NEW YORK UNIVERSITY

JOURNAL OF INTELLECTUAL PROPERTY AND ENTERTAINMENT LAW

VOLUME 8 SPRING 2019 NUMBER 2

ESSAY: WHAT REMAINS OF FAIR USE FOR SOFTWARE AFTER *ORACLE V. GOOGLE*?

SIMON J. FRANKEL AND ETHAN FORREST*

Two recent decisions from the Federal Circuit in the long-running litigation between Oracle and Google have upended the scope of copyright protection afforded to software. In both decisions, the court weighed in heavily on the side of strong copyright protection, even protecting the relatively functional code comprising application programming interfaces (APIs). In its most recent decision, the court found that Google's use in its Android software of certain APIs from Java was not fair use as a matter of law—notwithstanding a jury verdict of fair use. This essay focuses on how the Federal Circuit treated the four statutory fair use factors and suggests that the court's analysis, if applied by other courts, will make it very difficult for any use of software to qualify as a fair use. This is because, at every turn, the court's application of the fair use factors favors the copyright owner, creating copyright risk for any borrowing of copyright code in a new program. It remains to be seen if this approach will impact how software developers build on preexisting programs.

* Simon J. Frankel is a partner with Covington & Burling LLP in San Francisco and a lecturer-in-law at Stanford Law School. Ethan Forrest is an associate with Covington & Burling LLP in San Francisco. The authors are grateful to Sean Howell, an associate with Covington, and Rachel Dallal, a 2018 summer associate at Covington, for helpful assistance. The views expressed here are those of the authors only, and do not necessarily reflect the views of Covington & Burling LLP or any of its clients.

The *Oracle v. Google* case involved approximately 11,500 lines of code, two tech giants, and the birth of the now-ubiquitous Android operating system.[1] The Federal Circuit's March 2018 decision marked the culmination of two jury trials, two appeals, and years of litigation.[2] As the litigation lurches towards a conclusion—a damages trial remains, and Google is currently seeking Supreme Court review[3]—we pause to consider what the Federal Circuit's most recent decision may mean for copyright's fair use doctrine as applied to software.

For decades, courts have sought to achieve a careful balance between the copyright protection afforded to computer code and the functionality that computer code enables.[4] That is, courts have recognized that although code can reflect expressive choices, it is primarily functional and constrained, at least to some degree, by the specific purposes it is designed to achieve.[5] Consequently, courts have generally held that defendants accused of infringing software are liable only for literal copying of significant portions of underlying code.[6]

This approach to software—grounded in the primarily functional, rather than expressive, nature of most programming—has often permitted developers to build upon their predecessors' advances, at relatively minimal risk of infringement liability. Although some in Silicon Valley support this legal landscape, crediting it

---

[1] Oracle Am., Inc. v. Google L.L.C. (*Oracle IV*), 886 F.3d 1179 (Fed. Cir. 2018).

[2] *See* Oracle Am., Inc. v. Google Inc. (*Oracle II*), 750 F.3d 1339 (Fed. Cir. 2014).

[3] Petition for Writ of Certiorari, Google L.L.C. v. Oracle Am., Inc., No. 18-956 (U.S. Jan. 24, 2019).

[4] *See, e.g.*, Lexmark Int'l, Inc. v. Static Control Components, Inc., 387 F.3d 522, 535 (6th Cir. 2004) ("In ascertaining this 'elusive boundary line' between idea and expression, between process and non-functional expression, courts have looked to two other staples of copyright law—the doctrines of merger and scenes a faire.").

[5] *See, e.g.*, *id.* at 548; Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807, 815 (1st Cir. 1995), *aff'd*, 516 U.S. 233 (1996); Comput. Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 709 (2d Cir. 1992).

[6] *See, e.g.*, Apple Comput., Inc. v. Microsoft Corp., 35 F.3d 1435, 1439 (9th Cir. 1994), *cert. denied*, 513 U.S. 1184 ("When the range of protectable and unauthorized expression is narrow, the appropriate standard for illicit copying is virtual identity."); *Comput. Assocs.*, 982 F.2d at 714–15.

with enabling the tech industry's dynamism,[7] others criticize it for failing to adequately protect the creative efforts of rights-holders.[8] The recent decision in *Oracle v. Google* seems primed to address the concerns of the latter group by restricting the circumstances under which the fair use defense will protect software that incorporates parts of another program, however seemingly small or functional.

