright law); Flo & Eddie, Inc. v. Sirius XM Radio, Inc., 70 N.E.3d 936 (N.Y. 2016) (answering certified question). In Gunn v. Minton, 133 S. Ct. 1059 (2013), the Supreme Court approved state-court jurisdiction over legal malpractice claims arising from allegedly mishandled patent litigation—a result that, if applied broadly, may permit state courts to adjudicate other ancillary questions connected with patent cases.

209. *Publication Frequently Asked Questions*, AM. LAW. INST., https://www.ali.org/publications/frequently-asked-questions/ (last visited Mar. 1, 2017) [hereinafter ALI FAQ].

210. For a recognition of this general difficulty that nevertheless concludes that the ALI is capable of resolving such challenges, see Lance Liebman, *Law Reform Agenda as ALI Approaches Its Centennial*, 79 BROOK. L. REV. 821, 822–23 (2014). The ALI also offers alternative publications styled *Principles of the Law* that it believes more suitable than a *Restatement* "when an area is so new that there is little established law." *ALI FAQ*, *supra* note 209. This alternative may be better suited to software copyright law in its present state.

211. *See, e.g.*, *supra* note 143 and *infra* note 233 and accompanying text.

212. *See generally* BLAYNE HAGGART, COPYFIGHT: THE GLOBAL POLITICS OF DIGITAL COPYRIGHT REFORM 109–32 (2014); JESSICA LITMAN, DIGITAL COPYRIGHT 122–45 (2001).

213. CONG. RESEARCH SERV., R40076, CONGRESSIONAL COMMISSIONS: OVERVIEW, STRUCTURE, AND LEGISLATIVE CONSIDERATIONS 1 (2017) [hereinafter CONGRESSIONAL COMMISSIONS. Policy commissions made up "the vast majority" of the 107 congressional commissions established from 1989 to 2016. *See id*. at 4, 5.

cess and are subject to statutory obligations requiring, among other things, balance of competing viewpoints and the holding of public meetings.[214] Legislative reliance on the diverse viewpoints of outside experts can improve consensus-based policymaking and reduce the impact of partisan polarization.[215] The original CONTU Commission was one such body, of course. One need not accept the conclusions of the original CONTU report in every particular to celebrate the principle that policymaking should be informed by substantive expertise.[216]

Accordingly, Congress should consider the formation of a new legislative commission—staffed, as the original CONTU was not, with technologists and software experts, not only attorneys—to provide policy guidance.[217] A new CONTU could consider the issues from a broader perspective than any court, because courts ordinarily must accept the litigants' framing of the issues for review. The requirement of viewpoint diversity might better ensure that the public interest is taken into account in the commission's deliberations than in a legislative process from which the public is generally excluded.[218] And a legislative commission could reassess CONTU's perceptions of the interplay between copyright, patent, trade secrecy, and other possible forms of intellectual property protection for software in light of more recent trends in the law and the software industry.[219]

IV. An Agenda for a New CONTU

A new CONTU commission should address, at a minimum, the following issues:

---

214. *See* 5 U.S.C. App. §§ 5(b), 10 (2012); *see also* Colton C. Campbell, Discharging Congress: Government by Commission 3 (2002) (explaining these and other provisions of the Federal Advisory Committee Act that aim to "maximize the public and broadly representative nature of the panels").

215. *See* Congressional Commissions, *supra* note 213, at 6–8. Of course, such commissions are made up of fallible mortals who may be ignorant or wasteful, and who are in any event not directly accountable to the voters for their conduct (although Congress, of course, remains accountable for any actions it takes or fails to take based on an expert body's recommendation). *Id.* at 9–10.

216. *See* Liu, *supra* note 198, at 161 (citing CONTU as "at least one successful example of [the] approach" of "an impartial expert body or commission" convened to advise Congress in areas where it lacks substantive expertise).

217. *See* Samuelson, *supra* note 29, at 699 (noting that CONTU included "no computer scientists, no software or hardware industry representatives, nor any users of complex software systems").

218. *See supra* note 194 and accompanying text.

