Harvard Journal of Law & Technology
Volume 18, Number 1 Fall 2004

ANTITRUST ANALYSIS IN SOFTWARE PRODUCT MARKETS: A FIRST PRINCIPLES APPROACH

*Andrew Chin**

TABLE OF CONTENTS

---

* Associate Professor of Law, University of North Carolina at Chapel Hill. The author was a volunteer extern to the Hon. Thomas P. Jackson from September 1999 to December 1999, during which time the author assisted in the drafting of the Findings of Fact in United States v. Microsoft Corp., 84 F. Supp. 2d 9 (D.D.C. 1999). The author received Judge Jackson's express permission to comment publicly on the case. All views expressed herein are solely the author's own. The author would like to thank Scott Baker, Adrienne Davis, Dennis Karjala, Joe Kennedy, Mark Lemley, and David McIntosh for insightful suggestions, and Nathan Brown, Jesh Humphrey, Todd Ito, and Jaison Thomas for able research assistance.

I. INTRODUCTION

Antitrust cases frequently involve a determination as to whether the defendant has, or is likely to obtain, monopoly power or market power in some relevant market.[1] It is therefore critically important for parties and courts to begin their legal analyses by accurately defining the relevant market in question. This line of inquiry is expressly indicated by the federal antitrust statutes, which condemn monopolization of "any part of . . . trade or commerce"[2] and mergers that tend to lessen competition "in any line of commerce in any section of the country."[3] A properly defined relevant market is necessary to calculate a firm's market share for the purpose of inferring individual market power.[4] Also, the analysis of conduct under the rule of reason[5] ordinarily calls for the definition of a market in which the conduct may or may not be found to have unreasonable anticompetitive ef-

1. *See* United States v. E.I. du Pont de Nemours & Co., 351 U.S. 377, 391 (1956) ("Monopoly power is the power to control prices or exclude competition."); *see also* Eastman Kodak Co. v. Image Technical Servs., Inc., 504 U.S. 451, 464 (1992) ("[Market power is] the ability . . . to raise price and restrict output." (internal quotation marks omitted)). For a comprehensive discussion of the relationship between the market definition and market power inquiries, see 2A PHILLIP E. AREEDA ET AL., ANTITRUST LAW: AN ANALYSIS OF ANTITRUST PRINCIPLES AND THEIR APPLICATION ch. 5 (vol. IIA 1995).

2. 15 U.S.C. § 2 (2004).

3. 15 U.S.C. § 18 (2000).

4. AREEDA, *supra* note 1, ¶¶ 531c–d, at 157–58.

5. Most antitrust scrutiny is conducted under the "rule of reason," whereby the court must determine whether the practice in question constitutes an unreasonable restraint on competition under all the circumstances of the case. *See* Arizona v. Maricopa County Med. Soc'y, 457 U.S. 332, 342–43 (1982). Certain practices, however, are condemned as *per se* violations of the antitrust laws, meaning that they "are conclusively presumed to be unreasonable and therefore illegal without elaborate inquiry as to the precise harm they have caused or the business excuse for their use." N. Pac. Ry. Co. v. United States, 356 U.S. 1, 5–6 (1958).

fects.[6] Accordingly, market definition has become an integral part of common-law doctrines relating to such diverse conduct as monopolization,[7] mergers,[8] tying,[9] exclusive dealing,[10] territorial and customer restrictions,[11] and non-price horizontal restraints.[12] In sum, "the most important single issue in most [antitrust] enforcement actions — because so much depends on it — is market definition."[13]

The task of defining a relevant market "is as difficult an undertaking as any in antitrust,"[14] even when the products under consideration

---

6. *See* ABA SECTION OF ANTITRUST LAW, ANTITRUST LAW DEVELOPMENTS 495 (4th ed. 1997) ("Ascertaining the restraint's competitive effects [under the rule of reason] ordinarily requires a definition of the relevant market and an analysis of the restraint's effect on competition within that market . . . ." (citation omitted)); Phillip E. Areeda, *The Rule of Reason: A Catechism on Competition*, 55 ANTITRUST L.J. 571, 576–77 (1986) (noting that market definition is the usual approach to assessing the potential for anticompetitive effects, but that such an approach is "superfluous if we have already observed adverse effects"); *see also* Thomas E. Kauper, *The Problem of Market Definition Under EC Competition Law*, 20 FORDHAM INT'L L.J. 1682, 1685 (1997) ("Market definition is now an essential element in a broad range of U.S. cases. Indeed, one can plausibly argue that it is necessary in all but cartel and resale price maintenance cases.").

