ORACLE AMERICA, INC. V. GOOGLE, INC.: THE ONLY NONLITERAL ASPECTS OF JAVA API S PROTECTED UNDER COPYRIGHT LAW ARE THE ONES NOBODY WANTS TO COPY

Jonathan Ambrose*

In 1976, Congress revised the Copyright Act, extending protection to computer programs. The scope of this protection, however, has remained unclear. While all courts agree that source code is protected from direct copying, they disagree as to the extent of protection for nonliteral elements, such as the code's structure, sequence, and organization. Oracle America, Inc. v. Google, Inc. is the latest case considering the issue and is consistent with a trend of decreased protection. Practically speaking, Oracle offers no protection for the nonliteral aspects of Oracle’s APIs. This weak level of protection results from computer programs’ functionality. Some might be tempted to extend the court's reasoning to justify a more explicit distinction in copyright law, denying protection to the nonliteral elements of all computer programs. However, this conclusion would be contrary both to Oracle’s holding, which extends only to the APIs considered, and to the policy that the Copyright Act is designed to effectuate.

I. INTRODUCTION

The Copyright Act of 19761 extended copyright protection to computer programs for the first time.2 Congress made it clear,
however, that protection should be limited to the expression of copyrighted works, rather than the ideas therein. Applied to computer programs, this distinction is deceptively simple; copyright protects the code written to carry out a program’s function, but does not protect the function that the program carries out. In practice, however, courts have struggled to determine where this line should be drawn, and the correct scope of protection for computer programs has been extensively debated.

programming is not a subject matter specifically enumerated as protected in the Copyright Act of 1976, it is implicitly included in the category of literary works. Computer Assocs. Int’l., Inc. v. Altai, Inc., 982 F.2d 693, 702 (2nd Cir. 1992) (“While computer programs are not specifically listed as part of the . . . statutory definition, the legislative history leaves no doubt that Congress intended them to be considered literary works.” (citations omitted)); see 17 U.S.C. § 102(a) (2006).

See 17 U.S.C. § 102(b) (“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.”); see also Baker v. Selden, 101 U.S. 99, 107 (1879) (holding that copyright protects a book’s descriptions of a bookkeeping system but cannot be used to protect the bookkeeping system itself).

See Nat’l Comm’n on New Technological Uses of Copyrighted Works, Final Report 20 (1979) (stating that copyright protects the taking of another’s program but cannot bar the creation of software that performs the same process).

Courts that have considered the issue have agreed that copyright extends beyond the source code of a program to its nonliteral elements. See, e.g., Altai, 982 F.2d at 702; Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1234 (3rd Cir. 1986); Lotus Dev. Corp. v. Paperback Software Int’l., 740 F. Supp. 37, 54 (D. Mass. 1990). However, the exact demarcation between idea and expression is unclear. Compare Whelan, 797 F.2d at 1234 (holding that nonliteral elements that are not strictly necessary to implement the overall purpose of the entire program are protectable expressions), with Altai, 982 F.2d at 702 (holding that copyright protects only nonliteral elements that are not dictated by efficiency or external factors and are not within the public domain).

See Stephen Breyer, The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs, 84 Harv. L. Rev. 281, 347 (1970). Initially, it was doubted whether any copyright protection was necessary for computer programs. See id. (“In the face of this uncertain need it would seem unwise to extend copyright protection to virtually all computer programs.
This conflict played out in the summer of 2012 in Oracle America, Inc. v. Google Inc. In developing its Android smartphone platform, Google, Inc. (“Google”) copied the function of Java application programming interfaces (“APIs”) owned by Oracle America, Inc. (“Oracle”). Specifically, the titles, hierarchy, and organization of Google’s APIs were identical to Oracle’s, while the actual source code of the programs was different. Oracle contended that by copying the “structure, sequence and organization” of its work, Google had infringed on its copyright.

The court agreed with Google, holding that the “structure, sequence and organization” of Oracle’s APIs were not protected under U.S. copyright law. Thus, Google had not infringed on Oracle’s copyright.

Some commentators suggest that copyright protection should cover only literal copies of the source and object code of computer programs. See, e.g., Dennis S. Karjala, Distinguishing Patent and Copyright Subject Matter, 35 CONN. L.R. 439, 459 (2003) (“[T]he program copyright should protect only against mechanical and nearly exact duplication . . . .”). Others suggest that copyright should protect nonliteral elements of the code, such as its structure, sequence, and organization. See Whelan, 797 F.2d at 1238 (holding that copyright protection of programs can extend beyond literal source or object code to a program’s structure, sequence, and organization).


See id. at *3. Regardless of the legality of their actions, there is no dispute that Google’s intentions in creating their own APIs were to recreate the exact functions performed by Oracle’s APIs. See id. (“In light of its inability to reach agreement with Sun, Google decided to use the Java language to design its own virtual machine via its own software and to write its own implementations for the functions in the Java API that were key to mobile devices.”); Oracle America, Inc.’s Opposition to Google’s Motion for Summary Judgment on Count VIII of Oracle’s Amended Complaint at 1, Oracle, 2012 WL 1964523 (No. 396) (“The copying in this case is undisputed.”).

See Oracle, 2012 WL 1964523, at *4 (“[T]he Android platform replicated the same package, method and class names, definitions and parameters of the 37 Java API packages . . . .”).

Id. at *1 (“Oracle’s central claim . . . was that Google had replicated the structure, sequence and organization of the overall code for the 37 API packages.”).

Id. at *4.

Id.
Imagine that the scope of this holding were extended to a novel, also a literary work under U.S. copyright law. A copy could be made that had the exact same plot, characters, chapters, title, and structure of the original. As long as the copier changed the actual sentences that made up the story, that copy would not infringe on the original novel’s copyright.

In reality, novels enjoy a much greater scope of copyright protection for these nonliteral story elements. The comparison between these two works, one artistic and one functional, illustrates the apparent absurdity of applying copyright protection to computer programs. While computer programs are “original works of authorship fixed in [a] tangible medium of expression,” they are very different from the artistic or even informative works that are more commonly associated with copyright protection. Unlike these other works, computer programs are designed to function, and generally do so, without any direct human observation.

---


14 This protection is also extended to other subject matters, such as movies, plays, and television shows, even where the infringing work is in an entirely different medium. See Horgan v. Macmillan, Inc., 789 F.2d 157, 162 (2d Cir. 1986) (comparing a book of photographs and a ballet choreography); Twentieth Century-Fox Film Corp. v. MCA, Inc., 715 F.2d 1327, 1329 (9th Cir. 1983) (comparing a movie and a television series); Sid & Marty Krofft Television Prods., Inc. v. McDonald’s Corp., 562 F.2d 1157, 1167 (9th Cir. 1977) (comparing television commercials to a television show); Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930) (comparing movies and plays).


16 See Pamela Samuelson et al., A Manifesto Concerning the Legal Protection of Computer Programs, 94 COLUM. L. REV. 2308, 2316 (1994). This concept of functionality is distinguished from the mere usefulness of a literary work. In stating that computer programs are functional, it is meant that they independently perform the function of causing a computer to carry out a certain task. See id. (describing computer programs as machines designed to make computers perform tasks); see also Dennis S. Karjala, Copyright, Computer Software, and the New Protectionism, 28 JURIME TRICS J. 33, 38 (1987) (“[Computer programs] are not designed to communicate information, thought, or feeling to human beings, nor are they designed to communicate with, as opposed to physically control, computers.”). On some level, of course, all
Oracle America, Inc. v. Google, Inc.