The case centered on Java, a programming platform owned by Oracle but widely used throughout the tech world. In particular, the dispute involved Java's application programming interface, or API, and some of its associated software libraries.[9] The API is the interface designed to call functions from a different piece of software, and includes a pre-programmed collection of source code packages, each designed to execute a specific function.[10] APIs' function in this context is analogous to shorthand or incorporation by reference, which allow writers to call up complex or dense ideas without re-writing them every time. APIs are integral to the software industry.[11] Many APIs let engineers implement new code atop a pre-existing framework with which other programmers are already comfortable and familiar.[12] In other words, they provide a common foundation upon which developers can build compatible products using a mutually comprehensible language.

Oracle generally encourages the incorporation of its Java APIs into new software.[13] Depending on the circumstances, the company may license such use, or even allow it for free.[14] However, Google did not obtain a commercial license to use the Java APIs in order to develop the Android operating system or comply with

---

[7] *See* Brief Amici Curiae of Am. Comm. for Interoperable Sys. & Comput. & Commc'ns Indus. Ass'n in Support of Appellant Connectix Corp. at 3, Sony Comput. Entm't, Inc. v. Connectix Corp., 203 F.3d 596 (9th Cir. 2000) (No. 99-15852), 1999 WL 33623859, at *3 (expressing concern that providing too much copyright protection for software "would render unlawful software development processes used every day in Silicon Valley").

[8] *See* Annette Hurst, *The Report of API Copyright's Death Is Greatly Exaggerated*, 31 HARV. J.L. & TECH. 491, 492–93 (2018) (arguing for a broad interpretation of when software is expressive, and thus entitled to copyright protection); Ralph Oman, *Computer Software as Copyrightable Subject Matter: Oracle v. Google, Legislative Intent, and the Scope of Rights in Digital Works*, 31 HARV. J.L. & TECH. 639, 645 (2018) (arguing that the functional nature of software code should not preclude copyright protection).

[9] *See* Oracle Am., Inc. v. Google Inc. (*Oracle II*), 750 F.3d 1339, 1348 (Fed. Cir. 2014).

[10] *See id.* at 1348–50; Oracle Am., Inc. v. Google L.L.C. (*Oracle IV*), 886 F.3d 1179, 1186-88 (Fed. Cir. 2018).

[11] Brief of Amici Curiae Comput. Scientists in Support of Petitioner at 13, Oracle Am., Inc. v. Google Inc., 750 F.3d 1339 (Fed. Cir. 2014) (No. 14-410).

[12] *See Oracle II*, 750 F.3d at 1349.

[13] *See Oracle IV*, 886 F.3d at 1187.

[14] *Id.*

Oracle's terms for a free license.[15] Confronted with the magnitude of Android's success, and harboring its own interest in preserving Java's role and relevance in the smartphone and tablet industry, Oracle sued for patent and copyright infringement.

At the initial trial in 2012, the jury split on the two claims, finding copyright infringement but no patent infringement.[16] Following trial, however, the judge delivered a complete victory for Google, ruling as a matter of law that the Java APIs were not copyrightable.[17] Oracle appealed the copyright claim to the Federal Circuit—which had jurisdiction because the original suit included a patent claim and the Federal Circuit has exclusive jurisdiction over all appeals from cases where the original jurisdiction was based, at least in part, on a patent claim.[18] In a controversial 2014 opinion, the panel held that the Java APIs were sufficiently creative to warrant copyright protection, remanding Google's fair use defense to the district court for trial.[19]

The jury in the second trial found that Google's reimplementation of the APIs was fair use.[20] Once again, Oracle appealed. And once again, the Federal Circuit reversed,[21] issuing an opinion that—particularly if applied beyond the facts of the Java dispute and adopted by other circuits or the Supreme Court—has the potential to significantly alter the topography of software copyright law by narrowing the applicability of fair use in cases involving code.