7. *See, e.g.*, United States v. Grinnell Corp., 384 U.S. 563, 570–71 (1966) ("The offense of monopoly under § 2 of the Sherman Act has two elements: (1) the possession of monopoly power in the relevant market and (2) the willful acquisition or maintenance of that power as distinguished from growth or development as a consequence of a superior product, business acumen, or historic accident.").

8. *See*, *e.g.*, FTC v. H.J. Heinz Co., 246 F.3d 708, 715 (D.C. Cir. 2001) (explaining that the government establishes a presumption of anticompetitive effect by showing that a merger "would produce a firm controlling an undue percentage share of the relevant market, and [would] result[ ] in a significant increase in the concentration of firms in that market." (internal quotation omitted)).

9. *See*, *e.g.*, Jefferson Parish Hosp. Dist. No. 2 v. Hyde, 466 U.S. 2, 20–21 (1984) ("[A] tying arrangement cannot exist unless two separate product markets have been linked.").

10. *See, e.g.*, Tampa Elec. Co. v. Nashville Coal Co., 365 U.S. 320, 327 (1961) (stating that "the line of commerce" (i.e., the product market) and "the area of effective competition" (i.e., the geographic market) must be delineated in order to determine whether an exclusive dealing arrangement will foreclose "competition . . . in a substantial share of the line of commerce affected" (internal quotation marks omitted)).

11. *See* ABA SECTION OF ANTITRUST LAW, *supra* note 6, at 154–55 n.852 (4th ed. 1997) ("Courts [reviewing territorial and customer restrictions under the rule of reason] typically require plaintiffs to show that a supplier has sufficient market power to affect competition in the relevant market.").

12. *See, e.g.*, Rothery Storage & Van Co. v. Atlas Van Lines, Inc., 792 F.2d 210, 217 (D.C. Cir. 1986) (defining a nationwide market for the interstate carriage of used household goods in order to "analyze the economic nature and effects of the system [of non-price horizontal restraints] Atlas has created").

13. Robert Pitofsky, *New Definitions of Relevant Market and the Assault on Antitrust*, 90 COLUM. L. REV. 1805, 1807 (1990).

14. MILTON HANDLER ET AL., TRADE REGULATION: CASES AND MATERIALS 215 (3d ed. 1990); *see also* U.S. Healthcare, Inc. v. Healthsource, Inc., 986 F.2d 589, 598 (1st Cir. 1993) ("There is no subject in antitrust law more confusing than market definition."). Evaluating the practical problems of defining a market for a particular case, Luanne Sacks and Garrett Dillon observe that:

> If in fact the settlements fail, and the district court is faced with remand of the tying claim, it will be faced with a technically onerous product analyses, confused by self-serving, yet possibly meritorious,

are well understood; it may well be hopeless when the products are poorly understood.[15] In *United States v. Microsoft Corp.*, the district court issued separate findings of fact[16] and conclusions of law[17] holding Microsoft liable for tying under Section 1 of the Sherman Act.[18] The court of appeals upheld the findings of fact in their entirety,[19] but criticized the government's failure to explain "what constitutes a browser (i.e., what are the technological components of or functionalities provided by a browser) and . . . why certain other products are not reasonable substitutes."[20] The court of appeals ruled that the government had failed to establish "a precise definition of browsers" and "a careful definition of the tied good market" at trial, and would be precluded from doing so on remand.[21] In the face of these impediments, the government decided to drop the tying claim.[22]

The government's definitions may have been insufficiently explicit and precise, but Microsoft's definition of a software product was downright false and misleading. Throughout the litigation, Microsoft maintained the position that "software products consist of code and nothing else[.]"[23] This position is untenable, not least because it makes a mockery out of copyright. A person who, after legitimately obtaining the Windows software product, made and sold pirated copies of the software in the belief that he or she had "bought" or "leased" the code would promptly be disabused of that notion by Microsoft's own legal department.[24]

> arguments of innovation and efficiency offered by all the market participants, and little guidance from established case law.

