WHY 72 INTELLECTUAL PROPERTY SCHOLARS SUPPORTED GOOGLE'S COPYRIGHTABILITY ANALYSIS IN THE *ORACLE* CASE

Pamela Samuelson[†] *& Catherine Crump*[††]

ABSTRACT

In January 2020, 72 intellectual property scholars signed on to an *amicus curiae* brief in support of Google's position that it did not infringe Oracle's copyright when it incorporated parts of the Java Application Program Interface (API) into its Android software, an issue that came before the U.S. Supreme Court in *Google LLC v. Oracle America, Inc.*

In ruling that the declarations of the Java API at issue in this case, as well as the structure, sequence, and organization (SSO) embodied in the declarations, were protectable expression as a matter of U.S. copyright law, the Federal Circuit's 2014 *Oracle* decision adopted an erroneously narrow view of the Supreme Court's decision in *Baker v. Selden* and of congressional codification of *Baker*'s exclusion of systems and methods from copyright's scope in 17 U.S.C. § 102(b). Precedents from the Ninth and Second Circuits have persuasively held that program interfaces necessary for program compatibility are unprotectable by copyright law, decisions that the Federal Circuit either misconstrued or ignored.

The Federal Circuit also had a mistaken understanding of the merger doctrine as applied in computer program copyright cases. Its view is irreconcilable with *Baker* and other persuasive decisions. Case law from numerous circuits recognizes that when the design choices of subsequent programmers are constrained by the interface designs embodied in earlier programs, the merger doctrine applies so that programmers can reuse elements necessary to achieve compatibility.

Because of the Federal Circuit's numerous errors in analyzing Google's copyrightability defense, this Article, like the brief from which it is drawn, concludes that the Supreme Court would have been justified in overturning the Federal Circuit's ruling on copyrightability grounds. (The Supreme Court instead decided the case for Google on the fair use issue.) Allowing programmers to reuse interfaces that enable compatibility promotes the ongoing

DOI: https://doi.org/10.15779/Z38R49G952

The views expressed in this Article are original to the co-authors. Seventy-one other IP scholars, however, joined the brief, so our views of the issues were shared by others.

[†] Pamela Samuelson is Richard M. Sherman Distinguished Professor of Law, Berkeley Law. This Article is a derivative work of the Brief of 72 Intellectual Property Scholars as *Amici Curiae* in Support of Petitioner, Google LLC v. Oracle America, Inc., No. 18-956 (U.S. Jan. 13, 2020). The names of the signatories are not included in this Article but can be found in the brief, available at https://ssrn.com/abstract=3518887. Although one of us (Samuelson) has written several articles on the *Oracle v. Google* case and other software copyright rulings, we decided that this much more concise analysis of the issues warranted publication as a standalone article. We thank our signatories for their support and very helpful editorial suggestions.

[††] Catherine Crump is Clinical Professor of Law, Berkeley Law, and Director of the Samuelson Law, Technology & Public Policy Clinic.

progress in the field of computer programming as well as advancing the science of computing, in keeping with the constitutional purpose of copyright law.

TABLE OF CONTENTS

I. INTRODUCTION

The Federal Circuit's ruling against Google's copyrightability defense in *Oracle America, Inc. v. Google Inc.*[1] misconstrued the text of the Copyright Act, Supreme Court rulings, as well as software copyright case law persuasively establishing that interfaces that enable compatibility among programs are unprotectable by copyright law, thereby disrupting settled expectations of this $845 billion industry.[2] The U.S. Supreme Court granted Google's petition for certiorari and reviewed the Federal Circuit's decision in its October term 2020, ultimately deciding the case on fair use grounds.[3]

[1] 750 F.3d 1339 (Fed. Cir. 2014). For a detailed discussion of this case and the Federal Circuit's ruling on copyrightability, see, for example, Pamela Samuelson, *Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement*, 31 BERKELEY TECH. L.J. 1215 (2016). After the Federal Circuit's copyrightability ruling in 2014, Google petitioned the Supreme Court for review, which the Court denied in 2015. Subsequently, a jury found that Google's use of the Java API packages was a fair use, but the Federal Circuit reversed, concluding that Google's use was not fair as a matter of law. Oracle Am., Inc. v. Google LLC, 886 F.3d 1179, 1186 (Fed. Cir. 2018). The Supreme Court agreed to review the Federal Circuit's decisions regarding both copyrightability and fair use. *Id.*, *cert. granted*, 140 S. Ct. 520 (U.S. Nov. 15, 2019) (No. 18-956). This Article, like the brief from which it is drawn, focuses on the copyrightability issue only. While this Article was in process, the Court issued its opinion on April 5, 2021, reversing the Federal Circuit's fair use ruling and holding that Google's use of the Sun Java APIs was fair use as a matter of law. Google LLC v. Oracle Am., Inc., 141 S. Ct. 1183 (2021), *rev'g* 886 F.3d 1179 (Fed. Cir. 2018). Although the majority opinion written by Justice Breyer stated that it assumed, "for argument's sake," that the Java APIs were copyrightable, 141 S. Ct. at 1190, its fair use analysis employed reasoning and language that appear to support § 102(b) and merger arguments. *Id.* at 1192–93, 1201–02. Further discussion of the Court's opinion are beyond the scope of this Article. We remain convinced that the Court would have been justified in deciding this case on the § 102(b) and/or merger issues, as we believe the Federal Circuit's analysis of these issues is so deeply flawed.

[2] *See* BSA Foundation, *Software: Growing US Jobs and the GDP* (Sept. 2019), https://software.org/wp-content/uploads/2019SoftwareJobs.pdf (based on 2018 data).

[3] Oral argument in the *Google* case was originally scheduled for March 24, 2020, but because of the coronavirus pandemic, the Court moved this matter to its calendar for the October term 2020. On May 4, 2020, the Court requested supplemental briefing on the appropriate standard of review of the jury's fair use finding in favor of Google. Both parties

Until the Court of Appeals for the Federal Circuit's 2014 *Oracle* decision, software developers felt free to compete and innovate in the development of compatible software because major decisions from the Courts of Appeals for the Second and Ninth Circuits had established that copyright law does not protect software interfaces that enable the development of compatible programs.[4] These cases and their progeny recognized that unlike conventional literary works, computer programs are highly utilitarian.[5] They embody many copyright-unprotectable elements, such as compatibility-enabling interfaces, that must be filtered out before making infringement determinations. Computer programs consequently receive a relatively "thin" scope of copyright protection to ensure that subsequent programmers can freely reuse unprotectable elements in developing their own programs. As a matter of copyright law, the pro-compatibility decisions are sound as they facilitate fair competition by those who write new code while preserving copyright's role in protecting software from piracy and other wrongful appropriations.

The Federal Circuit's 2014 *Oracle* decision was a radical departure from these precedents and directly contradicted their rulings. It adopted an unduly narrow view of the Supreme Court's ruling in *Baker v. Selden*,[6] which excluded methods, systems, and their constituent elements from copyright's scope. It ignored Congress's codification of the method/system exclusions.[7] It misconstrued the case law properly interpreting those exclusions in relation to program interfaces.[8] The Federal Circuit also misapplied the merger doctrine and case law persuasively holding that interfaces that enable compatibility are unprotectable by copyright law. Because of the Federal Circuit's numerous

filed their respective supplemental letter briefs on August 7, 2020. The Court heard oral argument on October 7, 2020.