Much of the commentary on the decision has focused on the Federal Circuit's approach to the jury's verdict.[22] The court gave strikingly little deference to that general verdict finding fair use, explaining that deference was not appropriate as to "legal facts"—only as to "historical facts."[23] We do not address this issue beyond noting that the Federal Circuit's approach suggests that, going forward, fair use decisions will rest even more firmly in the hands of judges and not juries, giving the court's reasoning additional weight.

---

[15] *Id.*

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

[17] *See id.*

[18] 28 U.S.C. § 1295(a)(1) (2012).

[19] Oracle Am., Inc. v. Google Inc. (*Oracle II*), 750 F.3d 1339, 1358-73 (Fed. Cir. 2014).

[20] *Oracle IV*, 886 F.3d at 1186.

[21] *Id.*

[22] *See, e.g.*, David Nimmer, *Juries and the Development of Fair Use Standards*, 31 HARV. J.L. & TECH. 563 (2018).

[23] *Oracle IV*, 886 F.3d at 1192–96.

Accordingly, the court's reasoning on the fair use factors is the focus here. Parts I through IV discuss the four statutory fair use factors and how the Federal Circuit interpreted them. Applying the court's logic, three of these four factors would typically—if not always—weigh against a finding of fair use in software cases, while the remaining factor would carry only minimal weight, making fair use a tough argument in most software infringement cases.[24] As a result, and as we explain in Part V below, technology companies interested in building new products using another company's APIs are likely to have a harder time proving that any alleged copying was fair use—and may be more reluctant to rely on fair use in their development decisions. Still, subsequent courts may view the Federal Circuit's ruling as a one-off decision, limited to its arguably unique facts and parties.

I
FACTOR ONE: THE PURPOSE AND CHARACTER OF THE USE

The Federal Circuit began its analysis of the jury verdict with the first of the four fair use factors: the purpose and character of the defendant's use.[25] As a first step, the court evaluated the degree to which Google's use of the Java APIs was commercial.[26] The more that a given use can be described as purely commercial, the more challenging it is to be ruled fair—even though many courts have held that wholly commercial uses do not necessarily negate fair use.[27] At trial, Android's commerciality was a disputed factual question before the jury, which heard evidence both of Android's immense success and of Google's practice of making the operating system open-source and available free of charge.[28] The jury apparently gave weight to these non-commercial considerations in its general verdict of fair use. Yet the Federal Circuit ultimately held that, because Google's purpose in using the APIs was fundamentally commercial—for use in phones, which are commercial

[24] The four non-exhaustive fair use factors as set out in 17 U.S.C. § 107 are:

(1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes;

(2) the nature of the copyrighted work;

(3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and

(4) the effect of the use upon the potential market for or value of the copyrighted work.

[25] 17 U.S.C. § 107(1) (2012).

[26] *Oracle IV*, 886 F.3d at 1196-97.

[27] *See, e.g.*, Blanch v. Koons, 467 F.3d 244, 254 (2d Cir. 2006); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1522 (9th Cir. 1992).

[28] *Oracle IV*, 886 F.3d at 1197.

products—any other "non-commercial motives [were] irrelevant as a matter of law."[29]

Such reasoning appears to convert the first factor's commerciality analysis from a spectrum—where non-commercial motives might cut in favor of fair use, despite otherwise commercial features—to a binary choice. If the purpose of the use is meaningfully commercial, other mitigating considerations such as free distribution or open source code become irrelevant as a matter of law.[30] This poses a potentially significant hurdle for software cases, where most (if not nearly all) developers have at least partly commercial motives regardless of how freely accessible or modifiable they make their code. These motives render the developers' software, by the Federal Circuit's reasoning, entirely commercial. The court's approach would therefore limit the extent to which a defendant can dispute commerciality in a software case.

The Federal Circuit's ruling presents defendants with similar obstacles regarding the second element of the first fair use factor, which considers whether the use was "transformative."[31] Transformative use—a use that adds something new, altering the purpose or character of the underlying material—generally favors finding fair use.[32] But when evaluating the transformative quality of a work, courts must first grapple with the question of what that work actually is. For instance, in this case, is it the original APIs themselves? Or is it the APIs as reimplemented for their new context, a novel smartphone operating system? With software, the approach to this inquiry will generally decide the outcome of the factor one analysis. After all, an API in itself only has one purpose: to let one program talk to another. But in the context of a fuller software ecosystem, an API adapted for one implementation could have very different functionality than it would adapted in another implementation.