This Recent Development argues that, while the holding of Oracle offers relatively weak copyright protection for Oracle’s APIs, this is both an inevitable result of computer programs’ functional nature and the correct level of protection, given the current state of the computer programming industry. Furthermore, while the narrowness of Oracle’s holding may limit its application in future copyright disputes, it allows for flexibility in the application of copyright protection in the inevitable event of changes to that industry.

Part II describes the origins of copyright protection for computer programs and their incorporation into the Copyright Act. Part III explains the relevant technology and outlines how these aspects affect the analysis of copyright protection for computer programs. Part IV discusses the rationale of the Oracle and how the functionality of APIs, and concerns about their interoperability, shaped the court’s holding. Part V argues that, given the current state of the computer software industry, Oracle’s weak level of protection strikes the correct balance between fostering the efficient exchange of ideas and allowing innovators to profit from their ingenuity. Finally, Part VI considers Oracle’s place in the continuing development of copyright protection for computer programs and its ultimate fate on appeal.

II. HISTORY OF COPYRIGHT PROTECTION OF COMPUTER PROGRAMS

Baker v. Selden\(^1\) is the origin of the fundamental concepts


literary works have some “function,” for example to entertain, teach, or inform. However, these functions are generally inseparable from the act of reading or observing the copyrighted work. Where the function or purpose of other literary works is independent of communication to a reader, that function is likewise denied copyright protection. For example, copyright does not protect a system of bookkeeping described in a book, Baker v. Selden, 101 U.S. 99, 107 (1879), or a recipe in a cookbook, Publ’ns Int’l., Ltd. v. Meredith Corp., 88 F.3d 473, 482 (7th Cir. 1996).


\(^1\) 101 U.S. 99 (1879).
governing the scope of copyright protection for computer programs. Although the case predates the widespread use of computer programs by over a century, the principles described still provide courts with guidance in applying copyright law to computer programs. In amending the Copyright Act in 1976 and 1980, Congress made it clear that these principles would continue to govern the scope of copyright law and would be particularly applicable to computer software.

A. Idea/Expression Dichotomy

The dispute in *Baker* centered on the copyright for a book describing, and containing forms for, a novel system of bookkeeping, created by Selden. Baker created a book that used the same system and similar forms. Selden argued that, by copying the system and the forms contained in his book, Baker infringed upon his copyright.

The Court distinguished between Selden’s book and the system it described and implemented. It determined that Selden’s copyright extended only to the book as a description of the system and instructions on its use and could not be used to copyright the

---

19 *Id.* at 107; *see* Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 704 (2nd Cir. 1992) (identifying *Baker* as the starting point in the analysis of a utilitarian work).


21 *See* 17 U.S.C. §§ 101–102 (2006) (“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.”); H.R. REP. NO. 94-1476, at 57 (1976) (“[T]he basic dichotomy between expression and idea remains unchanged.”).

22 *Baker*, 101 U.S. at 100.

23 *Id.*

24 *Id.*

25 *Id.* at 101 (framing the issue as whether the copyright gave the owner an exclusive right to the system described or merely to the book’s description).
system of bookkeeping itself. This distinction has since become an unquestioned factor in determining the scope of copyright law; copyright does not protect the idea, function, or system of a work but only the manner in which that idea is expressed. 

The Court went on to state that even the expression of a work is not covered under copyright law where it is necessary to implement the idea of the work. Where a particular expression of an idea is the only way to effectuate that idea, that expression “merges” with the idea and is not copyrightable. This concept has subsequently been termed the “merger doctrine.”

These two concepts form the basic analysis used to determine what aspects of a computer program are afforded copyright protection. A computer program’s system, purpose, or function can never be protected by copyright, but the specific source or object code designed to implement those purposes may be protected by copyright. However, where the program’s function cannot be

---

26 Id. at 105 (“The description of the art in a book, though entitled to the benefit of copyright, lays no foundation for an exclusive claim to the art itself.”).
27 See Computer Assocs. Int’l., Inc. v. Altai, Inc., 982 F.2d 693, 703 (2nd Cir. 1992) (citing Baker, 101 U.S. 99) (stating that distinction between idea and expression is a fundamental principle of copyright law); Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1234 (3rd Cir. 1986) (“It is axiomatic that copyright does not protect ideas, but only the expressions of ideas.”); see also Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930) (discussing the difficulty of drawing the line between idea and expression).
28 Baker, 101 U.S. at 103 (stating that copyright does not protect methods and diagrams in a mathematical work that are necessary to explain or use that mathematical concept).
29 Id. (“Such methods and diagrams are to be considered as necessary incidents to the art, and given therewith to the public . . . .”).
31 See 17 U.S.C. § 102(b) (2006); H.R. REP. No. 94-1476, at 56–57 (1976) (explaining that copyright does not protect the processes or methods of the computer program).
32 See Oracle, 2012 WL 1964523, at *13 (“Everyone agrees that no one can copy line-for-line someone else’s copyrighted computer program.”).
carried out except by using the same source code, that code cannot be protected by copyright.\textsuperscript{33}

B. The Copyright Act of 1976 and CONTU

In 1976, Congress extended and revised the Copyright Act, officially extending copyright protection to computer programs.\textsuperscript{34} In doing so, however, Congress made it clear that the distinction between idea and expression outlined in \textit{Baker} would continue to govern copyright disputes.\textsuperscript{35} In particular, this distinction should be used to determine the scope of copyright protection for computer programs.\textsuperscript{36}

In 1980, Congress created the National Commission on New Technological Uses of Copyrighted Works (“CONTU”) in order to reevaluate computer programs’ status under copyright law.\textsuperscript{37} The only amendment recommended by CONTU was the inclusion of a definition of computer programs into the act.\textsuperscript{38} The scope of copyright protection for computer programs, however, should

\textsuperscript{33} \textit{See} Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 705 (2nd Cir. 1992) (“[T]hose elements of a computer program that are necessarily incidental to its function are similarly unprotectable.”).

\textsuperscript{34} \textit{See id.} at 702 (“While computer programs are not specifically listed as part of the . . . statutory definition, the legislative history leaves no doubt that Congress intended them to be considered literary works.” (citations omitted)); \textit{see also} 17 U.S.C. § 102(a).

\textsuperscript{35} \textit{See 17 U.S.C. § 102(b)} (“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.”); \textit{H.R. REP. NO. 94-1476, at 57} (“[T]he basic dichotomy between expression and idea remains unchanged.”).

\textsuperscript{36} \textit{See H.R. REP. NO. 94-1476, at 56–57} (explaining that copyright protects the expression adopted by the programmer but not the processes or methods of the program).

\textsuperscript{37} \textit{See Oracle}, 2012 WL 1964523, at *12 (“Congress established [CONTU] to recommend the extent of copyright protection for software.”).

\textsuperscript{38} \textit{See NAT’L COMM’N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, supra} note 4, at 16 (recommending that the Copyright Act be amended to more specifically identify computer programs as a protectable subject matter); \textit{see also} 17 U.S.C. § 101 (2006).
remain the same and should continue to be determined based on the Baker-like principles of 17 U.S.C. § 102(b).  