219. *See, e.g.* Pamela Samuelson, *Preliminary Thoughts on Copyright Reform*, 2007 Utah L. Rev. 551, 553–56 (arguing that the 1976 Act and post-CONTU legislation were drafted before software was well understood by policymakers, and that improvement in our understanding over intervening decades now presents an opportune moment to revise the statute).

A. *The Existence and Shape of Legal Protection*

1. Incentives and Copying

Naturally, parties who derive financial gains from the incumbent legal regime may be expected to argue that their benefits should be continued or even enlarged.[220] Congress should not, however, close its eyes to evidence that legal protection may be unnecessary in at least some situations to promote the development of computer software. CONTU itself viewed the issue as at least open to debate, and subsequent events suggest that Congress may be wise to revisit the question.[221]

Two post-CONTU developments illustrate why rethinking the extension of intellectual property protections to software works may be appropriate.

First, one assumption underlying the extension of intellectual property rules to computer software is that software development is a commercial endeavor. Giving software developers protections against copying is a way of preventing loss of the revenues they would otherwise derive from the sale of copies. At the time of CONTU, the assumption may have been generally valid; although even then, contrary examples existed. Since that time, however, an entire sector of the software industry has arisen that expressly embraces free copying and reuse. This so-called "open-source" or "free software" movement[222] has produced complex, powerful works that rival or even exceed commercial products in quality and market penetration; indeed, in enterprise sectors such as supercomputing, web servers, cloud computing, mobile computing, and embedded systems, use of open-source products vastly predominates over commercially produced software.[223]

To be sure, the open-source model has been more successful in some areas than in others, and there may well be some types of works (including some types of software) for which it will never supply sufficient encourage-

220. *See, e.g.*, Jessica Litman, *Real Copyright Reform*, 96 IOWA L. REV. 1, 26 (2010) ("The gist of many of the proposals currently making the rounds is to shore up weaknesses in the positions of established distributors and to strengthen their claims against makers and other intermediaries as well as readers, listeners, and viewers.").

221. *See, e.g.*, Peter S. Menell, *The Challenges of Reforming Intellectual Property Protection for Computer Software*, 94 COLUM. L. REV. 2644, 2647 (2007) (noting that "[m]any ground-breaking areas of computer software research have been supported generously" through "non-market sources of revenue" such as "[g]overnment and private subsidies of research," all of which bear on the "the need for intellectual property protection" for such works).

222. *See, e.g.*, Jacobsen v. Katzer, 535 F.3d 1373, 1378 (Fed. Cir. 2008) ("Open source licensing has become a widely used method of creative collaboration that serves to advance the arts and sciences in a manner and at a pace that few could have imagined just a few decades ago."); Wallace v. International Bus. Mach. Corp., 467 F.3d 1104, 1105–06 (7th Cir. 2006) (describing licensing arrangements allowing free copying, modification, and distribution of Linux operating system and other open-source projects).

223. *See, e.g.*, Glyn Moody, *2015: Open Source Has Won, But it Isn't Finished*, COMPUTERWORLD UK (Jan. 1, 2015), http://www.computerworlduk.com/blogs/open-enterprise/open-source-has-won-3592314, [http://perma.cc/6BRB-M8HN].

ment for creative development. The rise of the open-source alternative, however, disproves any assumption that strong intellectual property protections are the *sine qua non* of software development. To develop a more robust understanding (grounded in empirical data) of when intellectual property rules are necessary in the context of software, and when they are not, should rank high on the list of objectives for any new CONTU commission.[224]

Second, the CONTU report presumes that software is mass-distributed, in the form of copies, to purchasers who wish to use the software. This was, after all, CONTU's stated basis for rejecting the alternative of trade secrecy protection.[225] The assumption may have been tolerably accurate as a description of the software industry for many years. More recently, however, the rise of cloud computing and the "software as a service" model has made powerful software available to users who never receive a copy of the program, but merely interact with it online.[226] Law firms, to take only one example, increasingly rely on cloud computing applications for functionality as diverse as "practice management, document management, payroll, human resources, tax systems, accounting, timekeeping, e-mail spam filtering, litigation support, data hosting, and office productivity,"[227] all of which may be accessed over the web rather than being installed on the law firm's own computers. This change in how software is actually made and used may carry important implications for the proper scope of intellectual property protection.