Here, the Federal Circuit focused primarily on the purpose of the APIs themselves, rather than on the larger context of how Google had specifically incorporated the APIs into the Android operating system.[33] Google did not appropriate Java code in its entirety. Instead, Google copied key definitional aspects

---

[29] *Id.*

[30] *Id.*

[31] *Id.* at 1198 ("[T]he Supreme Court has stated that the 'central purpose' of the first fair use factor is to determine 'whether and to what extent the new work is transformative.'") (quoting Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 579 (1994)).

[32] *See Campbell*, 510 U.S. at 579.

[33] *Oracle IV*, 886 F.3d at 1197–1204.

of Java code including structure, sequence, and organization (SSO) for a specific purpose: clarifying which specific Java methods would be implemented.[34] For instance, the math declarations in Android still called up the math methods originally defined in Java, although Google had rewritten the implementation portions of those methods.[35] So, the "max" method in Android would find the maximum of two numbers—just as that method did in Java—but the underlying code in Android was totally different from the corresponding Java code. Even so, the court seemed to limit its focus here to *what* Google's code did, as opposed to *how* the code was written as compared to Oracle's code.[36]

By focusing on APIs' methods in themselves, divorced from the overall context in which they appear, the court may have made it difficult for almost any use of software to qualify as transformative in the fair use analysis. In a sense, all declaratory code structured in a certain way has one primary purpose—to execute the defined function. As Google argued, such code cannot both remain itself and acquire new purpose or use unless its context changes and the original method interacts with new implementations, creating something arguably new and transformative—such as a new type of operating system. But even in such a situation, the copied declaratory code retains its original purpose in a broad sense because it still orders a computer to perform the command for which it was defined. That portion of code was written to command a certain task and regardless of context will always command that task, whether in an operating system, ride-sharing app, or something else.

This is in contrast to other "functional" yet expressive works—a news photograph, for example. Speaking generally, code orders a computer to perform commands, while photographs convey information. But one's perception of the information a photograph conveys can change significantly depending on the photograph's use or the context of its presentation. A photograph's expression of information could be serious in one context or parodic in another, with just a few elements changed. For example, in the Second Circuit's 1998 opinion in *Leibovitz v. Paramount Pictures Corp.*, a movie studio Photoshopped comic actor Leslie Nielsen's head onto the body of a naked pregnant woman, to promote the actor's new film.[37] Nielsen's head aside, the lightning and body positioning were almost identical to those elements from a famous photograph of Demi Moore, taken by

[34] Oracle Am., Inc. v. Google Inc. (*Oracle III*), 2016 WL 3181206, at *4–5 (N.D. Cal. June 8, 2016).

[35] *Id.* at *3–7.

[36] *Oracle IV*, 886 F.3d at 1199–1202.

[37] Leibovitz v. Paramount Pictures Corp., 948 F. Supp. 1214, 1215 (S.D.N.Y. 1996).

portraitist Annie Leibovitz. Leibovitz sued Paramount for copyright infringement, but the court ruled the use was fair: compared to Leibovitz's serious portrait, Paramount's poster clearly parodies the original.[38] Placed in a new context, such works can serve entirely new purposes. They can communicate a very different message in a different context, even if the underlying work does not change much or at all.

Similarly, software can be used in different contexts to achieve different results. But under the Federal Circuit's analysis, declaratory code always has the same "purpose"—unless, in the somewhat limited example the court offered, the code is used for such a different "purpose" as "teaching how to design an API."[39] As the court elaborated, "merely copying the material and moving it from one platform to another without alteration is not transformative," even if the material is used in a new context that, viewed holistically, produces a new and very different work.[40] Indeed, as the court explained, the fact that "Google wrote its own implementing code [was] irrelevant" to the analysis because the underlying APIs themselves were unaltered.[41]

This reasoning suggests that almost any use of software code in a new context—save perhaps uses for instructional purposes—will fail the first fair use prong. At a minimum, this logic suggests that almost no use of pre-existing APIs could be transformative unless its specific function were modified in some way. But this would likely mean that the declaratory code was, to some significant degree, no longer the same code at all.