[4] *See* Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693 (2d Cir. 1992); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992).

[5] *See Altai*, 982 F.2d at 712; *Accolade*, 977 F.2d at 1524.

[6] 101 U.S. 99 (1880).

[7] 17 U.S.C. § 102(b).

[8] The District Court in *Oracle* relied on both the § 102(b) method/system exclusions and the merger doctrine in its analysis of the copyrightability issue. This practice is common. Once an author devises a particular method or system, there may be relatively few ways to express it. *See, e.g.*, Ho v. Taflove, 648 F.3d 489, 497–99 (7th Cir. 2011) (analyzing the copyrightability of a scientific model and equation under § 102(b) and the merger doctrine); Hutchins v. Zoll Med. Corp., 492 F.3d 1377, 1383–85 (Fed. Cir. 2007) (analyzing the copyrightability of the process of CPR and standard instructions for performing that process under § 102(b) and the merger doctrine); MiTek Holdings, Inc. v. Arce Eng'g Co., Inc., 89 F.3d 1548, 1556 n.19, 1557 n.20 (11th Cir. 1996) (analyzing application of copyright to a command tree structure under § 102(b) and the merger doctrine); Atari Games Corp. v. Nintendo of Am., Inc., 975 F.2d 832, 839–40 (Fed. Cir. 1992) (analyzing the copyrightability of a data stream for unlocking a console under both § 102(b) and the merger doctrine).

errors in analyzing Google's copyrightability defense, the Supreme Court would have been justified in overturning the Federal Circuit's ruling. Programmers should have to write their own implementation code, as Google did, but interfaces that enable compatibility should be free from copyright restrictions.

II. U.S. SUPREME COURT PRECEDENTS, THE TEXT OF THE COPYRIGHT ACT, AND SOUND COPYRIGHT POLICY REQUIRE THE EXCLUSION OF PROGRAM INTERFACES FROM COPYRIGHT'S SCOPE

Freedom to compete and innovate in the development of compatible software was first recognized in the Second Circuit's landmark decision in *Computer Associates International, Inc. v. Altai, Inc.*[9] It held that interfaces of computer programs that enable compatibility are unprotectable by copyright law.[10] It concluded that Altai did not infringe by reimplementing the same interface as Computer Associates in its competing scheduling program.[11] Later that year, the Ninth Circuit in *Sega Enterprises, Ltd. v. Accolade, Inc.*,[12] which cited approvingly to *Altai*, decided that the functional requirements for achieving compatibility are unprotectable by copyright law.[13] It characterized these requirements as "interface procedures" that are excluded from copyright protection under 17 U.S.C. § 102(b).[14] Accolade was thus free to adapt its videogames so that they could run on Sega's popular platform. Other courts followed these precedents.[15] *Altai* and *Accolade* recognized that the essentially utilitarian nature of computer programs means they embody many copyright-unprotectable elements, including interfaces that enable compatibility, hence programs enjoy "a relatively weak barrier against public access" to those unprotected elements.[16] This ensures that subsequent programmers can reuse those elements in developing their own programs.

[9] 982 F.2d 693 (2d Cir. 1992).

[10] *Id.* at 710.

[11] *Id.* at 715.

[12] 977 F.2d 1510 (9th Cir. 1992).

[13] *Id.* at 1522.

[14] *Id.*

[15] *See, e.g.*, Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1547 (11th Cir. 1996) (recognizing the need for compatibility between the defendant's application program and an operating system program).

[16] *Altai*, 982 F.2d at 712; *see also Accolade*, 977 F.2d at 1527 (finding that the incorporation of utilitarian elements in a computer program did not merit copyright protection).

Relying on these precedents and the method/system exclusions of 17 U.S.C. § 102(b), Google believed that the declarations of the Java API used in its Android software and the structure, sequence, and organization (SSO) embodied in the declarations were not within the scope of protection that copyright law provides to the work of authorship at issue,[17] namely, Java 2 SE (Java SE), whose contents include program code, specifications of the Java packages and their classes and methods, and related documentation.

The District Court made findings of fact from which it concluded that these declarations were not within the scope of protection that copyright law provided to Java SE.[18] It regarded the declarations as constituent elements of an interface system or method that should be excluded from the scope of copyright protection under 17 U.S.C. § 102(b).[19] This ruling is consistent with the Supreme Court's decision in *Baker v. Selden*, which held that the selection and arrangement of columns and headings in Selden's bookkeeping forms were not within the scope of protection that copyright law provided to his book.[20] The ruling is also consistent with congressional codification of *Baker*'s exclusion of methods and systems and with the Ninth Circuit's characterization of program interfaces as unprotectable procedures under § 102(b) in *Accolade*.[21] It is also consistent with the views of an information technology industry association known as the American Committee for Interoperable Systems (ACIS), whose founding member, Sun Microsystems, created the Java API.[22] In an amicus brief, ACIS advised the Court that "it can accurately be said that the interface specification is the 'system' or 'method of operation' that is 'expressed' by the program code."[23]

[17] According to the District Court, "all agree[] that Google had not literally copied the software but had instead come up with its own implementations of the 37 API packages." Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 975 (N.D. Cal. 2012). This is consistent with computer scientists' conception of the declarations as interfaces. *See* Brief of 78 Amici Curiae Computer Scientists in Support of Petitioner at 6, Google LLC v. Oracle Am., Inc., No. 18-956 (U.S. Feb. 25, 2019) [hereinafter 78 Computer Scientists Cert. Brief].

[18] *Oracle*, 872 F. Supp. 2d at 976.

[19] *Id.*

[20] 101 U.S. 99, 106 (1880).

[21] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1522 (9th Cir. 1992).

[22] Brief Amici Curiae of American Committee for Interoperable Systems and Computer & Communications Association in Support of Respondent at 1, n.1, Lotus Dev. Corp. v. Borland Int'l, Inc., 516 U.S. 233 (1996) (No. 94-2003) (listing ACIS membership).

[23] *Id.* at 19.

A. THE SUPREME COURT ORIGINATED THE EXCLUSION OF SYSTEMS, METHODS, AND THEIR CONSTITUENT ELEMENTS FROM THE SCOPE OF COPYRIGHT PROTECTION

Perris v. Hexamer was the first Supreme Court decision to rule that copyright protection does not extend to a system and its constituent parts embodied in a copyrighted work.[24] Perris sued Hexamer for using the same symbol system in a map of Philadelphia as Perris had used in a map of certain wards of New York City.[25] Both maps depicted the layout of lots and buildings using a set of symbols and color-coding to identify different types of buildings to aid in risk assessment for fire insurance purposes.[26] The Court concluded that Perris had "no more an exclusive right to use the form of the characters they employ to express their ideas upon the face of the map, than they have to use the form of type they select to print the key."[27] After all, Hexamer had not copied Perris's map but only "use[d] to some extent their system of arbitrary signs and their key."[28] The Court considered this system to be a "useful contrivance[] for the despatch of business."[29] It did not matter how original that system might have been or how many other symbol systems could have been devised. That system was simply not protectable by copyright law.