2. Public and Private Ordering

Relatedly, Congress should clarify the place of private ordering, through contracts and licensing, in the development and use of software. Private agreements on the enforcement of existing copyright rights have received

---

224. The need for more analytically rigorous assessment of copyright's actual effects on expressive output is not, of course, limited to the context of software. *See* HAGGART, *supra* note 212, at 52–53 ("the very question of copyright's empirical, as opposed to theoretical, effect on the production of creative works has gone largely unexamined. . . [T]he copyright debate has been driven much more by rhetoric than evidence.").

225. *See supra* notes 45–46 and accompanying text.

226. *See, e.g.*, Tim O'Reilly, *What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software*, 65 COMM. & STRATEGIES 17, 20 (2007) ("Software licensing and control over APIs—the lever of power in the previous era—is irrelevant because the software never need be distributed but only performed . . . ."); Horacio E. Gutiérrez, *Peering through the Cloud: The Future of Intellectual Property and Computing*, 20 FED. CIR. BAR J. 589, 604 (2011) ("Many cloud services will involve minimal distribution of software code to end users. . . . Thus, providers are likely to rely more heavily on access controls and technological protections to supplement or replace traditional copyright protection in the cloud."); Samuelson, *supra* note 99, at 1747–48.

227. Diane Murley, *Law Libraries in the Cloud*, 101 LAW LIBR. J. 245, 251 (2009) (footnote omitted).

general approbation from Congress,[228] but private agreements that supersede or extend those rights remain more controversial. The copyright statute strikes a particular balance between the rights of creators of expressive works and the rights of those who use the works.[229] Private agreements, however, may upset this balance: for example, a creator may condition a user's access to an expressive work upon the user's giving up some of the rights they would otherwise enjoy under the statute.[230] For this reason, many scholars have expressed concern that judicial deference to contract terms drafted solely with licensors' interests in mind may effectively upset Congress's statutory accommodation of competing interests, to the public's detriment.[231]

In a world in which most computer software is merely licensed rather than sold, the terms contained in licensing agreements may operate to constrain end-user behavior much more directly than the terms of the governing statute. Given the importance of licensing arrangements in this sector, a new CONTU commission should examine whether software licensors should be able to override longstanding legal doctrines such as first sale or fair use, or

228. *See Role of Voluntary Agreements in the U.S. Intellectual Property System: Hearing Before the Subcomm. on Courts, Intellectual Property, and the Internet of the House Comm. on the Judiciary*, 113 Cong., 1st Sess, Serial No. 113-49 (2014). Most of the witnesses who testified at this hearing expressed general support for the use of voluntary agreements to facilitate copyright enforcement—citing, for example, commitments among internet service providers to "three strikes" policies that impose escalating restrictions on users repeatedly accused of infringing activity.

229. *See, e.g.*, Twentieth Century Music Corp. v. Aiken, 422 U.S. 151, 156 (1975) (recognizing the copyright statute's "balance of competing claims upon the public interest"); Lenz v. Universal Music Corp., 815 F.3d 1145, 1152–53 (9th Cir. 2015) (describing copyright's fair use doctrine as a statutory authorization for user conduct with which copyright holders may not interfere — in essence, a user-held right); Patricia Aufderheide & Peter Jaszi, *Reclaiming Fair Use: How to Put Balance Back in Copyright*, 11 J. HIGH TECH L. 1 (2011) ("The fact that fair use is a defense does not make it any less of a right."). A classic formulation of the "balance of rights" approach is, of course, L. RAY PATTERSON & STANLEY W. LINDBERG, THE NATURE OF COPYRIGHT: A LAW OF USERS' RIGHTS (1991) (enumerating distinctive rights held by authors, publishers, and users in the copyright ecosystem).

230. *See, e.g.*, JASON MAZZONE, COPYFRAUD AND OTHER ABUSES OF INTELLECTUAL PROPERTY LAW 102–17 (2011) (noting, with many examples, that licensing agreements and other contracts are frequently drafted to enlarge the scope of publishers' rights beyond what copyright law provides and to place additional limits on how licensed works may be used).