III. TECHNOLOGY AND STRUCTURE OF APIS

The basic proposition of copyright law is that the purpose or function of a computer program is ineligible for copyright, while the code that expresses or carries out that function may be protected. In practice, however, the specific structure of computer programs makes this standard difficult to apply. The ordered, hierarchical structures of computer programs make it difficult to determine at what level of abstraction the line should be drawn between the function of a program and the code that expresses that function.

A. Computer Programs in General

At the most general level, computer programs can be defined as the overall purpose that they are designed to carry out. This purpose, what the computer program is designed to ultimately do, is the first step in the programming process. It is universally agreed that, at this level, the computer program is not protected;

39 See Nat’l Comm’n on New Technological Uses of Copyrighted Works, supra note 4, at 20 (reconciling proposed definition of computer programs with the limits contained in 17 U.S.C. § 102(b)).
40 See Oracle, 2012 WL 1964523, at *19–20 (stating that copyright cannot protect the function of the program, but it does protect the specific way that the code is written).
42 See Altai, 982 F.2d at 706 (describing the process of abstraction through which courts should evaluate functions of tasks within the program to determine whether these functions are expressions protected by copyright); Englund, supra note 41, at 897–98.
43 See Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1240 (3rd Cir. 1986) (characterizing the idea of the software at issue as the efficient management of a dental practice).
44 Englund, supra note 41, at 870 (“A programmer begins the process of writing a program by precisely defining the program’s ‘function.’ ”).
the creator of a computer program can never protect the ultimate purpose or function of his program using copyright.45

In order to implement these general purposes, which are sometimes very complex, a programmer uses a collection of subtasks, which, performed on aggregate, enact the program’s general purpose.46 These subtasks can in turn be subdivided into smaller tasks, and so on, until a hierarchy is created whereby a collection of simple, independent tasks interact with each other to produce the desired function.47 These simpler tasks are then written out in a programming language, along with instructions for how the individual modules should behave and interact.48 At this level of abstraction, the literal source code of a program, a program is eligible for copyright protection.49 The actual line-by-line code of a program cannot be copied, unless that expression is the only existing method to implement the program’s purpose.50

While courts do not differ on the principles used to determine the boundary between idea and expression in a computer program, they are inconsistent in determining the correct level of abstraction at which to make this distinction.51 The overall function of a

45 See Whelan, 797 F.2d at 1240 (holding that only the overall idea of the software, the efficient running of a dental practice, was unprotected by copyright); Oracle, 2012 WL 1964523, at *2 (stating that anyone is free to write code with the same function or purpose as a copyrighted work).

46 Englund, supra note 41, at 870.

47 Id. at 871 (“The functions of the modules in a program together with each module’s relationships to other modules constitute the ‘structure’ of the program.”).

48 Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 698 (2nd Cir. 1992) (“Once each necessary module has been identified, designed, and its relationship to the other modules has been laid out conceptually, the resulting program structure must be embodied in a written language that the computer can read.”).

49 See Lotus Dev. Corp. v. Borland Int’l, Inc., 49 F.3d 807, 820 (1st Cir. 1995) (stating that the actual source code of a program appears to be within the reach of copyright protection), aff’d, 516 U.S. 233 (1996).

50 See Oracle, 2012 WL 1964523, at *13 (“Everyone agrees that no one can copy line-for-line someone else’s copyrighted computer program.”).

51 Compare Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222 (3rd Cir. 1986) (holding that nonliteral elements that are not strictly necessary to implement the overall purpose of the entire program are protectable
computer program is dependent on a collection of underlying simpler functions, which can generally be carried out using subroutines and operations arranged quite differently from the one chosen by the programmer. Thus, it is an open question to what degree the structure, sequence, and organization of a program can be copyrighted, when that structure is not necessary to carry out the program’s purpose.

B. APIs and Interoperability

APIs, like other computer programs, are designed using the general structure described above. Functionally, however, APIs are designed as an interface between independently designed computer programs. These interfaces contain protocols and instructions for sharing information and functions between programs. For example, almost all applications on a computer require APIs in order to communicate with the operating system for basic functions, such as accessing files. APIs are also used between

expressions), with Altai, 982 F.2d at 720–21 (holding that copyright protects only nonliteral elements that are not dictated by efficiency or external factors and are not within the public domain).

52 Englund, supra note 41, at 871 (“The functions of the modules in a program together with each module’s relationships to other modules constitute the ‘structure’ of the program.”).

53 See Whelan, 797 F.2d at 1238 (pointing out that the existence of similar programs to the one that was alleged to be copied that had the same purpose but different structure).

54 See Oracle, 2012 WL 1964523, at *19 (stating that the structure, sequence, and organization of an API may be copyrightable, as long as it represents an expression, rather than an idea); see also Altai, 982 F.2d at 702 (stating that nonliteral elements of computer programs are protected under copyright law); Whelan, 797 F.2d at 1248 (holding that the structure, sequence, and organization of dental scheduling software was protected expression).

55 Dave Roos, How to Leverage an API for Conferencing, HOW STUFF WORKS, http://money.howstuffworks.com/business-communications/how-to-leverage-an-api-for-conferencing.htm (last visited Nov. 18, 2012) (“With APIs, applications talk to each other without any user knowledge or intervention.”).

56 Id.

57 David Orenstein, Quickstudy: Application Programming Interface (API), COMPUTER WORLD (Jan. 10, 2000, 12:00 PM), http://www.computerworld.com/s/article/43487/Application_Programming_Interface.
programs that interact remotely, such as applications that are
generated or displayed on a web browser.\textsuperscript{58}

APIs’ function, allowing other programs to properly interact, creates novel considerations for the purposes of copyright law. The structure of APIs, while not strictly necessary to implement the individual purposes of individual programs, does affect the functionality of the API.\textsuperscript{59} APIs designed with different structures can lead to “fragmentation.”\textsuperscript{60} Thus, if multiple programmers creating applications wish for their individual programs to interact seamlessly, it may be desirable to use APIs that have the same structure, even where another structure would function.

\textbf{IV. ORACLE AMERICA, INC. V. GOOGLE, INC.}

The dispute at the center of Oracle arose from the creation by Google of APIs in the Java programming language in order to implement their Android smartphone platform.\textsuperscript{61} When Google’s attempts to negotiate a license for Oracle’s APIs for use with the Android platform fell through, Google developed its own APIs modeled after Oracle’s.\textsuperscript{62}

The 37 Google APIs at issue had exactly the same structure as Oracle’s, including the order and the titles (referred to as “declarations” or “headers”).\textsuperscript{63} Only the source code used to implement these different subroutines was different.\textsuperscript{64} Thus, in order for Oracle to triumph in their copyright infringement claim, it was necessary for the sequence, structure, and organization of Oracle’s APIs to be protected by copyright.

\begin{itemize}
  \item \textsuperscript{58} Roos, supra note 55.
  \item \textsuperscript{59} See Oracle, 2012 WL 1964523, at *26 (stating that Google APIs needed to have the same structure as Oracle APIs in order to interface with software created using Oracle APIs).
  \item \textsuperscript{60} Id.
  \item \textsuperscript{61} Id. at *3–4.
  \item \textsuperscript{62} Id. at *4.
  \item \textsuperscript{63} Id. at *3–4.
  \item \textsuperscript{64} Id. at *4.
\end{itemize}
A. Copyrighting the Structure, Sequence, and Organizations of Computer Programs

In arguing that the structure, sequence, and organization of APIs were eligible for copyright protection, Oracle relied primarily on *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.* In *Whelan*, the Third Circuit considered a copyright claim based on scheduling software designed for a dentist’s office. Jaslow designed software that used the same system, but was written in a different code for dental practices using different computers.