II
FACTOR TWO: THE NATURE OF THE WORK

Under the second fair use factor, a court analyzes the nature of the copyrighted work.[42] It evaluates whether the copied material is more creative—and therefore nearer the heart of copyright protection—or more functional, informational, or factual.[43] Typically, fair use is "more difficult to establish" when the copied work is

[38] *Id.* at 1226.
[39] *Oracle IV*, 886 F.3d at 1201.
[40] *Id.*
[41] *Id.*
[42] 17 U.S.C. § 107(2) (2012).
[43] *See* Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 586 (1994).

predominantly creative, but easier to establish when the copied work is less creative.[44]

In this case, the Federal Circuit recognized that the Java APIs were substantially functional, even if they "involved some level of creativity."[45] As the Federal Circuit acknowledged, this should usually cut in favor of fair use. Presumably, the jury's fair use verdict reflected a finding that the APIs at issue were closer to the functional end of the spectrum of creativity. But the court ultimately concluded that factor two should generally not figure significantly one way or the other in the fair use analysis, because giving significance to this factor "could effectively negate Congress's express declaration—continuing unchanged for some forty years—that software is copyrightable."[46]

Notably, the Federal Circuit seemed to view the case as being about software generally—not about APIs in particular. Software as a category can include a range of code, from implantation code, to APIs, to simple programs, to functional but highly complex and creative programs, to abstract or purely expressive programs. But the Federal Circuit did not cabin its analysis to APIs or even extremely functional, though still protectable, programs.[47] Rather, it treated all code as software and all software as protectable, such that its particular degree of creativity should not be discounted at all in the fair use analysis.[48]

Perhaps the Federal Circuit panel felt constrained by its broad 2014 ruling on protectability of APIs, making it harder for the court to draw nuanced lines between expression and functionality in its decision on fair use.[49] In any event, the court's approach to the second factor presents a hurdle for software copyright defendants claiming fair use. Given the functional nature of much code, one might presume that this factor should nearly always favor the defendant—whether or not the factor was significant in the overall balancing of factors in a specific case. But the Federal Circuit's approach essentially reads this factor out of the statute, rendering it at best neutral in software cases.

---

[44] *See id.*
[45] *Oracle IV*, 886 F.3d at 1205.
[46] *Id.*
[47] *See id.*
[48] *See id.*
[49] *See* Oracle Am., Inc. v. Google Inc. (*Oracle II*), 750 F.3d 1339 (Fed. Cir. 2014).

III
FACTOR THREE: THE AMOUNT AND SUBSTANTIALITY OF THE USE

The Federal Circuit's analysis of the third fair use factor, which evaluates "the amount and substantiality of the portion used in relation to the copyrighted work as a whole,"[50] similarly seems tilted against finding fair use in cases involving software code. At trial, the jury's general verdict apparently reflected a factual determination that Google had copied a relatively small portion of the work at issue—the 37 API packages of the Java codebase.[51] Although the parties had stipulated that only 170 lines of code were necessary for programmers to write in the Java language, Google had copied approximately 11,500 lines of code.[52] But this was a tiny percentage of the roughly 5 million lines of code in Java as a whole.[53]

The Federal Circuit, however, did not dwell on whether Google copied only a very small portion of the "copyrighted work as a whole," as the statute says.[54] In not doing so, the court's approach arguably deviates from the approach most courts have used since the Supreme Court's 1985 decision in *Harper & Row v. Nation Enterprises*, which focuses on the percentage of the whole and significance of what the defendant copied.[55] Instead, the Federal Circuit was more concerned with the fact that Google had copied certain APIs in their entirety, regardless of the fact that those APIs were a small fraction of the total number of lines of codes comprising the Java programming environment.[56] This narrow approach mirrored the court's transformative use inquiry, which considered only the copied code by itself, as opposed to in its new context. The court's approach also resulted in a similarly defendant-unfriendly finding. Because Google copied entire APIs, this factor counted against fair use even though what Google copied was not much compared to the "copyrighted work as a whole"—so long as the "work" is limited to constituent pieces of a bigger, more comprehensive, piece of software.[57]

The court also stressed that the portions copied by Google could not be "qualitatively insignificant, particularly when the material copied was important to the creation of the Android platform."[58] First, this reasoning inverts the way courts