231. *See, e.g.*, David Nimmer et al., *The Metamorphosis of Contract into Expand*, 87 CAL. L. REV. 17, 63–68 (1999) (suggesting several scenarios under which copyright's preemption doctrine should render particular licensing terms unenforceable); Niva Elken-Koren, *Copyright Policy and the Limits of Freedom of Contract*, 12 BERKELEY TECH. L. J. 93, 108–13 (1997) (highlighting risks to competition and to the creation and reuse of information); Maureen A. O'Rourke, *Drawing the Boundary Between Copyright and Contract: Copyright Preemption of Software License Terms*, 45 DUKE L. J. 479, 541–55 (1995) (articulating a framework under which courts would assess whether particular copyright licensing arrangements run afoul of other settled public policies). For an argument that the controversy has been overstated, however, see Guy A. Rub, *Copyright Survives: Rethinking the Copyright-Contracts Conflict*, 103 VA. L. REV. 1143 (2017).

whether and under what circumstances the terms of the statute should supersede incompatible contractual terms.

B. *Are Alternative Forms Superior to Copyright?*

The CONTU report settled on copyright protection for software essentially by default. Post-CONTU history, however, shows that the rejected alternatives may deserve a closer look.

1. The Patent Alternative

CONTU rejected patenting software on largely theoretical grounds. Based on subsequent developments, however, we now have several decades of experience with software patenting, and a richer appreciation of the pros and cons of extending patent protection to software.[232] Furthermore, the debate now has a global dimension, with some countries embracing patent protections for software and others resisting it.[233] The supporting arguments on both sides deserve a fair hearing and a thoughtful response. Perhaps even if patent protection is appropriate for some types of software, it is not appropriate for all types. Perhaps patent protection is inappropriate in any instance, and the core concerns of developers who wish to guard against wrongful takings of their code may be addressed adequately via copyright, contract, trade secrecy, or misappropriation principles, or through technological means.[234] Or perhaps patent protection is highly motivating to software development and should be retained or even expanded (in which case, legislative action to limit the reach of the Supreme Court's decision in *Alice* may be appropriate).[235] The point is simply that the issue is, at a minimum, ripe

---

232. *See, e.g.*, Mark H. Webbink, *A New Paradigm for Intellectual Property Rights in Software*, 2005 DUKE L. & TECH. REV. 12 (2005) (noting conflicting evidence on whether patent protection has positive effects on software innovation); Chad King, Note, *Abort, Retry, Fail: Protection for Software-Related Inventions in the Wake of* State Street Bank & Trust Co. v. Signature Financial Group, Inc., 85 CORNELL L. REV. 1118, 1159–69 (2000) (arguing that patent protection is superior to the available alternatives due to its focus on protecting software's functional attributes, which other forms of intellectual property protection exclude).

233. *See* Convention on the Grant of European Patents art. 52(2)(c) (excluding "programs for computers" from scope of under European patent law); *see also, supra* note 143.

234. *See Innovation in America (Part I and II): Hearings Before the Subcomm. on Courts, Intellectual Property, and the Internet of the House Comm. on the Judiciary*, 113th Cong., 1st Sess., Serial No. 113-47, at 83 (2014) ("Evidence as to whether providing software with patent protection in addition to copyright protection has promoted innovation is not encouraging.") (prepared statement of Computer & Communications Industry Association). For a cogent argument that strong patent protections for software cause producers to shift investments away from inventive activity to defensive measures that yield no social benefit, *see* Jason M. Schultz & Brian J. Love, *Brief of Amici Curiae Law, Business, and Economics Scholars in Support of Respondents in Alice Corp. Pty. Ltd. v. CLS Bank International, et al.*, 4 N.Y.U. J. INTELL. PROP. & ENT'MT L. 358 (2015).