The court held that, by copying the sequence, structure, and organization of the original program, Jaslow had infringed copyright. The court applied the idea/dichotomy distinction outlined in *Baker* but defined the idea of the software in the broadest possible terms: The idea in this case was the overall purpose of Whelan’s software, and its detailed structure was one possible expression of that idea. Because Jaslow’s software could have performed this same overall goal while using a variety of different organizations, the structure of Whelan’s software was protected by copyright.

*Whelan* has been widely criticized as overly broad. The court began with the faulty premise that computer programs are comprised solely of one underlying idea and that all other aspects

---

65 797 F.2d 1222 (3rd Cir. 1986).
66 Id. at 1226.
67 Id. at 1222 (affirming the district court’s holding that the Dentalab copyright was infringed because the overall structure of Dentcom was substantially similar).
68 Id. at 1240.
69 Id.
of the program were merely expressions of that idea.\textsuperscript{71} However, a computer program is more properly thought of as a collection of smaller programs, each of which have their own underlying idea.\textsuperscript{72} Accordingly, courts have declined to follow Whelan’s broad rule.\textsuperscript{73}

When confronted with the issue in \textit{Computer Associates International, Inc. v. Altai, Inc.},\textsuperscript{74} the Second Circuit created a three-step test for separating protectable expression from unprotected idea.\textsuperscript{75} First, courts should determine the levels of abstraction present within the program, going from the highest level of functionality and determining each underlying level of subtasks.\textsuperscript{76} Having identified each level and system within the program, the court should next proceed with a process of filtering out unprotected elements.\textsuperscript{77} This step should eliminate those elements of the program that are dictated by efficiency or external factors, as well as those within the public domain.\textsuperscript{78} Finally, the court should compare the remaining elements of the programs to determine if these protected elements have been infringed.\textsuperscript{79}

The rule the court ultimately used in \textit{Oracle} is most similar to the one applied in \textit{Lotus Development Corp. v. Borland International, Inc.}.\textsuperscript{80} In \textit{Lotus}, the First Circuit held that the command structure of a spreadsheet program was a method or system unprotected by copyright.\textsuperscript{81} This conclusion was not based on whether the structure was necessary to implement a spreadsheet

\textsuperscript{71}\textit{Whelan}, 797 F.2d at 1236 ("[T]he 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.").

\textsuperscript{72}See \textit{Altai}, 982 F.2d at 705 ("[A] computer program’s ultimate function or purpose is the composite result of interacting subroutines.").


\textsuperscript{74}982 F.2d 693 (2nd Cir. 1992).

\textsuperscript{75} Id. at 706.

\textsuperscript{76} Id. at 707.

\textsuperscript{77} Id.

\textsuperscript{78} Id. at 708.

\textsuperscript{79} Id. at 710.

\textsuperscript{80}49 F.3d 807 (1st Cir. 1995), \textit{aff’d}, 516 U.S. 233 (1996).

\textsuperscript{81} Id. at 815.
program or whether the structure employed was the most efficient.\footnote{Id. at 816 ("The fact that Lotus developers could have designed the Lotus menu command hierarchy differently is immaterial to the question of whether it is a ‘method of operation.’").} Rather, the court determined that, because the structure of the program is the method by which the program performs its function, that method is an idea unprotected by copyright.\footnote{Id.} This is true regardless of whether the structure could be changed significantly while still retaining the overall function of the program.\footnote{Id.}

\textit{Lotus} was affirmed by an evenly divided Supreme Court,\footnote{See Lotus Dev. Corp. v. Borland Int’l, Inc., 516 U.S. 233 (1996), aff’d, 516 U.S. 233 (1996).} and the Court announced no rule or guiding principle for determining the extent of copyright protection for the structure, sequence, and organization of a computer program.\footnote{See id. Another possible reason for the ubiquity of the \textit{Baker} analysis in computer programming cases, other than its fundamental character, is that it is arguably the closest thing to binding Supreme Court precedent that exists on the question of copyright protection for computer programs. \textit{See} Baker v. Selden, 101 U.S. 99 (1879); \textit{see also} Feist Publ’ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340 (1991) (considering the alleged infringement of a phone book).} Decisions have thus proceeded using an inconsistent, ad hoc approach.\footnote{See Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 705 (2nd Cir. 1992) (“While \textit{Baker} v. \textit{Selden} provides a sound analytical foundation, it offers scant guidance on how to separate idea or process from expression, and moreover, on how to further distinguish protectable expression from that expression which ‘must necessarily be used as incident to’ the work’s underlying concept.”).}

\textbf{B. Weak Copyright Protection of Java APIs as a Result of Their Functionality}

The court in \textit{Oracle} held that the structure, sequence, and organization of Oracle’s APIs were not protected.\footnote{Oracle Am., Inc. v. Google Inc., No. C 10-03561 WHA, 2012 WL 1964523, at *28 (N.D. Cal. May 31, 2012).} Furthermore, the small amount of source code in Google’s APIs that were
identical to Oracle’s89 (mostly the declarations at the beginning of the different sections) were necessary in order for the code to function as intended and were thus excluded from copyright protection by the “merger” doctrine.90

The reasoning of the court’s decision in Oracle hinged on the functionality of the aspects alleged to be copyrighted.91 U.S. copyright law protects expression rather than ideas.92 Because the sequence, structure, and organization of Oracle’s APIs were related to their function, they represented an unprotected method of operation, rather than expression.93 The Oracle court stated:

That a system or method of operation has thousands of commands arranged in a creative taxonomy does not change its character as a method of operation. Yes, it is creative. Yes, it is original. Yes, it resembles a taxonomy. But it is nevertheless a command structure, a system or method of operation—a long hierarchy of over six thousand commands to carry out pre-assigned functions. For that reason, it cannot receive copyright protection—patent protection perhaps—but not copyright protection.94

Under Oracle’s reasoning, whether an aspect of a computer

89 Id. at *8. In addition to the structure, sequence, and organization of these thirty-seven APIs, “Oracle . . . made much of nine lines of code that crept into both Android and Java.” Id. Copyright claims for this code were not discussed in the order and are beyond the scope of this Recent Development. Id.

90 Id. at *12, *25 (“[W]hen there is only one way to write something, the merger doctrine bars anyone from claiming exclusive copyright ownership of that expression. Therefore, there can be no copyright violation in using the identical declarations.”).

91 Id. at *25–26 (concluding that, because the APIs’ structure was a command system designed to carry out certain functions, it was a method of operation unprotected by copyright).

92 See H.R. REP. NO. 94-1476, at 56, 57 (1976) (“[T]he expression adopted by the programmer is the copyrightable element in a computer program, and . . . the actual processes or methods embodied in the program are not within the scope of the copyright law.”); see also 17 U.S.C. § 102(b) (2006) (“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.”).


94 Id. at *26 (emphasis added).
program is protected depends on its relation to the program’s function. As in *Lotus*, this analysis does not depend on whether the command structure used is the only arrangement that can be used to perform the same function or whether it is the most efficient structure. A program’s sequence, structure, and organization are therefore only surely protected by copyright if they are unrelated to the function of the program.