---

[50] 17 U.S.C. § 107(3) (2012).
[51] *Oracle IV*, 886 F.3d at 1206.
[52] *Id.*
[53] *Id.*
[54] 17 U.S.C. § 107(3).
[55] Harper & Row, Publishers, Inc. v. Nation Enters., 471 U.S. 539, 564–66 (1985).
[56] *Oracle IV*, 886 F.3d at 1206–07.
[57] *Id.*
[58] *Id.* at 1207.

have generally approached the third factor, as it focuses on the significance of the copied material to the *defendant's* work instead of its significance to the *plaintiff's* work.[59] Second, even focusing on the significance to defendants, Google copied only 37 of the 168 APIs in the Android platform,[60] meaning even relatively small portions satisfy the significance test the Federal Circuit used. This approach seems to put a heavy thumb on the scale of the fair use analysis. Because practically any copied code will serve a function in the defendant's program, such code will usually be "important."[61] Again, the Federal Circuit's approach, if followed by other courts, makes it likely that the third factor will generally favor the software copyright holder.

IV
FACTOR FOUR: MARKET HARM

The fourth fair use factor considers how the defendant's work affects the market for the original, with any market harm cutting against finding fair use.[62] This market includes potential future markets for derivative uses of the original, including unrealized works that the copyright holders or licensees may develop.[63] The Federal Circuit again found that this factor favored Oracle.[64]

The court focused on Android's potential to harm Oracle's efforts in the smartphone industry. Although the jury's general verdict seemingly reflected agreement with Google's argument that the Java APIs' market was limited to desktop and laptop computers, Oracle pointed to evidence that it had licensed Java for use in early smartphones before Google created the more-sophisticated Android operating system.[65] The Federal Circuit was persuaded that this presented problematic market harm Oracle could have suffered. It stated that smartphones were a "traditional, reasonable, or likely to be developed market" subject to the factor four analysis.[66] It also pointed to evidence that Android was already being used as a direct substitute for Java—such as when Amazon negotiated a discounted Java licensing

---

[59] *See, e.g.*, Peter Letterese & Assocs., Inc. v. World Inst. of Scientology Enters., 533 F.3d 1287, 1314 (11th Cir. 2008); Consumers Union of U.S., Inc. v. Gen. Signal Corp., 724 F.2d 1044, 1050 (2d Cir. 1983).

[60] Oracle Am., Inc. v. Google Inc. (*Oracle II*), 750 F.3d 1339, 1350-51 (Fed. Cir. 2014).

[61] *Oracle IV*, 886 F.3d at 1207.

[62] 17 U.S.C. § 107(4) (2012).

[63] Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 590 (1994); Harper & Row, Publishers, Inc. v. Nation Enters., 471 U.S. 539, 568 (1985).

[64] *Oracle IV*, 886 F.3d at 1210.

[65] *Id.* at 1209.

[66] *Id.*

fee from Oracle based on Android being a free alternative.[67] This, for the court, proved actual market harm.[68]

This may be the right result on the facts before it, but the Federal Circuit's broad approach can be read to suggest that functional code—if not other works—will very often be perceived as having a broad potential market. The court's reasoning appeared to be that almost *any* market where a copyrighted work, or part of it, can be used is within the "potential market" of the copyright holder.[69] As the court explained, "a market is a potential market even where the copyright owner has no immediate plans to enter it or is unsuccessful in doing so."[70] Under this reasoning, once a copyright defendant has succeeded in a market using the plaintiff's work, that market is almost necessarily a "potential market" the plaintiff might have exploited—and has therefore lost—because of the defendant's copying. As a result, the fourth factor will usually favor the plaintiff, as it did here.

The Federal Circuit also emphasized that the two companies had previously been involved in licensing negotiations regarding the potential use of Oracle's Java software in a Google smartphone.[71] Although these negotiations were unproductive, the court regarded their existence as further evidence of Oracle's longstanding interest in entering the smartphone market.[72] While this reasoning may have a certain logic, it is also puzzling. Courts analyzing fair use have sometimes considered whether the defendant sought permission to copy the plaintiff's work, but they have usually asked this question in the context of the first factor, in looking at the character of the use.[73] Cautioning against taking the issue too far, the Supreme Court's decision in *Campbell v. Acuff-Rose* suggested that unsuccessfully seeking permission should not be read to show bad faith inconsistent with fair use. The Court reasoned: "[i]f the use is otherwise fair, then no permission need be sought or granted."[74] Perhaps trying to avoid the issue, the Federal Circuit said it was not

---

[67] *Id.*

[68] *Id.* at 1209–10.