235. This is the suggestion of David McKinney, Comment, *Alice: Tumbling Down the Rabbit Hole of Software Patent Eligibility*, 84 UMKC L. REV. 261, 280–81 (2015). If one concludes that the functional character of software entitles it to protection under patent law, a

for reexamination based on all that has happened in the years since the original CONTU.[236]

2. The Trade Secrecy Alternative

As already noted, the stated factual basis for CONTU's rejection of trade secrecy protection for software—to wit, that software is distributed to the public in a way that makes "secrecy" untenable—is an increasingly inaccurate description of reality.[237] In the meantime, CONTU's concerns about state-by-state variation in trade secret law have been at least partly ameliorated through the enactment of the first federal trade secrecy statute[238] and by the adoption by the great majority of states of the Uniform Trade Secrets Act.[239] These developments counsel revisiting CONTU's conclusion that trade secrecy protection is inadequate for software.[240]

3. *Sui Generis* Protection

Renowned copyright scholar Benjamin Kaplan recognized over half a century ago that software may not fit *any* of our existing analytical paradigms for intellectual property; it may be too functional for copyright but too expressive for patenting.[241] The original CONTU report selected copyright as the closest fit of the many imperfect alternatives it considered. At the time the original CONTU made its recommendations, copyright was essentially a

---

logical consequence should be a corresponding withdrawal of copyright protection. *See, e.g.*, Lemley, *supra* note 62, at 26 ("As software patents gain increasingly broad protection, whatever reasons there once were for broad copyright protection of computer programs disappear."); Pamela Samuelson, *Are Copyright and Patent Overlapping or Mutually Exclusive in Protecting Software Innovations?*, PATENTLY-O (May 27, 2017), https://patentlyo.com/patent/2017/05/overlapping-protecting-innovations.html (discussing 1991 study jointly prepared by USPTO and the U.S. Copyright Office that concluded that overlapping copyright and patent protection could not exist for software), *archived at* [perma.cc/S8C3-R8F4].

236. For an interesting suggestion that patent-like analysis should be imported into copyright law for software cases, yielding a hybrid form of protection, *see* Kristina Soderquist, *Yet Another Suggestion for Solving the Computer Program Dilemma*, 6 VA. J.L. & TECH. 14, ¶¶ 31–53 (2001).

237. *See supra* notes 45–46, 226–227 and accompanying text.

238. Defend Trade Secrets Act of 2016, Pub. L. No. 114-153, 130 Stat. 376. *See also* Tony Dutra, *New Trade Secret Law: More to Consider in Patent Trade-Off*, 92 PAT. TRADEMARK & COPYRIGHT J. (BNA) No. 2264, at 340 (June 3, 2016).

239. *See* UNIF. TRADE SECRETS ACT (amended 1985) (Nat'l Conference of Comm'r on Unif. State Laws 1985).

240. For an argument that trade secrecy protection may be a superior alternative to software patenting, *see* Michael Risch, *Hidden in Plain Sight*, 31 BERKELEY TECH. L.J. 1635 (2016); *but cf.* Derek E. Bambauer, *Secrecy is Dead—Long Live Trade Secrets*, 93 DENVER U. L. REV. 833, 846–49 (2016) (noting that expansion of trade secrecy protection potentially creates new conflicts with other significant social values).

241. *See supra* note 28 and accompanying text. Other calls for *sui generis* protection for software antedated the 1976 Act. *See, e.g.*, *supra* note 21 and accompanying text. And many scholars have taken up the call in the intervening decades. *See, e.g.*, Samuelson, et al., *supra* note 4, at 2312–13 n.6.

unitary body of law that applied the same rules to all expressive works. In later years, however, Congress has come to recognize that some industries are different and need different forms of protection tailored to their particular characteristics. For example, the copyright statute now includes chapters defining *sui generis* forms of legal protection for semiconductors[242] and the design of boat hulls and decks.[243] It would not be inconsistent with these more recent enactments to conclude that software, too, demands a nuanced form of legal protection attuned to its particular characteristics.[244] The possible benefits of providing *sui generis* protection to software works would have to be balanced, of course, against the possible costs in the form of fragmentation of doctrine and the introduction of concepts into the law that have no clear contextual meaning.[245]

C. *What Limits to Copyright Protection?*

A new CONTU commission could provide valuable guidance to the courts even if it adheres to the original CONTU Report's conclusion that copyright is the form of protection best suited to computer software.