Soon thereafter, the Court reviewed a similar infringement claim in *Baker*. Because *Baker* is such a foundational case and its proper interpretation is disputed by the litigants, we provide some details about the case. Prior to Charles Selden's claimed invention of a novel bookkeeping system, the standard process by which officials kept account books was slow and inefficient. Bookkeepers had to record information about each transaction in a journal for accounts of that kind and then record details again in a ledger where all transactions were recorded in sequential fashion.[30] Because the relevant information was spread out over multiple volumes, it was difficult to prepare a balance sheet for each period and to detect errors or fraud.

Selden's key innovation was figuring out a way (as the book's title, *Selden's Condensed Ledger, or Bookkeeping Simplified,* suggests) to condense the journals

[24] 99 U.S. 674 (1879).

[25] *Id.* at 675.

[26] *Id.*

[27] *Id.* at 676.

[28] *Id.*

[29] *Id.* at 675.

[30] Supreme Court Record at 92, 106, Baker v. Selden, 101 U.S. 99 (1880) [hereinafter Record]. For further details about the *Baker* litigation, see Pamela Samuelson, *The Story of* Baker v. Selden: *Sharpening the Distinction Between Authorship and Invention*, *in* INTELLECTUAL PROPERTY STORIES 159 (Rochelle Cooper Dreyfuss & Jane C. Ginsburg eds., 2005) [hereinafter Baker *Story*]. To view simulations of the relevant forms, see *id.* at 170–71.

and ledger, so that users could record pertinent information about transactions and accounts on one page or two adjoining pages.[31] It enabled a much more efficient accounting process, making the preparation of trial balances and detection of errors and fraud much easier.[32]

Selden's sense of the magnitude of his achievement was expressed in the preface to his book: "To greatly simplify the accounts of extensive establishments doing credit business . . . would be a masterly achievement, worthy to be classed among the greatest benefactions of the age."[33] The preface revealed that Selden had sought a patent on forms embodying his system to "prevent their indiscriminate use by the public."[34]

Although Selden knew about Baker's competing book and similar forms during his lifetime, it was his widow who charged Baker with infringement, claiming that "the ruled lines and headings, given to illustrate the system, are a part of the book, and, as such, are secured by the copyright; and that no one can make or use similar ruled lines and headings . . . without violating the copyright."[35]

The Court had no doubt that a work on bookkeeping could be copyrighted or that it would be "a very valuable acquisition to the practical knowledge of the community."[36] But the Court perceived "a clear distinction between the book, as such, and the [useful] art which it is intended to illustrate."[37] Copyright law could protect the author's explanation of a useful art, but not the useful art itself, no matter how creative it was. "To give to the author of the book an exclusive property in the [useful] art described therein," the Court said, "would be a surprise and a fraud upon the public. That is the province of letters-patent, not of copyright."[38] That Mrs. Selden intended to assert patent-like rights through copyright is evident from her announcement to Baker's customers that they too were infringers.[39] Had Selden obtained the patent he sought, it would have given him and his heirs exclusive rights to control uses of the system, as well as making and selling the forms that embodied the system.[40] But no such patent had been issued.

[31] *Id.* at 160.

[32] *Id.*

[33] Record, *supra* note 30, at 21.

[34] *Id.* at 21–22.

[35] Baker v. Selden, 101 U.S. 99, 101 (1880).

[36] *Id.* at 102.

[37] *Id.*

[38] *Id.*

[39] Record, *supra* note 30, at 79–80.

[40] *See* Baker *Story*, *supra* note 30, at 174.

The Court recognized that Selden's claim seemed plausible because of the "peculiar nature of the [useful] art described in [his] books" in which "the illustrations and diagrams employed happen to correspond more closely than usual with the actual work performed by the operator who uses the art."[41] Someone who kept books using Selden's method would necessarily use forms with the same or substantially similar headings and columns. Usually, the Court observed, useful arts are "represented in concrete forms of wood, metal, stone, or some other physical embodiment."[42] But "the principle is the same in all" regardless of whether the useful art is embodied in a writing or in metal.[43] The Court concluded that Selden's system was unprotectable by copyright law, as were the ruled lines and headings that instantiated the system.[44]

Baker illustrates why copyright law should allow second comers to build upon methods and systems embodied in a first author's works and why authors of writings on methods and systems should not have too much control over subsequent adaptations of these creations. Selden's forms may have been a substantial improvement over the old-fashioned bookkeeping methods previously in use, but they were only one stage in the evolving art of bookkeeping. Selden's death meant that any further innovation in this field would have to come from others. Baker advanced the state of the art by redesigning the forms so that entries could be made as transactions occurred rather than having to wait until the end of the week or month as Selden's forms required.[45] Baker went on to write other books and he, not Selden, can be credited with having advanced the state of the art of bookkeeping in the nineteenth century.[46] Had Mrs. Selden prevailed, further improvements in the bookkeeping field might well have been stunted until Selden's copyrights expired. This outcome would have disserved both patent and copyright goals, as it would have slowed progress in the science and useful art of bookkeeping.

B. CONGRESS CODIFIED THE WELL-ESTABLISHED EXCLUSION OF SYSTEMS AND METHODS IN § 102(b)

Dozens of cases followed *Baker*'s conclusion that methods, systems, and their constituent elements are beyond the scope of copyright protection in writings that embody useful arts. Two courts, for example, rejected claims of infringement against authors who wrote books about the plaintiffs' original

[41] *Baker*, 101 U.S. at 104.

[42] *Id.* at 105.

[43] *Id.*

[44] *Id.* at 106.

[45] Baker *Story*, *supra* note 30, at 162.

[46] *See id.* at 169, 193 n.76.

shorthand systems: *Brief English Systems, Inc. v. Owen* and *Griggs v. Perrin.*[47] Another court, in *Aldrich v. Remington Rand, Inc.*, dismissed a claim of infringement for copying the plaintiff's tax record system, which Aldrich claimed to be "the most modern and efficient system of property revaluation for tax purposes."[48] Numerous other *Baker*-inspired cases ruled that original methods and systems for contests, games, rules, and strategies for playing games were beyond the scope of copyright protection.[49]

The *Baker*-inspired exclusions of methods and systems from copyright's scope was so well-established that Congress decided to codify these exclusions in the Copyright Act of 1976. Thus, 17 U.S.C. § 102(b) provides: "In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work." By codifying the method and system exclusions in § 102(b), Congress sought to ensure that the copyright in computer software and other works that embody functional elements would not be construed too broadly.[50] Courts in software copyright cases accordingly are charged with applying the § 102(b) exclusions as meaningful limits on the scope of copyright protection available to computer programs.

C. THE FEDERAL CIRCUIT'S *ORACLE* DECISION IGNORED THE § 102(b) SYSTEM/METHOD EXCLUSIONS

In *Oracle*, the Federal Circuit ruled that the Java declarations used by Google were copyright-protectable expression as a matter of law.[51] In so doing, the court articulated a very narrow understanding of § 102(b) and misconstrued the way in which the District Court analyzed the copyrightability issue in this case.

The Federal Circuit's *Oracle* opinion focused on whether copyright extends at all to works that incorporate functional elements.[52] We certainly agree with the Federal Circuit that § 102(b) should not be interpreted so literally that it

[47] *See* Brief English Sys., Inc. v. Owen, 48 F.2d 555 (2d Cir. 1931); Griggs v. Perrin, 49 F. 15 (C.C.N.D.N.Y. 1892).

[48] 52 F. Supp. 732, 733 (N.D. Tex. 1942).