Luanne Sacks & Garrett Dillon, *The Microsoft Decision: A Vivid Reminder That Market Definition Can Make or Break Your Case*, *in* 22ND ANN. INST. ON COMPUTER L. 429, 474 (PLI Intellectual Prop. Course, Handbook Series No. G-691, 2002).

15. *See, e.g.*, United States v. Microsoft Corp., 980 F. Supp. 537, 545 (D.D.C. 1997) (appointing then-Harvard law professor Lawrence Lessig as special master to resolve, *inter alia*, "the complex issues of cybertechnology" in connection with the predecessor contempt case); *cf.* United States v. Microsoft Corp., 147 F.3d 935, 950 (D.C. Cir. 1998) (declining to "put[ ] judges and juries in the unwelcome position of designing computers" (internal quotation marks omitted)); Richard A. Posner, *Antitrust in the New Economy*, 68 ANTITRUST L.J. 925, 937 (2001) ("Computer science and communications technology are much more difficult areas than the average body of scientific or engineering knowledge that lay judges and jurors are asked to absorb en route to rendering a decision.").

16. United States v. Microsoft Corp., 84 F. Supp. 2d 9 (D.D.C. 1999).

17. United States v. Microsoft Corp., 87 F. Supp. 2d 30 (D.D.C. 2000).

18. *See id*. at 56.

19. *See* New York v. Microsoft Corp., 224 F. Supp. 2d 76, 98 (D.D.C. 2002) ("Because all of the district court's factual findings survived challenge on appeal, they comprise the law of this case and may be relied upon during the remedy phase of this proceeding.").

20. United States v. Microsoft Corp., 253 F.3d 34, 81–82 (D.C. Cir. 2001).

21. *See id*.

22. *See* New York v. Microsoft Corp., 224 F. Supp. 2d at 95 (citing Joint Status Report (Sept. 20, 2001) at 2).

23. Defendant's Revised Proposed Findings of Fact at 263, United States v. Microsoft Corp., 84 F. Supp. 2d 9 (D.D.C. 1999) (No. 98-1232) (on file with author).

24. *See, e.g.*, Microsoft Corp. v. CMOS Techs., Inc., 872 F. Supp. 1329 (D.N.J. 1994) (granting summary judgment of copyright infringement to Microsoft).

The purchaser of a software product does not acquire plenary property rights in the accompanying software; rather, he or she purchases legal rights and technological capabilities to use certain services that may be performed by his or her computer system when the accompanying software is installed and executed on the system under certain specified conditions. The purchase of a software product is not the purchase of software code, but the purchase of these rights and capabilities. A software product market is not a market for software code, but a market for these rights and capabilities.

Despite these seemingly basic points, the entire record of the *Microsoft* case is virtually devoid of a reasonably accurate working definition of a software product for purposes of antitrust analysis. There have been two qualified exceptions. During the trial before Judge Jackson, one of the government's computer science expert witnesses, Princeton University Professor Edward Felten, sought to draw a distinction between software products and software code.[25] His efforts were met with puzzlement, however, because he was unable to articulate this distinction in legal terms.[26] Later, in an amicus brief filed at Judge Jackson's request,[27] Stanford Law School Professor Lawrence Lessig observed that viewing software products as code "would create many potential paradoxes of identity,"[28] and concluded that a software product should instead be defined as "functionality separately valued by consumers."[29] Judge Jackson, however, did not find this definition explicit or precise enough to dissuade him from relying on the more intuitive notion that software products consist of code.[30]

The fallacious premise that software products consist of code has also been pervasive in the previous legal literature on *Microsoft*. Vari-

---

25. *See* June 10, 1999 P.M. Session Trial Transcript at 17, United States v. Microsoft Corp., 87 F. Supp. 2d 30 (D.D.C. 2000) (No. 98-1232) (testimony of Edward Felten) ("I'm talking about Microsoft's browser product, which is, as I've said, I do not identify with any particular lines of code."), *available at* 1999 WL 380891.