Computer programs, however, have negligible value outside of their functionality. Unlike other literary or even informative works, it is not necessary for a user to see the source code of a program or its structure for it to perform its desired function. For the average user, a copy that performs exactly the same function and has the same interoperability of the original program is equally as valuable as the original. Thus, copyright only protects those aspects of computer programs that are inefficient, arbitrary, or meaningless. In other words, copyright protects the nonliteral...

---

95 *Lotus Dev. Corp. v. Borland Int’l*, Inc., 49 F.3d 807, 816 (1st Cir. 1995) (“The fact that Lotus developers could have designed the Lotus menu command hierarchy differently is immaterial to the question of whether it is a ‘method of operation.’”), *aff’d*, 516 U.S. 233 (1996); *Oracle*, 2012 WL 1964523, at *26 (holding that Oracle APIs structure was a command system, which cannot be copyrighted even if the exact structure is not necessary to create a program with the same function).

96 *Oracle*, 2012 WL 1964523, at *26 (reasoning that because the structure of Oracle’s APIs were the method by which they carried out pre-assigned functions, they were ineligible for copyright protection).


98 See Karjala, *supra* note 16, at 38 (“[Computer programs] are not designed to communicate information, thought, or feeling to human beings, nor are they designed to communicate with, as opposed to physically control, computers.”).

99 See Phillips, *supra* note 97, at 1009 (“In general, consumers purchase software for its utilitarian value in accomplishing certain functions.”).

100 See Dennis S. Karjala, *Protecting Innovation in Computer Software, Biotechnology, and Nanotechnology*, 16 VA. J.L. & TECH. 42, 52 (2011) (“Nonliteral elements of program code, such as SSO, are not protected if they are designed with an eye on efficiency.” (citation omitted)).
aspects of computer programs that are least likely to be copied.\textsuperscript{101}

As illustrated above, and by the court’s reasoning, it is this functional distinction between computer programs and other works that make them particularly unamenable to broad copyright protection.\textsuperscript{102} This distinction between functional and nonfunctional works is the basis for the broad division of intellectual property law between copyright and patent.\textsuperscript{103}

The distinction between functional and nonfunctional works has also been implicated as the basis for the differing scopes of protection and admission between these two intellectual property regimes.\textsuperscript{104} Because functional works are more likely to be

\textsuperscript{101} Id.

\textsuperscript{102} See Computer Assocs. Int’l, Inc. v. Altai, Inc., 982 F.2d 693, 712 (2nd Cir. 1992) (observing that the utilitarian nature of computer programs leads to narrow copyright protection).

\textsuperscript{103} A review of the history of the idea/expression dichotomy as it is applied to utilitarian works shows that it is essentially a method by which courts determine which aspects of a work may be protected by copyright and which must be protected by patent, if at all. See Baker v. Selden, 101 U.S. 99, 102 (1879) (stating that property for the useful art described in a book can only be secured by patent); Lotus Dev. Corp. v. Borland Int’l, Inc., 49 F.3d 807, 819–20 (1st Cir. 1995) (observing that the utility of computer programs has consequences in deciding whether they should be afforded copyright protection or can only receive patent protection), aff’d, 516 U.S. 233 (1996); Altai, 982 F.2d at 712 (observing that the hybrid nature of computer programs both as literary works and functional components leads to weak copyright protection supplementing patent protection); Oracle Am., Inc. v. Google Inc., No. C 10-03561 WHA, 2012 WL 1964523, at *21 (N.D. Cal. May 31, 2012) (holding that functional elements of command structure were unprotected by copyright, and could only be protected by patent). For a more extensive discussion of how functionality determines the eligibility of subject matter for protection under either copyright or patent law, see generally Karjala, supra note 6. But see Glynn S. Lunny, Jr., Lotus v. Borland: “Copyright and Computer Programs,” 70 TUL. L. REV. 2397, 2430 (identifying the ease of copying a work as being the determining factor of whether that work should be copyrightable or protected by patent).

\textsuperscript{104} Englund, supra note 41, at 872 (stating that borrowing and improving on the ideas within a program are an important way in which computer technology progresses); Karjala, supra note 6, at 454 (“The policy basis for the radically different natures of the patent and copyright regimes is . . . the social desirability of allowing all those skilled in the relevant art to try their hands at incremental
objectively improved by incremental improvements, they should not be as strictly protected as nonfunctional works. Thus, a broad, long lasting protection for functional works is inappropriate because it would give the original innovator “the sole right to make improvements for a century, inhibiting much more improvement in the useful arts . . . than it would engender.” For artistic and informative works, the usefulness of any addition is more subjective, and thus the likelihood of a damaging monopoly is not as much of a threat.

Because of this difference in the value of incremental improvement between functional and nonfunctional works, the

---

105 Karjala, supra note 6, at 453–54. Considering the example of a vehicle, Karjala states that:

Functional works . . . are usually amenable to objectively measurable incremental improvement. Attaching a motor to a drive shaft and some wheels may be a fundamental advance in the art of transport, but, once available, that basic vehicle can be improved in an incremental series of smaller steps by the addition of brakes, headlights, a roof, windows, bumpers and other crash-protection devices, and so forth. These incremental improvements can themselves each usually be improved, again most often in incremental fashion. If the original vehicle were protected by copyright, there is a good chance that the improved vehicle . . . would infringe under copyright’s “substantial similarity” test.

Id. 106 Id. at 454.

107 See Dennis S. Karjala & Peter Menell, Applying Fundamental Copyright Principles to Lotus Development Corp. v. Borland International, Inc., 10 HIGH TECH. L.J. 177, 185 (1995) (“We prefer to have one hundred different war novels than one hundred versions of WAR AND PEACE that differ only in their final chapter.”); see also Baker, 101 U.S. at 103–04 (distinguishing between artistic designs, where the form is the essence, and mathematical and scientific tests, where the end is in the use of those concepts).
application of broad protections to computer software under U.S. copyright law conflicts with the policy that intellectual property protection is designed to encourage. Thus, the correct level of protection is one that provides the proper incentive to innovate for initial inventors, but allows for the free exchange and improvement of ideas for the benefit of consumers. To confer broad copyright protection on computer programs would allow copyright owners to “claim ownership over any and all ways to carry out methods for 95 years—without any vetting by the Copyright Office of the type required for patents.” Offering such sweeping and lengthy protections to computer programs would needlessly stifle innovation and the free exchange of ideas, inconsistent with the goals of U.S. copyright law.

C. Weak Protection as a Result of the Interoperability of APIs

Another aspect of the copied APIs that the Oracle court mentioned as important in its decisions was interoperability. The court pointed out the widespread use of Java as a programming language even before the creation of Android. In order for these other programs (which use the Oracle APIs) to run on Android, Google needed to retain the overall structure of the thirty-seven

---

108 See Feist, 499 U.S. at 349 (“The primary objective of copyright is not to reward the labor of authors, but [t]o promote the Progress of Science and useful Arts.” (internal quotations omitted)).

109 See id.; Dennis S. Karjala, A Coherent Theory for the Copyright Protection of Computer Software and Recent Judicial Interpretations, 66 U. Cin. L. Rev. 53, 57 (1997) (“The purpose of [copyright protection] is to draw a balance between, on the one hand, providing an incentive for the creation of works desired by society and recognizing in some fair and just way the efforts of their creators, and on the other hand, ensuring a broad public domain that permits later inventors and authors to build on the existing foundation to advance technology and culture for the overall benefit of society.”).