[49] *See* Pamela Samuelson, *Why Copyright Law Excludes Systems and Processes from the Scope of Its Protection*, 85 TEX. L. REV. 1921, 1936–44 (2007) (reviewing post-*Baker* method/system copyright cases).

[50] *See* H.R. REP. NO. 94-1476, at 57 (1976); S. REP. NO. 94-473, at 54 (1976).

[51] Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1368 (Fed. Cir. 2014).

[52] *Id.* at 1367. The Federal Circuit seemed to characterize the District Court's order as suggesting that computer programs are per se uncopyrightable due to their functional aspects, *id.*, but that overbroad characterization misconstrues the District Court's analysis, see Samuelson, *supra* note 1, at 1256.

would deprive authors of machine-executable programs of the copyrights that Congress intended them to have just because programs are machine processes.[53]

The Federal Circuit's *Oracle* decision may have properly quoted the text of § 102(b),[54] but it treated ideas as the only unprotectable element of copyrighted software, giving no substantive meaning to the other seven terms of exclusion.[55] This is, as Justice Scalia once stated, "a stark violation of the elementary principle that requires an interpreter 'to give effect, if possible, to every clause and word of a statute,' " to which he added:

> Lawmakers sometimes repeat themselves . . . [They] do not, however, tend to use terms that "have no operation at all." So while the rule against treating a term as a redundancy is far from categorical, the rule against treating it as a nullity is as close to absolute as interpretive principles get.[56]

When a statute such as § 102(b) specifically identifies several categories of uncopyrightable elements and says "[i]n no case" should any of these be within the scope of copyright's protection, reading all but one of the terms out of the statute, as the Federal Circuit did in *Oracle*, violates this rule. It thus failed to be "deferential to the judgment of Congress in the realm of copyright."[57]

Although the text of § 102(b) is unambiguous in light of the holdings in *Baker* and its progeny, it is worth noting that Congress added the method/system exclusions to the statute, in part, to allay concerns about the risk of an excessive scope of copyright protection for software:

> Some concern has been expressed lest copyright in computer programs should extend protection to the methodology or processes adopted by the programmer, rather than merely to the "writing" expressing his ideas. Section 102(b) is intended, among other things, to make clear that the expression adopted by the programmer is the copyrightable element in a computer program, and that the actual processes or methods embodied in the program are not within the scope of the copyright law.[58]

[53] *See Oracle*, 750 F.3d at 1367.

[54] *Id.* at 1354.

[55] *Id.* at 1367.

[56] King v. Burwell, 576 U.S. 473, 502 (2015) (Scalia, J., dissenting) (citations omitted).

[57] Eldred v. Ashcroft, 537 U.S. 186, 198 (2003).

[58] H.R. REP. NO. 94-1476 at 57 (1976); S. REP. NO. 94-473 at 54 (1975). During hearings on copyright revision bills, several witnesses recommended adoption of a specific provision to limit the scope of copyright protection in computer programs. *See Copyright Law Revision, Hearings before the Subcomm. on Patents, Trademarks, & Copyrights of the S. Comm. on the Judiciary*, 90th Cong. 196–97 (1967) (statement of Arthur R. Miller). Miller foresaw a risk that

The Federal Circuit tellingly recited only that part of the legislative history stating that § 102(b) codified the idea/expression distinction[59] and omitted congressional expressions of concern about excessive copyright protection for software. It overlooked the Supreme Court's directive not to "alter the delicate balance Congress has labored to achieve."[60]

1. *The Method and System Exclusions of § 102(b) Avert Patent/Copyright Overlaps*

Consistent with the *Baker* tradition, codification of the system/method exclusions in § 102(b) aims, in part, to ensure that domains of copyright and patent protection for programs should be kept separate. The Federal Circuit once recognized this purpose in *Atari Games Corp. v. Nintendo of America, Inc.*[61] After quoting § 102(b)'s exclusion of procedures, processes, systems, and methods of operation, it stated that patent and copyright laws protect "distinct aspects" of programs.[62] The role of copyright, said the court, was to protect program expression, not any methods or processes that might be eligible for patenting under the Patent Act.[63]

The Federal Circuit's *Oracle* decision, however, instead seemingly endorsed the view that computer program innovations such as interfaces were eligible for both copyright and patent protection.[64] This was pertinent because both Sun and Oracle had obtained utility patents on program interfaces.[65]

2. *Unprotectable Elements in Computer Programs Must Be Filtered Out Before Assessing Infringement*

The Federal Circuit's *Oracle* decision also failed to recognize that the utilitarian nature of computer programs differentiates them from conventional literary works because programs contain many functional design elements, including methods and systems, that are beyond the scope of copyright under

courts would interpret copyright to extend to computer processes "that the program uses to achieve a functional goal," which would confer "patentlike [sic] protection under the guise of copyright." *Id.* at 197. He recommended that Congress should affirm that copyright would extend "solely to duplication or replication of the program" and not to "the art, process or scheme that is fixed in the program" because only patent law could protect "systems, schemes, and processes." *Id.* at 197, 199. For a fuller discussion of the genesis of § 102(b) exclusions, see Samuelson, *supra* note 49, at 1944–61.

[59] *Oracle*, 750 F.3d at 1357.

[60] Stewart v. Abend, 495 U.S. 207, 230 (1990).

[61] 975 F.2d 832 (Fed. Cir. 1992).

[62] *Id.* at 839.

[63] *Id.*

[64] *Oracle*, 750 F.3d at 1380 (erroneously quoting Mazer v. Stein, 347 U.S. 201, 217 (1954), which considered only potential design patent and copyright overlaps).

[65] Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 996 (N.D. Cal. 2012).

§ 102(b) and other doctrines.[66] The higher quantum of unprotectable elements in programs, as compared with novels, explains why courts such as the Second Circuit in *Altai* have directed that numerous types of unprotectable elements of programs be "filtered out" before deciding infringement claims in software copyright cases.[67] Although the Federal Circuit criticized the lower court for not following *Altai*,[68] the appellate court itself performed no filtration whatsoever.

3. *Methods and Systems Are Part of Program Structure, Sequence, and Organization, So SSO Obscures Rather Than Clarifies Expressive Aspects of Software*

The Federal Circuit accepted without question Oracle's claim that the SSO of computer programs is protectable expression.[69] By contrast, the Second Circuit wisely recognized in *Altai* that SSO is not a useful term with which to distinguish nonliteral elements of programs that may be expressive enough to be copyright-protectable from nonliteral elements that are excluded from copyright protection.[70]

By their very nature, methods and systems, when embodied in computer programs, are parts of SSO. Under the Federal Circuit's *Oracle* decision, it would be trivially easy for software developers to claim SSO copyright protection in methods or processes for which they failed to seek patent protection, or even to claim SSO copyright protection in processes for which patent protection is now unavailable in the aftermath of *Alice Corp. v. CLS Bank International.*[71] The Federal Circuit's ruling thus undermines the Supreme Court's holding in *Alice*.

D. KEY POST-1976 ACT DECISIONS FOLLOW *BAKER* IN EXCLUDING METHODS, SYSTEMS, AND THEIR CONSTITUENT ELEMENTS FROM COPYRIGHT'S SCOPE

An exemplary decision applying *Baker* and § 102(b) to exclude systems and their constituent parts from the scope of copyright is *Bikram's Yoga College of India, L.P. v. Evolation Yoga, LLC.*[72] Similar to *Baker*, in which Selden claimed copyright in the selection and arrangement of headings and columns in his

[66] *See, e.g.*, Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1524 (9th Cir. 1992).