26. On cross-examination during his rebuttal testimony, Felten engaged in the following colloquy with Microsoft attorney Steven Holley:

> Q. [Holley] Let me see if I can understand that one. You say that you can claim a copyright on software code which is somehow different than the product? . . .
>
> The Witness [Felten]: I admit I'm not an expert on copyright law, but whether you can — but code and products are different things, as I said many times. So, whether you can copyright code or copyright products, I don't know. I don't see the connection.

Id. at 34.

27. Brief of Professor Lawrence Lessig as Amicus Curiae at v, United States v. Microsoft Corp., 87 F. Supp. 2d 30 (D.D.C. 2000) (No. 98-1232) (Feb. 1, 2000), *available at* http://cyberlaw.stanford.edu/lessig/content/testimony/ab/ab.pdf.

28. *Id*. at 20.

29. *Id*.

30. *See* United States v. Microsoft Corp., 87 F. Supp. 2d 30, 50 (D.D.C. 2000) (characterizing the challenged tying arrangement as requiring consumers "to take, and pay for, the entire package of software").

ous commentators have discussed the sale of code,[31] used the terms "software" and "software product" interchangeably,[32] and referred to the code that comprises Microsoft's Windows and Internet Explorer software products.[33]

As one of the most important and most studied cases in the history of antitrust, *Microsoft* has rightly been added to the elite canon of principal cases that comprise the basic antitrust curriculum.[34] Even so, the *Microsoft* decisions, and their accompanying secondary literature, fail to teach any generally applicable procedures for defining a software product and a relevant software product market.[35] Without these

---

31. *See, e.g.*, Michael L. Katz & Carl Shapiro, *Antitrust in Software Markets*, *in* COMPETITION, INNOVATION AND THE MICROSOFT MONOPOLY: ANTITRUST IN THE DIGITAL MARKETPLACE 29, 66 (Jeffrey A. Eisenach & Thomas M. Lenard eds., 1999) (defining the tying of software products "carefully" as the refusal "to sell program A (the 'tying' good) unless the customer also purchases program B (the 'tied' good)"); *id*. at 76 (noting that a possible antitrust response to *Microsoft* would be "a policy of requiring a modular approach to the production and sale of code, with well-defined, open interfaces between the modules"); George L. Priest, *Letter to Larry*, INDUS. STANDARD, June 26, 2000 ("Judge Jackson concluded that it is predatory for Microsoft to include Internet Explorer in Windows and to not charge extra for the added browser code.") *available at* 2000 WL 31584005.

32. *See, e.g.*, David K. Lam, *Revisiting the Separate Products Issue*, 108 YALE L.J. 1441, 1446–47 (1999) ("Microsoft can easily offer the two products separately because 'software code by its nature is susceptible to division and combination.'" (quoting United States v. Microsoft Corp., 147 F.3d 935, 951 (D.C. Cir. 1998))); Janusz A. Ordover & Robert D. Willig, *Access and Bundling in High-Technology Markets*, *in* COMPETITION, INNOVATION AND THE MICROSOFT MONOPOLY: ANTITRUST IN THE DIGITAL MARKETPLACE 103, 121 (Jeffrey A. Eisenach & Thomas M. Lenard eds., 1999) (describing Windows 98 as "a new product that, in effect, combines both the operating system software and the browser software into one technologically inseparable product.").