111 Id. at *26 (“Interoperability sheds further light on the character of the command structure as a system or method of operation.”).

112 Id. (“[M]illions of lines of code had been written in Java before Android arrived.”).
APIs. That the structure of Google’s APIs was necessary in order for them to properly interface with existing and future software provided further evidence that that structure was a system or method of operation under 17 U.S.C. § 102(b).

To support this assertion the court cited two Ninth Circuit cases that dealt with interoperability. In both Sega Enterprises Ltd. v. Accolade, Inc., and Sony Computer Entertainment, Inc. v. Connectix Corp., the court considered the reverse engineering of game hardware in order to determine its functional elements. In Sega, a game manufacturer did so to determine how to manufacture games on Sega’s game console. In Sony, the purpose was to design an emulator that would play games designed for Sony’s system. In both cases it was determined that, because the copied elements were interface procedures required for interoperability, they were “functional requirements for compatibility” and were not copyrightable under 17 U.S.C. § 102(b).

While the court in Oracle found interoperability to be persuasive in determining that the structure of Oracle’s APIs was a method or system, it was hesitant about using interoperability alone as a basis for this finding. Doing so, it reasoned, might have implications for the licensing of Java APIs. Thus, interoperability concerns were not dispositive in the court’s

---

113 See Google’s Trial Brief at 12, Oracle, 2012 WL 1964523 (No. 534) (“The API packages and their organization are functional elements of the Java language that are essential for interoperability.”).

114 Oracle, 2012 WL 1964523, at *27.

115 977 F.2d 1510 (9th Cir. 1992).

116 203 F.3d 596 (9th Cir. 2000).

117 Id. at 603–05; Sega, 977 F.2d at 1533.

118 Sega, 977 F.2d at 1515.

119 Sony, 203 F.3d at 601.


121 Id.

122 Id. at *26 (“While fragmentation is a legitimate business consideration, it begs the question whether or not a license was required in the first place to replicate some or all of the command structure.”).
decision that the APIs’ structure, sequence, and organization were ineligible for copyright protection, but those concerns were a factor leading to this conclusion.123

V. SOME MINIMAL, FLEXIBLE COPYRIGHT PROTECTION IS NECESSARY

The proposition that the functional nature of computer programs should so severely restrict their protection under copyright leads to the obvious question of why copyright should protect computer programs at all. Assuming that the broad division between copyright and patent law is largely determined by the functionality of the work in question, the obvious solution would be to eliminate copyright protection for computer programs and instead rely solely on software patents.

A. Need for Copyright Protection

Two reasons exist for copyright protection as a method for protecting computer programs from infringement.124 First, the ease and utility with which computer code can be copied necessitate that direct, mechanical copying be protected by copyright law. Compared to mechanical inventions, computer programs are quick, easy, and inexpensive to copy.125 Even unpatented mechanical inventions enjoy a substantial lead-time due to the inherent inertia in discovering the original and building copies.126 For computer programs, the ease of copying reduces this lead-time “essentially to zero.”127 If inventors cannot profit from their inventions, they have less incentive to create programs for the public to use. Thus, this ease of copying suggests a need for some level of protection, if

123 Id.
124 A third reason, an initial unwillingness to grant patents for software, is largely historical and beyond the scope of this Recent Development. See Karjala, supra note 6, at 442 (“Patent’s initial foot-dragging . . . allowed copyright to establish a firm beachhead in this crucial field of technology.” (citations omitted)).
125 Id. at 458.
126 Id.
127 Id.
“only against mechanical and nearly exact duplication.”128

Additionally, although the functional nature of computer programs seems to suggest patent as the correct method of protection, certain aspects of computer programs make them less amenable to patent protection. Because even expensive programs will be “the result of direct application of computer science principles to solve a well-defined problem,” they may be ineligible for patents.129 Also, while copyright protection has given way to an increase in the availability and use of software patents,130 these patents are “generally more useful for defensive, rather than for offensive, purposes.”131 This means that seekers of patents generally procure them in order to allow security in creating new software or for creating portfolios of assets for trade or sale.132 Patents are less useful for offensive purposes to combat infringement “because such elements are typically difficult to discern in commercially distributed object code.”133

However, even though patent protection of computer programs has practical limitations and the cost of copying is minimal, these reasons alone are insufficient justification for extending copyright protection to computer programs.134 As stated above, the assumed goal of intellectual property protection is to provide incentive to innovate for initial inventors while allowing unrestricted exchange and improvement of ideas.135 Thus, even if programs are easy to

---

128 Id. at 459.
129 Karjala, supra note 100, at 50.
130 See Diamond v. Diehr, 450 U.S. 175, 191–93 (1981) (holding that while a mathematical algorithm was not eligible for patent protection, its implementation for a specific use is protected); In re Alappat, 33 F.3d 1526, 1543 (Fed. Cir. 1994) (en banc) (extending Diamond’s holding to software running on a general purpose computer).
132 Id.
133 Id.
134 See Breyer, supra note 6, at 344 (“[A] large difference between the cost of producing a work initially and the cost of copying is not alone sufficient to show that copyright protection is desirable.”).
135 See supra note 109 and accompanying text.
copy, this should only lead to copyright protection if its absence would mean programmers had no incentive to create.\textsuperscript{136} A better assessment of the proper scope of copyright protection, therefore, is “whether program developers are able to recoup the costs of development in a meaningful way . . . .”\textsuperscript{137} If software companies are able to get returns on their investments, then they have incentive to create new software.\textsuperscript{138}

A survey of the state of the computer software industry under current copyright protection seems to confirm that the level of protection conferred meets these goals. The computer software industry has seen considerable growth in the twenty-first century.\textsuperscript{139} Given the incredible growth and profitability of computer programs, it is difficult to argue, at least at this point in time, that the current negligible level of copyright protection is inappropriate.

B. \textit{Changes Require Varying Levels of Copyright Protection}

While Oracle’s reasoning might be extended to limit the copyright protection of nonliteral aspects of other computer programs, the court was careful to narrow its holding to the facts presented:

\begin{footnotesize}
\begin{itemize}
\item[\textsuperscript{136}] See Breyer, \textit{supra} note 6, at 282 (stating that copyright law is a way of assuring enough revenue to assure adequate production).
\item[\textsuperscript{137}] Samuelson, \textit{supra} note 131, at 1747.
\item[\textsuperscript{138}] See \textit{supra} note 136 and accompanying text.
\item[\textsuperscript{139}] Samuelson, \textit{supra} note 131, at 1775–76 (“According to one industry report, there were 6918 software-development firms in the United States in 2010 that were expected to generate more than $150 billion in revenues that year. The Business Software Alliance (“BSA”), a trade industry group, has painted an even rosier picture of the U.S. Software industry. BSA reports that software and related service industries employed 1.7 million people in 2007 and contributed more than $261 billion to the U.S. gross domestic product that year. BSA also reported that in 2008, the U.S. software firms generated $136.6 billion in revenues globally from the sale of packaged software, representing a 45.9% share of the market.”).
\end{itemize}
\end{footnotesize}
This order does not hold that Java API packages are free for all to use without license. It does not hold that the structure, sequence and organization of all computer programs may be stolen. Rather, it holds on the specific facts of this case, the particular elements replicated by Google were free for all to use under the Copyright Act.140

It is difficult to imagine, given the reasoning used in the case, under what circumstances the structure, sequence, and organization of a computer program would be protected, unless it was arranged in such an arbitrary fashion as to have no relation to the function of the program.141 Furthermore, the narrowness of the holding gives little guidance to future courts faced with similar disputes or owners of software copyrights trying to determine if their copyrights have been infringed.