[67] Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 706–11(2d Cir. 1992).

[68] *Oracle*, 750 F.3d at 1358.

[69] *Id.* at 1365.

[70] *Altai*, 982 F.2d at 706.

[71] 573 U.S. 208, 212 (2014) (holding a computer program patent-ineligible because its claim merely consisted of an abstract idea implemented on a generic computer).

[72] 803 F.3d 1032 (9th Cir. 2015).

novel bookkeeping forms, Bikram Choudhury claimed copyright in a sequence of twenty-six yoga poses and two breathing exercises described and illustrated in books and videos.[73] After Evolation Yoga began teaching the same sequence, Bikram's Yoga College sued it for infringement. Relying on *Baker* and its codification in § 102(b), the Ninth Circuit held that the Bikram Yoga Sequence was "not a proper subject of copyright protection."[74]

It did not matter whether Choudhury's arrangement of poses and breathing exercises was beautiful or graceful.[75] Nor did it matter that "the Sequence may possess many constituent parts," for "[v]irtually any process or system could be dissected in a similar fashion."[76] Also irrelevant was "that similar results could be achieved through a different organization of yoga poses and breathing exercises."[77] What mattered was that "[a]n essential element of this 'system' is the order in which the yoga poses and breathing exercises are arranged."[78] Choudhury's books directed his pupils to perform the yoga moves "in the strict order given in this book."[79] Choudhury had, moreover, repeatedly characterized his sequence as a method or system for improving health and well-being, which rendered the system and its constituent parts too functional for copyright protection.[80]

As in *Baker*, the Ninth Circuit in *Bikram* opined that to get exclusive rights in a functional system, such as the Yoga Sequence, it would be necessary to obtain a patent.[81] As in *Baker*, copyright protected Choudhury's explanation of his method or system, not the system itself or downstream uses of it. His books invited readers to practice the method the books taught.[82] Echoing *Baker*, the Ninth Circuit said that this objective "would be frustrated if the knowledge could not be used without incurring the guilt of piracy of the book."[83] "Consumers would have little reason to buy Choudhury's book if Choudhury held a monopoly on the practice of the very activity he sought to popularize,"[84] just as it would make little sense for consumers to buy Selden's book unless

[73] *Id.* at 1035–36.

[74] *Id.* at 1034. *Bikram* discusses *Baker* and its progeny. *Id.* at 1037–38.

[75] *Id.* at 1040.

[76] *Id.* at 1041.

[77] *Id.* at 1042.

[78] *Id.* at 1039.

[79] *Id.*

[80] *Id.* at 1038–39.

[81] *Id.* at 1039–40.

[82] *Id.* at 1035.

[83] *Id.* at 1041 (quoting Baker v. Selden, 101 U.S. 99, 103 (1880)).

[84] *Id.*; *see also* Ho v. Taflove, 648 F.3d 489, 498–99 (7th Cir. 2011) (holding a scientific model and its constituent elements unprotectable by copyright law); Palmer v. Braun, 287 F.3d 1325, 1334 (11th Cir. 2002) (holding meditation exercises were uncopyrightable processes).

they would thereby have the right to make use of the system and the forms that embodied it.

Consistent with *Bikram* was the Ninth Circuit's *Accolade* decision, which stated that program "interface procedures" that constituted "the functional requirements for [achieving] compatibility" were unprotectable by copyright law under 17 U.S.C. § 102(b).[85] While these statements appeared in a ruling that Accolade's reverse engineering of Sega program code was fair use, they were not mere dicta nor of only slight importance to the outcome of the fair use ruling, as the Federal Circuit asserted.[86] The statements were the very linchpin of the *Accolade* ruling. Accolade's disassembly and reverse engineering of Sega object code was legitimate because disassembly was "necessary in order to understand the functional requirements for Genesis compatibility."[87]

The Ninth Circuit explained that "[i]f disassembly of copyrighted object code is *per se* an unfair use, the owner of the copyright gains a *de facto* monopoly over the functional aspects of his work—aspects that were expressly denied copyright protection by Congress."[88] Channeling *Baker*, the Ninth Circuit said that if Sega wanted to enjoy a legal monopoly over the interface procedures, it would have to "satisfy the more stringent standards imposed by the patent laws."[89] Allowing reverse engineering would enable new entrants such as Accolade to make compatible products available in the market.[90]

Compatibility considerations were also important in *Lotus Development Corp. v. Borland International, Inc.*[91] Lotus charged Borland with infringement for reusing the Lotus 1-2-3 menu command hierarchy for the emulation mode of its competing spreadsheet program.[92] The District Court held that this hierarchy was protectable SSO because there were other ways to organize commands for spreadsheet program functions.[93]

The First Circuit recognized that "Borland had to copy the Lotus menu command hierarchy" if it wanted to enable users "to operate its programs in

[85] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1522 (9th Cir. 1992).

[86] Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1369 (Fed. Cir. 2014).

[87] *Accolade*, 977 F.2d at 1526. Baker made account books based on a similar system to Selden's in order to create a comparable and competitive product. *See Baker*, 101 U.S. at 101.

[88] *Accolade*, 977 F.2d at 1526 (citing § 102(b)).

[89] *Id.*

[90] *See id.* at 1523–24; *see also* Sony Comput. Entm't, Inc. v. Connectix Corp., 203 F.3d 596, 605, 608 (9th Cir. 2000) (finding reverse engineering to achieve partial compatibility fair use); Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 1000 (N.D. Cal. 2012) ("Contrary to Oracle, 'full compatibility' is not relevant to the Section 102(b) analysis.").

[91] 49 F.3d 807 (1st Cir. 1995), *aff'd by an equally divided Court*, 516 U.S. 233 (1996) (per curiam).

[92] *Id.* at 810.

[93] *Id.* at 810–11.

substantially the same way" as Lotus 1-2-3.[94] Borland's emulation mode enabled users of the Lotus program who had constructed macros for common sequences of functions to port those macros to Borland's program.[95] For those macros to be executable, Borland had to employ the same command terms arranged in exactly the same order. As the First Circuit explained:

> Under the district court's holding, if the user wrote a macro to shorten the time needed to perform a certain operation in Lotus 1-2-3, the user would be unable to use that macro . . . Rather, the user would have to rewrite his or her macro using that other program's menu command hierarchy. This is despite the fact that the macro is clearly the user's own work product.[96]

The First Circuit concluded that this menu command hierarchy was an unprotectable method of operating a spreadsheet program under § 102(b).[97]

Judge Boudin, concurring, observed:

> If Lotus is granted a monopoly on this pattern, users who have learned the command structure of Lotus 1-2-3 or devised their own macros are locked into Lotus, just as a typist who has learned the QWERTY keyboard would be the captive of anyone who had a monopoly on the production of such a keyboard.[98]

Lotus' command hierarchy "look[s] hauntingly like the familiar stuff of copyright; but the 'substance' probably has more to do with problems presented in patent law."[99]

E. CONSISTENT WITH *BAKER* AND § 102(b), PROGRAM INTERFACES SHOULD BE CONSIDERED UNPROTECTABLE PROCEDURES, METHODS, OR SYSTEMS

The Supreme Court articulated a clean distinction in *Baker* between the copyrightable expression in Selden's book and the uncopyrightable bookkeeping system, constituent elements of which were embodied in the forms.[100] A clean distinction is also possible in *Oracle*. Google and Java programmers around the world should be free to use the Java SE declarations

[94] *Id.* at 816. The First Circuit invoked *Baker*, noting that "Lotus wrote its menu command hierarchy so that people could learn it and use it," thus "fall[ing] squarely within the prohibition on copyright protection established in *Baker v. Selden* and codified by Congress in § 102(b)." *Id.* at 817.