1. Functional versus Expressive Variation

The courts in *Whelan* and *Oracle* both justified the extension of broad copyright protection to software on the grounds that programmers could have expressed their ideas in the software in multiple ways.[246] But this reasoning merely sidesteps the question, confronted forthrightly in *Altai* and *Lotus*, whether *functional* variation stands on the same legal footing as *expressive* variation in copyright.[247] Outside the software context, the courts in copyright cases have recognized a distinction between a variation in a work's utilitarian or functional attributes (which cannot receive copyright

242. *See* Semiconductor Chip Protection Act of 1984, 17 U.S.C. §§ 901–914 (2012)).

243. *See* Vessel Hull Design Protection Act, 17 U.S.C. §§ 1301-1332 (2012).

244. To consider only one of those characteristics: although works of art and literature may enjoy commercial success spanning multiple decades, most commercial software has a far shorter viable shelf life. Consider two copyrighted works released in the year 1985: the film *Back to the Future* and the Microsoft Windows 1.0 operating system. The former work continues to enjoy consumer demand three decades later and the latter does not. The fact that the copyright system treats both works as works made for hire and extends them each the same duration of legal protection (nearly a century) under 17 U.S.C. § 302(c) illustrates the occasionally strange consequences of treating software the same as all other expressive works.

245. *See, e.g.*, J.H. Reichman, *Legal Hybrids Between the Patent and Copyright Paradigms*, 94 COLUM. L. REV. 2432, 2502–03 (1994) (criticizing various forms of *sui generis* intellectual property protection as ignoring the copyright and patent acts' efforts to "balance the public interest in competition against specific incentives to create"; *sui generis* protections are "improvised responses to sectoral protectionist demands" that "lack coherent theoretical foundations and reflect different economic premises"); Leo J. Raskind, *The Uncertain Case for Special Legislation Protecting Computer Software*, 47 U. PITT. L. REV. 1131 (1985).

246. *See supra* notes 67–69, 110–113.

247. *See supra* notes 86–87, 98 and accompanying text.

protection) and variation in a work's expressive attributes (which can).[248] In the context of software, however, the courts' views on this question remain muddled, at least in part because the original CONTU Report's views on the same question were similarly uncertain.[249] A new CONTU may shed new light on the question and achieve a level of consensus that has so far eluded the courts.

2. Imitation, Incentives, and Copyright

A new CONTU should also explore the role, widely recognized in the intervening decades, that successful imitation of prior functionality plays in both the training of software developers and the making of software itself. To a degree perhaps underappreciated in the case law, radical innovation in software development can be dysfunctional, introducing problems that are avoided by making cautious, incremental changes to earlier working code.[250] To the extent that training and development in software engineering rests in part upon borrowing and imitation of earlier works, introducing excessively strong copyright principles into the picture may exert an inhibiting force. For that reason, it would be highly desirable for a new CONTU commission to enunciate principles relating to authorship, originality, and fair use of software works should it conclude that continued protection under copyright is the best available alternative.

---

248. *Compare* ATC Distribution Group, Inc. v. Whatever It Takes Transmissions & Parts, Inc., 402 F.3d 700, 706–10 (6th Cir. 2005) (finding a catalog listing of part numbers to show only functional variation rather than expressive variation); *with* American Dental Ass'n v. Delta Dental Plans Ass'n, 126 F.3d 977, 979 (7th Cir. 1997) (finding expressive variation in a taxonomy and written description of dental procedures).

249. *See, e.g.*, *supra* notes 48, 54 and accompanying text; *see also* Menell, *supra* note 119, manuscript at 46–47 (arguing that Federal Circuit's opinion in *Oracle v. Google* conflates expressive and technological variation).

250. This is a key insight of observers of the open-source software phenomenon. As one author put it:

> 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.

STEVEN WEBER, THE SUCCESS OF OPEN SOURCE 75 (2004); *see also* ERIC S. RAYMOND, THE CATHEDRAL & THE BAZAAR 4 (1999) ("[a]n important trait of the great [programmers] is constructive laziness. . .. it's almost always easier to start from a good partial solution than from nothing at all").