33. *See, e.g.*, David S. Evans, *All the Facts that Fit: Square Pegs and Round Holes in U.S. v. Microsoft*, 22 REG., Winter 1999, at 61 ("[T]he court does not mention the evidence . . . that the presence of software code that is within the court's apparent definition of 'IE' supports an improved 'Help' system for Windows itself and provides other benefits to Windows users."). Another analysis of Windows and Internet Explorer argues that:

> Like all software, browsers are, at bottom, binary code arranged in files, as is the operating system. To the extent Internet Explorer is a different product from Windows 95, it is because the sequences of 0's and 1's that perform 'browser functions' differ from the sequences of 0's and 1's that perform 'Windows 95 functions.'

Mark A. Lemley & David McGowan, *Could Java Change Everything? The Competitive Propriety of a Proprietary Standard*, 43 ANTITRUST BULL. 715 (1998).

34. For recent casebooks devoting extensive coverage to *Microsoft*, see, for example, ANDREW I. GAVIL ET AL., ANTITRUST LAW IN PERSPECTIVE 827–58, 908–17, 1028–38, 1081–90 (2002); THOMAS D. MORGAN, MODERN ANTITRUST LAW AND ITS ORIGINS 741–70 (2d ed. 2001); E. THOMAS SULLIVAN & HERBERT HOVENKAMP, ANTITRUST LAW, POLICY AND PROCEDURE: CASES, MATERIALS, PROBLEMS 569–77, 716–36 (5th ed. 2003).

35. In *Microsoft*, the D.C. Circuit affirmed Judge Jackson's delineation of a relevant worldwide market for Intel-compatible personal computer operating system software products, in which Windows 98 competes. *See Microsoft*, 253 F.3d at 54. Judge Jackson's analysis did not, however, begin with a precise definition of Microsoft's operating system software product. *See Microsoft*, 84 F. Supp. 2d at 12 (defining "operating system" as "a software program that controls the allocation and use of computer resources . . . [and] supports the functions of software programs, called 'applications,' that perform specific user-oriented tasks," but failing to define an operating system software product). His approach therefore cannot be relied on, as a more general matter, to identify a defendant's software

basic staples of antitrust analysis in hand, further study of the *Microsoft* case can provide only limited guidance to future antitrust practitioners in the software industry.

The purpose of this Article is to develop legally sufficient and generally applicable procedures for identifying the legal rights and technological capabilities that constitute a software product and for delineating the relevant market or markets in which a given software product competes. Because these techniques will be grounded in basic software engineering concepts and prevailing copyright and antitrust doctrines, this Article will refer to them collectively as the "first principles approach" to antitrust analysis.

At the outset, it will be necessary to expunge the false and misleading intuition that software products consist solely of code, and to supplant it with a legally and technologically accurate definition of a software product. In general terms, a software product is defined by reference to accompanying software and documentation, and consists essentially of the necessary legal rights, and technological capabilities, to install and run the software on a system according to the documentation; it does not include any of the software or documentation itself.[36] More explicit detail will be needed, however, to obviate the reliance of antitrust analysis on misleading intuitions;[37] for this, it will be necessary to look to copyright law and software engineering.

In addition to a more accurate definition of a software product, antitrust analysis also requires a rigorous methodology for defining the relevant markets in which a given software product competes. Accordingly, a series of structured inquiries is needed to identify, *inter alia*, products that "have reasonable interchangeability for the purposes for which they are produced"[38] or that support particular end uses for the given product that may be susceptible to price discrimination.[39] The role of antitrust may be understood in this context as promoting "well-functioning software product markets" by protecting price and/or quality competition among the identified products.

In Part II, this Article develops a structured approach to delineating software product markets. This Part begins by describing the role of product and geographic market definitions in antitrust jurisprudence and reviewing the legal doctrines governing product market definition. Next, this Article describes the purposes that a software product serves, and introduces the concept of an "essential use case," which describes software functionality at an appropriate level of ab-

---

product and to explain "why certain other products are not reasonable substitutes" for the defendant's product. *Microsoft*, 253 F.3d at 81–82.

36. *See infra* text accompanying note 165.

37. *See supra* text accompanying note 30.