[95] *Id.* at 811–12.

[96] *Id.* at 818.

[97] *Id.* at 817–18.

[98] *Id.* at 821 (Boudin, J., concurring).

[99] *Id.* at 820.

[100] *See* Baker v. Selden, 101 U.S. 99, 100–01 (1880).

to develop compatible programs, subject only to the norm that they must instantiate those interfaces in independently written code that copyright law protects from misappropriation.

Characterizing program interfaces as unprotectable procedures under § 102(b) is consistent with *Baker*, the text of § 102(b), and the case law properly interpreting it. The District Court's characterization of the declarations as methods or systems is similarly consistent, as was the ACIS amicus brief in *Borland.*[101] Interfaces are methods insofar as they enable one program to function effectively with other software or with hardware. Some program interfaces are relatively simple, as in *Accolade*, while others are more complex, as in *Oracle*. But as the Supreme Court so aptly said in *Baker*, "the principle is the same in all."[102] Allowing programmers to reuse interfaces that enable compatibility promotes the ongoing progress in the field of computer programming as well as advancing the science of computing, in keeping with the constitutional purpose of copyright law.[103]

III. THE FEDERAL CIRCUIT'S MERGER ANALYSIS IS IRRECONCILABLE WITH *BAKER* AND OTHER PERSUASIVE DECISIONS

The merger doctrine is often traced to the Supreme Court's *Baker* decision.[104] In *Baker*, the Court concluded that the forms embodying Selden's bookkeeping system were unprotected by copyright law because using these or similar arrangements of columns and headings was necessary to implement the underlying system.[105] As the Court explained:

> [W]here the [useful] art [a work] teaches cannot be used without employing the methods and diagrams used to illustrate the book, or such as are similar to them, such methods and diagrams are to be considered as *necessary incidents* to the art, and given therewith to the public; not given for the purpose of

[101] *See* Brief Amici Curiae of American Committee for Interoperable Systems and Computer & Communications Association in Support of Respondent, *supra* note 22.

[102] *Baker*, 101 U.S. at 105.

[103] U.S. CONST. art. I, § 8, cl. 8.

[104] *See, e.g.*, Arica Inst., Inc. v. Palmer, 970 F.2d 1067, 1076 (2d Cir. 1992); Pamela Samuelson, *Reconceptualizing Copyright's Merger Doctrine*, 63 J. COPYRIGHT SOC'Y U.S.A. 417, 419–20 (2016). While *Baker* did not originate the term "merger," it nonetheless articulated principles congruent with what came to be known as the merger doctrine and that guide the outcome here.

[105] *Baker*, 101 U.S. at 103.

> publication in other works explanatory of the art, but for the purpose of practical application.[106]

This "necessary incidents" language serves to prevent copyright from extending to unprotectable systems when the reuse of some expression is inseparable from the systems.

The Federal Circuit's analysis of the merger doctrine in *Oracle* cannot be reconciled with *Baker*. It is, moreover, contrary to persuasive authorities recognizing the merger doctrine as a shield against infringement for software interfaces that enable the development of compatible programs. Consistent with these authorities, the District Court found that the declarations had to be identical for the functionality they enable to be available in Android, leading it to conclude correctly that the merger doctrine barred Oracle's infringement claim.[107]

A. THE FEDERAL CIRCUIT'S ANALYSIS OF THE MERGER DOCTRINE IS AT ODDS WITH *BAKER* IN THREE KEY RESPECTS

When Charles Selden devised his novel bookkeeping system, he could have designed it in a number of ways. The *Baker* Court recognized that anyone who wanted to implement the Selden system would have little choice but to select and arrange columns and headings in a substantially similar way.[108] Since copyright does not protect useful arts such as bookkeeping systems, but only authorial expression,[109] Baker was free to publish similar forms to instantiate the Selden system. The Court ruled that the forms were uncopyrightable.[110] *Baker* importantly distinguished between authorship (the original expression that copyright protects) and invention (the functional creativity, which only utility patent law can protect).[111]

With regard to merger, the Federal Circuit conflicts with *Baker* in three significant ways. *First*, the Federal Circuit incorrectly concluded that merger can only be found if a first author had no or only extremely limited alternative ways to express an idea when creating his work.[112] For example, it pointed to

[106] *Id.* (emphasis added).

[107] Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 976 (N.D. Cal. 2012).

[108] *Baker*, 101 U.S. at 101.

[109] *Id.* at 101–02.

[110] *Id.* at 105.

[111] *Id.*

[112] The Federal Circuit construed the merger doctrine inconsistently. It correctly describes the merger doctrine as applying "when there are a limited number of ways to express an idea," Oracle Am., Inc. v. Google Inc., 750 F.3d 1339, 1359 (Fed. Cir. 2014), but elsewhere it incorrectly characterized the doctrine as applying exclusively when an idea "can be expressed in only one way." *Oracle*, 750 F.3d at 1360.

the existence of alternative names for Java functions, such as "Arith.larger" instead of "Math.max," in finding that the merger doctrine did not apply to the Java SE declarations.[113] In *Baker*, it did not matter whether column headers such as "Bro'ght Forward." or "Distribution" could have been worded differently when implementing Selden's accounting system.[114]

Thus, merger is a viable argument against copyrightability when the range of available alternatives for functions is limited, as the District Court concluded,[115] and as was true in *Baker*. The District Court made a finding that there *was*, in fact, only one way to write the name of each function: "Under the rules of Java, [declarations] *must be identical* to declare a method specifying the *same* functionality—even when the implementation is different."[116] Thus, any programmer wishing to invoke the functionality of "Math.max" would have to use the exact phrase "Math.max."[117]

Second, the Federal Circuit's opinion conflicts with *Baker* in concluding that courts in merger cases can consider only constraints on the plaintiff's creation and never constraints on the defendant's expressive choices.[118] The Court in *Baker* did not consider whether Selden's own choices in designing a bookkeeping system were constrained. Nor is there anything in *Baker* suggesting that the Court rejected Selden's copyright claim because Selden had no choice about how to select and arrange columns and headings for his bookkeeping forms. Indeed, Baker's forms were somewhat different.[119] Instead, the Court decided that once Selden designed his bookkeeping system,

[113] *Id.* at 1361.

[114] *See* Baker *Story*, *supra* note 30, at 171 (showing replica of a page from Selden's bookkeeping form).

[115] Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 997 (N.D. Cal. 2012).

[116] *Id.* at 976.