While the narrowness of Oracle’s holding may seem contrary to its reasoning or invite future uncertainty, it is reflective of the volatility of the computer software industry. Although a thin protection is necessary at this point in time, changes in the technology, infrastructure, and practices of the industry may call for greater protection, or even less; indeed, since copyright began to be used to protect computer programs, changes to the industry have called for varying levels of protection.142 Thus, Oracle’s narrow ruling on the facts of the case allows for the necessary flexibility in applying copyright protection to software in future disputes.

That the principles of copyright protection and their scope should be applied flexibly for computer programs is also supported by an examination of Congress’s amendment of the Copyright Act in 1980. When called upon to discuss the proper scope of copyright protection for computer programs, CONTU stated that:

141 Id. at *15 (observing that Altai’s filtering of structures dictated by efficiency resulted in the copyrightability of inefficient structures).
142 See Samuelson, supra note 131, at 1747 (observing that the growth of the computer industry vindicated Congress’s decision to extend copyright protection to computer programs, but that recent developments complicate the case for copyright protection).
The many ways in which programs are now used and the new applications which advancing technology will supply may make drawing the line of demarcation more and more difficult. To attempt to establish such a line in this report written in 1978 would be futile. . . . Should a line need to be drawn to exclude certain manifestations of programs from copyright that line should be drawn on a case-by-case basis by the institution designed to make fine distinctions—the federal judiciary.

Thus, while a level of copyright protection applied narrowly on a case-by-case basis may lead to uncertainty going forward, it is ultimately the best policy for adapting to the changing needs of the software industry, and it is arguably consistent with the policy envisioned by Congress when it amended the Copyright Act in 1980 upon CONTU’s recommendation.

1. Past Changes to the Industry

A flexible regime of copyright protection helped in part to foster growth of the industry and availability of computer programs. When Congress first began to consider revising the Copyright Act over forty years ago, Justice Breyer expressed doubts that computer programs needed copyright protection. Justice Breyer determined, based on the state of the computer software industry at the time, that copyright protection did not need to be extended to computer programs.

Justice Breyer based his opinion on three aspects of computer

143 NAT’L COMM’N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, supra note 4, at 22–23.
144 Congress did not prepare any reports discussing CONTU’s report, but did amend the Copyright Act as recommended. See 17 U.S.C. § 101 (2006) (containing CONTU’s recommended definition of computer programs). This silence seems to implicitly suggest that they agreed with CONTU’s conclusions regarding the scope of copyright protection for computers.
145 Samuelson, supra note 131, at 1758 (“The availability of copyright as a form of legal protection for computer programs does not deserve all of the credit for the phenomenal growth of the software industry, but it did play a nontrivial role in the industry’s success.” (citation omitted)).
146 See supra note 2 and accompanying text.
147 Breyer, supra note 6, at 340–50.
148 Id.
software at the time. First, most software was created by hardware manufacturers and bundled with that hardware, with programs designed specifically to work with those machines.149 Second, most computer software was developed for personal use.150 Finally, programs were sold in “packages” rather than “off the shelf.”151 Because of these facts, there was little opportunity to copy programs, and even assuming they were copied, there would be little economic effect.152 In addition, Justice Breyer noted potential developments that, should they occur, would strengthen the case for the application of copyright protection to computer programs.153 In order for there to be an increased need for copyright protection, there would have to be a rise in “(1) generally usable programs, (2) produced by independent software companies, (3) selling ‘off the shelf’ (4) at low prices (5) to large numbers of widely dispersed buyers.”154

Each of these conditions came to pass.155 Driven by antitrust concerns, IBM ceased to bundle its software with hardware, and a large market developed for independently produced software.156 As this market for independent software expanded, so did the need for copyright protection.157 The increasing growth and profitability of the software industry was fostered in part by the subsequently increased copyright protection.158

149 Id. at 344–45.
150 Id. at 345.
151 Id.
152 Id.
153 Id.
154 Id. at 347.
155 For an excellent summary of the development of the computer software industry and how it fits with Breyer’s analysis, see generally Samuelson, supra note 131.
156 Id. at 1756.
157 Id.
158 Id. at 1758 (“The availability of copyright as a form of legal protection for computer programs does not deserve all of the credit for the phenomenal growth of the software industry, but it did play a nontrivial role in the industry’s success.” (citation omitted)).
2. Present and Future Changes

In addition to changes in the past that have called for the flexible application of copyright, several potential changes and current developments make it likely that the level of protection that should be afforded to computer programs will continue to change.\textsuperscript{159} A few of the more interesting developments are discussed below.\textsuperscript{160}

First, the rise of the Internet has led to an increase in software that, rather than running on individual computers, is accessed remotely.\textsuperscript{161} Because the actual code of these programs is not disseminated to users, the danger that they will be quickly copied and resold is diminished.\textsuperscript{162} Thus, the case for their protection under copyright is very thin.\textsuperscript{163}

Second, much like software in Justice Breyer’s day was bundled with the hardware it was meant to run on, a great portion of today’s software is “embedded in hardware of all kinds (cars, toasters, cell phones, just to name a few”).\textsuperscript{164} The sales in these cases are likely driven by demand for the hardware itself, rather than the software contained within. Thus, copyright protection is less likely to create any incentive to create this sort of software.

Third, seventy percent of investment in software development in the U.S. is currently for software developed for internal uses.\textsuperscript{165} Because these are developed specifically to solve problems for a certain company rather than being distributed or sold, they are less

\begin{itemize}
\item \textsuperscript{159} \textit{Id.} at 1776.
\item \textsuperscript{160} For a more in depth discussion of these developments, and some not listed here, see generally \textit{id.} at 1774.
\item \textsuperscript{161} \textit{id.} at 1776.
\item \textsuperscript{162} \textit{Id.} at 1778–79; see also Horacio E. Gutiérrez, \textit{Peering Through the Cloud: The Future of Intellectual Property and Computing}, 20 FED. CIRCUIT B.J. 589, 604 (2010–2011) (“Many cloud services will involve minimal distribution of software code to end users.”).
\item \textsuperscript{163} Samuelson, \textit{supra} note 131, at 1779 (“If no one but the developer of such software ever has access to a machine-executable form of the program, copyright protection is arguably unnecessary.”).
\item \textsuperscript{164} \textit{Id.} at 1776.
\item \textsuperscript{165} \textit{Id.} at 1777.
\end{itemize}
likely to need copyright protection.\textsuperscript{166}

A fourth consideration that may lead to decreased protection, as illustrated by Oracle, is interoperability. The ubiquity of computer programs creates necessity that these programs work together.\textsuperscript{167} APIs, as stated above, are programs that allow applications to interact with each other.\textsuperscript{168} Because the structure of the APIs determines how well these programs interact with each other, programmers are incentivized to design their programs with a standard structure.\textsuperscript{169} If the original creator of this standard structure (in this case Oracle) has broad copyright protection, it may lead to an undesirable monopoly, contrary to the goals of U.S. copyright law.