[69] *Id.* at 1210.

[70] *Id.* (citations omitted).

[71] *Id.* at 1209.

[72] *Id.* Notably, the negotiations did not concern the limited portions of code Google actually copied—they were about the entirety of the Java APIs, including all interfaces and implementing code. *See* Oracle Am., Inc. v. Google Inc. (*Oracle III*), 2016 WL 3181206, at *11 (N.D. Cal. June 8, 2016).

[73] *See* Simon J. Frankel & Matt Kellogg, *Bad Faith and Fair Use*, 60 J. COPYRIGHT SOC'Y U.S. 1, 9–12 (2012).

[74] Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 585 n.18 (1994).

considering that the negotiations were unsuccessful—only that they showed "Oracle's interest in the potential market for smartphones."[75]

This explanation arguably proves too much. Of course, there will only be litigation over fair use when licensing negotiations are unsuccessful. Otherwise, infringement claims are unlikely to arise. But the fact that a party approaches a copyright holder and seeks a license does not mean that the copyright holder—here, Oracle—necessarily has an "interest in the potential market."[76] Most copyright holders, presented with a request to license their works for new uses, would probably be willing to at least consider negotiating. However, such willingness to negotiate does not mean the copyright holder would have exploited the market on its own. In essence, the Federal Circuit seems to have taken failed license negotiations—which *Campbell* effectively banned from consideration under the first factor—and evaluated them under the fourth factor through the guise of market harm.[77] Time will tell if other courts adopt this approach.

V
OVERALL IMPLICATIONS

The Federal Circuit's analysis appropriately focused on the facts before the court. And it may well be that on those facts, reasonable minds could disagree about the appropriate result. But stepping back, the court's analytical approach to the fair use factors may have the long-term effect of tilting the fair use playing field sharply against defendants in software code cases.

Perhaps most striking, the Federal Circuit's analysis of the first fair use factor appears to make it very difficult for defendants accused of infringing API packages and their SSO to show they are using the APIs for a new and different purpose, such that it would qualify as "transformative." Outside of some kind of teaching context, as the Federal Circuit suggested,[78] the API packages will almost always be serving the same narrow function in the defendant's work as in the plaintiff's, even if the overall work where the copied APIs appear or the implementation is new and different. Combined with the court's analysis of the other three factors—which will almost always either disfavor fair use or be neutral when computer code is at issue—it is difficult to conceive of circumstances where using more than a shred of an SSO or API (other than for teaching, perhaps) can now qualify as fair.

---

[75] *Oracle IV*, 886 F.3d at 1209 n.14.
[76] *Id.*
[77] *Campbell*, 510 U.S. at 585 n.18.
[78] *Oracle IV*, 886 F.3d at 1201.

If this understanding of the Federal Circuit's analysis is correct, it may become harder for one developer to use another's APIs in new products. After all, one apparent result of the court's analysis is that it may now be more difficult to make a fair use of software, as compared to use of a more expressive work. If this is the opinion's practical result, *Oracle v. Google* departs from the common view that fair use should be a more accessible defense where, as with software, the disputed material is mostly functional.[79] Again, only time will tell if that is the effect of the Federal Circuit's decision—or if the decision turns out to be one largely limited to its unusual facts, regarding a discrete portion of functional code, copied to make a new and unusually successful product. It is also possible that courts will look to other copyright doctrines, such as merger or *scènes à faire*, to allow borrowing of APIs to some extent. Such doctrines may become more prominent if fair use fades. For now, however, the potential application of fair use to software appears substantially diminished, and the practical impact of the Federal Circuit's decision on software development remains to be seen.

[79] *See, e.g.*, Sony Comput. Entm't, Inc. v. Connectix Corp., 203 F.3d 596, 605 (9th Cir. 2000); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1522 (9th Cir. 1992).