[117] Courts since *Baker* have also concluded that merger is an available defense when there is a limited number of alternatives (and not just one choice). For example, in *Morrissey v. Procter & Gamble Co.*, the court concluded that a set of sweepstakes rules Morrissey authored was original, and that there were different ways to express the rules. 379 F.2d 675, 678 (1st Cir. 1967). However, the court noted that the range of possible expressions of sweepstakes rules admitted of little variation "so that 'the topic necessarily requires' if not only one form of expression, at best only a limited number, [so] to permit copyrighting would mean that a party or parties, by copyrighting a mere handful of forms, could exhaust all possibilities of future use of the substance." *Id.* (citations omitted). The court rejected this outcome, writing that "[w]e cannot recognize copyright as a game of chess in which the public can be checkmated." *Id.* at 679 (citing *Baker*). *Cf.* TrafFix Devices, Inc. v. Mktg. Displays, Inc., 532 U.S. 23, 33–34 (2001) (rejecting test for functionality of trade dress based solely on the existence of alternative designs).

[118] *See Oracle*, 750 F.3d at 1361.

[119] *See* Baker v. Selden, 101 U.S. 99, 101 (1880).

Baker's design choices for arranging columns and headings to implement the same system were constrained by the choices that Selden had made.[120]

Third, the Federal Circuit's decision conflicts with *Baker* in holding that merger can be a defense to infringement claims, but not a basis for denying copyrightability.[121] The Court in *Baker* held that Selden's forms were uncopyrightable because the selection and arrangement of columns and headings were embodiments of the bookkeeping system.[122] Thus, the merger doctrine can be part of the copyrightability analysis and is not solely a defense to infringement.

There is a consensus among major authorities in copyright law that merger can present a copyrightability issue, not just a defense to infringement. Two major treatises now recognize that merger can serve as a bar to copyrightability.[123] The U.S. Copyright Office's *Compendium of U.S. Copyright Office Practices* also identifies merger as one of the bases on which the Office may refuse registration applications.[124]

B. The Merger Doctrine Provides a Sound Basis for Holding That Program Interfaces That Enable Compatibility Are Uncopyrightable

Since the Second Circuit's *Altai* decision, there has been broad-based consensus that computer program interfaces that enable the development of compatible software programs are not within the scope of copyright protection. Computer Associates claimed that Altai infringed by copying the structure of the compatibility component of its scheduling software designed

[120] *Id.* More recent appellate decisions also support the idea that a first comer's choices can limit the options of those who come after. In *Veeck v. Southern Building Code Congress International, Inc.*, 293 F.3d 791 (5th Cir. 2002) (en banc), merger precluded enforcement of SBCCI's claim against Veeck for his online posting of a privately written code that had been adopted as law in Anna and Savoy, Texas. 293 F.3d at 800–02. It did not matter how many possible alternative expressions existed when the codes were initially created. *Id.* What mattered was that once enacted, there was no other way to express what the law was. *Id.* at 802.

[121] *Oracle*, 750 F.3d at 1358.

[122] *Baker*, 101 U.S. at 107; *see also* Kern River Gas Transmission Co. v. Coastal Corp., 899 F.2d 1458, 1463–64 (5th Cir. 1990) (holding that the idea of pipeline location and its embodiment in a map are inseparable and not copyrightable); Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971) (citing *Baker*, 101 U.S. at 103, and concluding no copyright attached in a jeweled bee pin whose idea and expression merged).

[123] *See* 1 PAUL GOLDSTEIN, GOLDSTEIN ON COPYRIGHT §§ 2.3.2, 2:38.1 (2015); 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2A.05[A][2][b] (2019).

[124] *See* U.S. COPYRIGHT OFF., COMPENDIUM OF U.S. COPYRIGHT OFFICE PRACTICES § 313.3(B) (3d ed. 2015).

to run on IBM operating systems.[125] The *Altai* court invoked *Baker* as the "doctrinal starting point" of its analysis.[126]

Altai articulated a three-step "abstraction, filtration, and comparison" test for judging non-literal infringement of software copyrights.[127] The first step creates a hierarchy of abstractions of the plaintiff's program; the second step filters out unprotectable elements; and the third step compares the remaining expressive elements of the plaintiff's program with the defendant's program to determine if the defendant's program is substantially similar to expressive elements copied from the plaintiff's program.[128] Among the unprotectable elements to be filtered out are those dictated by efficiency, those constrained by external factors—such as the need to be compatible with hardware or software—and those in the public domain.[129] The court concluded that the similarities between Altai's and Computer Associates's programs were constrained by external factors, namely, the need to be compatible with IBM programs.[130]

Courts have invoked the merger doctrine in concluding that even literal copying may be excused from infringement when needed to achieve compatibility.[131] The Federal Circuit once recognized this principle in *Atari Games Corp. v. Nintendo of Am. Inc.*[132] Atari Games claimed its copying of Nintendo's data stream was necessary to enable videogames to run on its platform.[133] Had Atari Games copied only as much of the Nintendo data stream as was actually necessary to achieve compatibility with the then-current version of the Nintendo platform, the Federal Circuit said it would have ruled

[125] Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 701 (2d Cir. 1992).

[126] *Id.* at 704.

[127] *Id.* at 706–11.

[128] *Id.*

[129] *Id.* at 707–10.

[130] *Id.* at 714–15.

[131] The only decision—other than the Federal Circuit ruling in *Oracle*—to cast doubt on the lack of copyright protection for computer program elements required for interface compatibility was the Third Circuit's decision in *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240 (3d Cir. 1983). Its anti-compatibility dicta should be given little weight for two reasons. First, Franklin made no effort to reimplement the interface procedures embedded in the Apple OS in independently written code. It made exact copies of the Apple programs. *Apple Computer*, 714 F.2d at 1245. Second, these statements were made at an early stage in the evolution of software copyright law, well before *Altai* and other cases described above provide more thorough analyses of the copyright implications of a second comer's reimplementation of interface procedures necessary for interoperability.

[132] 975 F.2d 832, 840 (Fed. Cir. 1992).

[133] *Id.* at 836–37.

in Atari's favor on merger grounds.[134] Because it copied more than was necessary, its merger defense failed.[135]

Drawing in part on *Atari Games*, the Sixth Circuit concluded that literal copying of program code to enable compatibility was justifiable under the merger doctrine in *Lexmark Int'l, Inc. v. Static Control Components*, *Inc.*[136] Lexmark challenged Static's copying of a program installed in Lexmark printer cartridges. Static defended by saying this copying was necessary for its chip customers to manufacture printer cartridges that interoperated with Lexmark printers.[137] There was no other way for unlicensed cartridges to perform the digital handshake with Lexmark's printer software to authenticate the cartridge so it would work in Lexmark printers.[138] The court decided that "[t]o the extent compatibility requires that a particular code sequence be included in the component device to permit its use, the merger and scenes a faire doctrines generally preclude the code sequence from obtaining copyright protection."[139]

The Eleventh Circuit rendered a similar ruling in *Bateman v. Mnemonics, Inc.*[140] After Bateman stopped licensing the operating system on which Mnemonics had run its automated parking garage program, Mnemonics developed its own compatible operating system that reimplemented Bateman's interface.[141] The Eleventh Circuit concluded that even literal code may be filtered out under the abstraction, filtration, and comparison test pioneered in *Altai*.[142] It faulted the District Court for failing to instruct the jury "that compatibility . . . is a consideration that applies at the literal level."[143] Although the court declined to hold that interface specifications are wholly outside the scope of copyright,[144] it nonetheless concluded that "external considerations such as compatibility may negate a finding of infringement."[145] Where literal copying is "dictated by compatibility requirements,"[146] copyright does not apply.