These developments suggest that perhaps an even thinner scope of copyright protection may be necessary in the future. While current developments suggest a narrow level of copyright protection for software, that level of protection will need to be reconsidered as technology and the industry continue to grow and evolve.\textsuperscript{170} For example, a rise in so-called “cloud” computing may result in less need for protection for the source code of these programs, which, due to the programs’ remote use, are never disseminated to users.\textsuperscript{171} This reduction in emphasis on the actual code of the programs may, in turn, lead to an emphasis on protection of nonliteral elements, such as the interface used to

\textsuperscript{166} See id. ("Nor do firms really need copyright protection for software they develop for their own internal uses, such as quality assurance software.").

\textsuperscript{167} See Orenstein, supra note 57 (stating that all applications on a computer require APIs in order to communicate with the operating system for basic functions, such as file access).

\textsuperscript{168} See supra Part III.B.


\textsuperscript{170} NAT’L COMM’N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, supra note 4, at 22–23.

\textsuperscript{171} See supra note 162 and accompanying text.
control the programs.\textsuperscript{172}

\section*{VI. The Future of Oracle America, Inc. v. Google, Inc.}

Regardless of the narrowness of Oracle, the case is representative of a progressive limiting of copyright protection for software since the high-water mark of Whelan.\textsuperscript{173} The broad protection of Whelan has been largely disavowed as courts have become more accustomed to applying copyright principles to the novel subject matter of computer programs.\textsuperscript{174} Furthermore, the emergence of patent protection as an alternative to copyright has eliminated much of the need for copyright protection.\textsuperscript{175} Given this trend, it seems unlikely that, on appeal, the nonliteral elements of Oracle’s APIs will be considered protectable under copyright law.\textsuperscript{176}

A more dangerous possibility of reversal on appeal is that an appellate court will be tempted to more clearly draw a distinction between the literal and nonliteral elements of programs, protecting the former, but not the latter. Such a bright line distinction would allow for greater clarity in applying copyright law to computer programs and would protect against wholesale copying of

\begin{footnotesize}
\begin{itemize}
\item \textsuperscript{172} See Gutiérrez, \textit{supra} note 162, at 605 (predicting that protection of user interfaces may become more important in cloud computing). The protection of user interfaces, and in particular graphic user interfaces, is an important aspect of copyright protection for computer software that is beyond the scope of this Recent Development. \textit{See, e.g.}, Apple Computer, Inc. v. Microsoft Corp., 35 F.3d 1435, 1442 (9th Cir. 1994) (“Apple argues that the ‘total concept and feel’ of its works—that is, the selection and arrangement of related images and their animation—must be compared with that of the Windows and NewWave GUIS for substantial similarity.”).
\item \textsuperscript{173} See \textit{supra} Part IV.A.
\item \textsuperscript{174} See Karjala, \textit{supra} note 100, at 43 (“After a very clumsy beginning, courts eventually settled the doctrine . . . .” (citation omitted)).
\item \textsuperscript{175} See \textit{supra} note 130 and accompanying text.
\end{itemize}
\end{footnotesize}
computer programs, which is most damaging to the industry.\textsuperscript{177} However, it is ultimately contrary to the policy of striking the correct balance in this dynamic industry.\textsuperscript{178}

In its amendments to the Copyright Act in 1976 and 1980, Congress made it clear that the idea/expression dichotomy first explicated in \textit{Baker v. Selden}\textsuperscript{179} should be applied to computer programs.\textsuperscript{180} While computer programs should be afforded only narrow copyright protection, courts can, and should, accomplish this by the correct application of these separate principles to the specific facts of the particular dispute.\textsuperscript{181}

\section*{VII. Conclusion}

The holding of \textit{Oracle America, Inc. v. Google Inc.} suggests that the protection afforded to the nonliteral aspects of computer programs is largely nonexistent.\textsuperscript{182} In addition, the court went out of its way to state that the reasoning used in the opinion should not be construed to apply broadly to all computer programs, but only the facts of the case before it.\textsuperscript{183} By offering such a narrow scope of protection to computer programs, while limiting its holding to the case before it, the court’s opinion seems unlikely to please anyone. Proponents for stricter copyright protection may point out that if copyright protects only those aspects of the computer

\begin{enumerate}
\item\textsuperscript{177} \textit{See} Karjala, \textit{supra} note 6, at 458.
\item\textsuperscript{178} \textit{See supra} Part V.B.
\item\textsuperscript{179} 101 U.S. 99, 100–01 (1879).
\item\textsuperscript{180} \textit{See supra} note 21 and accompanying text.
\item\textsuperscript{181} \textit{See Computer Associates Int’l, Inc. v. Altai, Inc.}, 982 F.2d 693, 712 (2d Cir. 1992) (stating that application of fundamental copyright concepts leads to a narrow scope of protection for computer programs). This is not to say that it would be unwise for either Congress or the Supreme Court to provide \textit{some} guidance to the courts. Ultimately, \textit{Baker v. Selden} and its codification in 17 U.S.C. § 102(b) are the only precedents that are universally binding to the district courts. \textit{See supra} note 86 and accompanying text. In order to allow courts to uniformly apply these concepts, it would be beneficial for either the Court or Congress to provide some guidance in their application.
\item\textsuperscript{183} \textit{Id.} at *28.
program that are nonfunctional, it only protects those aspects of programs that are also the least in need of protection.\textsuperscript{184} Those that wish for less copyright protection may point out that the narrow holding invites uncertainty and will lead to confusion and more costly disputes in the future.\textsuperscript{185}

In spite of these criticisms, the holding affords exactly the amount of copyright protection necessary at this time, while maintaining flexibility in the likely event of future changes. Computer programs are protected from direct, mechanical copying,\textsuperscript{186} the most damaging type of infringement and the one most likely to inhibit new innovation.\textsuperscript{187} In addition, the narrowness of the holding means that future disputes can be addressed on a case-by-case basis, allowing courts to take the inevitable changes in the industry into account when determining how much protection to afford computer programs. Thus, the level of protection can fluctuate to meet the current needs of the industry.

Copyright protection is far from ideal when applied to computer programs. Faced with a dispute, judges are forced make the hard decision of whether a computer program should be protected from some form of copying either for ninety-five years or not at all.\textsuperscript{188} A move to patent protection is one alternative, but has its own problems.\textsuperscript{189} Future events may call for a more thorough reevaluation of the intellectual property regime as applied to computer programs.\textsuperscript{190} However, in the short term, a weak but

\textsuperscript{184} See \textit{supra} note 100 and accompanying text.
\textsuperscript{185} See Karjala, \textit{supra} note 100, at 52 (“[T]he failure of the judiciary to adopt a coherent theoretical approach to program copyrights still leaves many courts chasing the will-o-the-wisp in trying to separate idea from expression in a functional work.”).
\textsuperscript{186} Karjala, \textit{supra} note 6, at 458.
\textsuperscript{187} \textit{Id.} at 458–59.
\textsuperscript{188} \textit{Oracle}, 2012 WL 1964523, at *24 (“Oracle would bypass this entire patent scheme and claim ownership over any and all ways to carry out methods for 95 years . . . .”).
\textsuperscript{189} Karjala, \textit{supra} note 6, at 458.
\textsuperscript{190} Samuelson, \textit{supra} note 131, at 1776.
flexible level of copyright protection meets the needs of the computer software industry.
Oracle America, Inc. v. Google, Inc.