These decisions affirm the conclusions of the National Commission on New Technological Uses of Copyrighted Works, whose report Congress

[134] *Id.* at 839–40.
[135] *Id.* at 840.
[136] 387 F.3d 522 (6th Cir. 2004).
[137] *Id.* at 529–30.
[138] *Id.*
[139] *Id.* at 536.
[140] 79 F.3d 1532, 1547 (11th Cir. 1996).
[141] *Id.* at 1536–37.
[142] *Id.* at 1545.
[143] *Id.* at 1546.
[144] *Id.* at 1547.
[145] *Id.*
[146] *Id.*

commissioned and relied upon when regulating software copyrights.[147] The report explained that "[w]hen 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."[148]

C. THE FEDERAL CIRCUIT IGNORED THE DISTRICT COURT'S FACT FINDINGS THAT SUPPORTED ITS HOLDING THAT THE INTERFACES AT ISSUE WERE UNPROTECTABLE UNDER THE MERGER DOCTRINE

As these authorities demonstrate, merger is a viable argument against copyrightability when the range of available alternatives for expressing a particular idea or method is very limited. The District Court made a finding that there *was*, in fact, only one way to write the name of each function: "Under the rules of Java, [declarations] *must be identical* to declare a method specifying the *same* functionality—even when the implementation is different."[149] Thus, any programmer wishing to invoke the functionality of "Math.max" would have to use the exact phrase "Math.max."

Its conclusion that there was only one way to write the declarations is bolstered by the amicus brief submitted by 78 computer scientists.[150] They explain that, with a very limited exception addressed below, the Java programming language requires that declarations be written in a precise form; that reuse of software interfaces such as the Java SE declarations is a foundational practice in computer science that allows programmers to write software that performs on multiple platforms at once; and, that this reimplementation requires exact duplication of an interface's declarations and organizational scheme.[151]

D. THE DISTRICT COURT PROPERLY HELD THAT NAMES AND SHORT PHRASES ARE NOT PROTECTABLE BY COPYRIGHT

The only part of the declarations not precisely dictated by the Java language are names given to specific functions.[152] But this does not bring Oracle's

[147] NAT'L COMM'N ON NEW TECH. USES OF COPYRIGHTED WORKS, FINAL REPORT (1978).

[148] *Id.* at 20 (emphasis added).

[149] Oracle Am., Inc. v. Google Inc., 872 F. Supp. 2d 974, 976 (N.D. Cal. 2012).

[150] *See* 78 Computer Scientists Cert. Brief, *supra* note 17.

[151] *Id.* at 3.

[152] *Id.* at 8–9.

interface within the scope of copyright. As the District Court concluded,[153] names are not protected by copyright law.[154]

Among the circuit courts concluding that identifiers of functional items are unprotectable by copyright law is *Southco, Inc. v. Kanebridge Corp.*,[155] in which the Third Circuit considered whether the serial numbers used to uniquely identify hardware parts were copyrightable; it decided that they were not.[156] The court explained that part numbers are "excluded from copyright protection because they are analogous to short phrases or the titles of works."[157]

The Sixth Circuit has rendered similar rulings. In *ATC Distribution Group v. Whatever It Takes Transmissions & Parts, Inc.*,[158] the court held that a taxonomy for assigning unique identifiers to auto transmission parts by sorting them into categories and sub-categories was not copyrightable.[159] The taxonomy for assigning numbers was itself an uncopyrightable idea,[160] and the numbers generated through application of the taxonomy were unprotected because they were unoriginal or else merger had occurred.[161] Beyond this, the court concluded that there were additional reasons not to grant copyright protection "to short 'works,' such as part numbers."[162] It recognized that allowing copyright in such short works would substantially raise the risk of litigation for those who use such works legitimately and would not meaningfully advance the progress of science and useful arts.[163] Accordingly, U.S. Copyright Office

[153] Oracle Am., Inc. v. Google Inc., 810 F. Supp. 2d 1002, 1009 (N.D. Cal. 2011).

[154] Efficiency and other functional considerations constrain declaration names. *See* 78 Computer Scientists Cert. Brief, *supra* note 17, at 7–9.

[155] 390 F.3d 276 (3d Cir. 2004) (en banc) (Alito, J.).

[156] *Id.* at 277–78.

[157] *Id.* at 285. The Court also held that the serial numbers were not original expressions. *Id.* at 282.

[158] 402 F.3d 700 (6th Cir. 2005).

[159] *Id.* at 706.

[160] *Id.* at 707.

[161] *Id.*

[162] *Id.* at 709.

[163] *Id.* at 709–10; *see also* Lexmark Int'l, Inc. v. Static Control Components, Inc., 387 F.3d 522, 542 (6th Cir. 2004) (invoking the names and short phrases doctrine to reject Lexmark's claim that inserting stock ticker symbols into code was creative expression). Other circuits have denied copyright protection to names on other grounds. *See, e.g.*, Mitel, Inc. v. Iqtel, Inc., 124 F.3d 1366, 1373 (10th Cir. 1997) (holding four-digit numeric codes used to access features of telecommunications hardware were not copyrightable due to unoriginality and scènes a fàire).

procedures recognize that "[w]ords and short phrases such as *names*, titles, and slogans" are not copyright-protectable.[164]

The only parts of the declaration that are not precisely dictated by the Java programming language are names, and they too are unprotectable. Thus, no aspect of the declarations is protectable by copyright law.

IV. CONCLUSION

This Article, drawn from our brief amicus curiae in support of the Petitioner in *Google v. Oracle*, offers an analysis of the scope of software copyright law that is consistent with the most pertinent Supreme Court precedents, case law decided both before and after the 1976 Act, and the text of the § 102(b) exclusions of methods and systems from copyright's scope. The Oracle brief ignored almost all of the decisions we rely upon and ignored all but one of the § 102(b) exclusions.[165] The danger of excessively broad copyright protection for computer programs, which prompted Congress to add these exclusions to the statute, is posed by this case. As the amicus curiae briefs of 83 computer scientists, IBM Corp., Microsoft, the Developers Alliance,[166] among others, explained, the Court's decision in *Google v. Oracle* will have profound effects on the software industry. No wonder the case was called the "copyright case of the century."[167]

[164] 37 C.F.R. § 202.1(a) (emphasis added). The Copyright Office's Circular 34, *Copyright Protection Not Available for Names, Titles, or Short Phrases*, was updated to become Circular 33, *Works Not Protected by Copyright*, available at https://www.copyright.gov/circs/circ33.pdf. The update is noted in the Office's Circular Update Guide, https://www.copyright.gov/circs/circular-update-guide.pdf.

[165] Brief for Respondent, Google LLC v. Oracle Am., Inc., No. 18-956 (U.S. Feb. 12, 2020).

[166] Amicus briefs in support of petitioner Google filed by 83 Computer Scientists, International Business Machines Corp. et al., Microsoft Corp., and Developers Alliance, along with other submissions in the *Google v. Oracle* case, can be found on the U.S. Supreme Court site: https://www.supremecourt.gov/search.aspx?filename=/docket/docketfiles/html/public/18-956.html.

[167] *See, e.g.*, Scott Graham, *Supreme Court, Finally, Takes Up* 'Google v. Oracle,' LAW.COM (Nov. 15, 2019), https://www.law.com/nationallawjournal/2019/11/15/supreme-court-finally-takes-up-google-v-oracle/?slreturn=20200124134715 (quoting Professor Mark Lemley).