38. United States v. E.I. du Pont de Nemours & Co., 351 U.S. 377, 399 (1956); *see also infra* Part II.A.

39. *See infra* text accompanying notes 141–144.

straction for purposes of determining reasonable interchangeability of use. Several technological impediments that may constitute structural barriers to entry into a software product market are also identified. This Section formulates a procedure is then formulated for defining the relevant product market in which a given software product competes by using concepts relating to demand and supply substitutability. Finally, it explains the practical relevance of protecting competition in these software product markets by describing the characteristics of a well-functioning market from the perspective of software development and innovation.

Part III explains the provenance of the legal rights that constitute a software product by examining the Copyright Act's grant of specific exclusionary rights to a software developer. First, it reviews legal doctrines governing the scope of copyrightable subject matter in software. Then, this Section examines the scope of the exclusionary rights that are implicated by a consumer's use of a software product. These rights may be conferred either by the copyright statute's default allocation of rights or by the terms of an enforceable user license. This analysis not only yields a more precise description of the legal rights and technological capabilities that constitute a software product, but also clarifies the limits of the Copyright Act as a warrant for exclusionary conduct that involves the licensing of copyrighted software.

To conclude, Part IV illustrates the practical applicability of the first principles approach by considering another case, *Syncsort Inc. v. Sequential Software, Inc.*, in which the plaintiff failed to take sufficient care in defining the relevant market in which the defendant's software product competes.[40] This Section shows that the pursuit of well-functioning software product markets may inform the law's response to software innovation in addressing the current controversy over the use of peer-to-peer network ("P2P") software products to trade copyrighted files over the Internet. Some final remarks follow regarding the pursuit of "human-centric" computing.

Obviously, the first principles approach also has potentially profound implications for *Microsoft*. If the government plaintiffs had been able to establish "a precise definition of browsers" and "a careful definition of the tied good market" at trial, they may have been able to prevail on the tying claim before the D.C. Circuit. To see whether this would actually have been the case, however, will require a detailed review of the tying claim's extraordinarily complex litigation history and a careful liability analysis under each of the alternative doctrines for adjudicating that claim in light of the first principles approach pre-

40. 50 F. Supp. 2d 318 (D.N.J. 1999).

sented here. This analysis is beyond the scope of this Article, but is fully set forth in a companion piece.[41]

II. DEFINING SOFTWARE PRODUCT MARKETS

A. Market Definition Generally

In general, two products belong in the same relevant market when the ability of consumers and producers to substitute between them imposes an effective competitive constraint against the exercise of monopoly power.[42] The definition of a relevant market serves to describe a boundary between products[43] that compete with each other in this way and those that do not. This boundary has two dimensions which are determined through separate lines of analysis: a geographic market and a product market. A geographic market defines the "area of effective competition . . . in which the seller operates, and to which the purchaser can practicably turn for supplies."[44] A product market identifies "producers which, because of the similarity of their products, have the ability — actual or potential — to take significant amounts of business away from each other."[45]

Market definition, like the rest of antitrust jurisprudence, is not an exact science,[46] and the method described herein constitutes only one of many potentially valid approaches to defining product markets in the software industry.[47] The Supreme Court has characterized the product market inquiry as identifying those "products that have reasonable interchangeability for the purposes for which they are pro-

---

41. Andrew Chin, *Decoding* Microsoft*: A First Principles Approach*, 39 WAKE FOREST L. REV. (forthcoming 2005).

42. *See, e.g.*, U.S. DEP'T OF JUSTICE & FED. TRADE COMM'N, HORIZONTAL MERGER GUIDELINES §§ 1.11, 1.21 (1992), *reprinted in* 4 Trade Reg. Rep. (CCH) ¶ 13,104 [hereinafter HORIZONTAL MERGER GUIDELINES] (describing product and geographic markets as product groupings and regions in which a "hypothetical monopolist" could profitably impose a "small but significant and nontransitory" price increase); George J. Stigler, *Introduction* to NAT'L BUREAU OF ECON. RES., BUSINESS CONCENTRATION AND PRICE POLICY 4 (1955) ("An industry should embrace the maximum geographical area and the maximum variety of productive activities in which there is a strong long-run substitution.").

43. Throughout this Article, the term "products" refers to both products and services.

44. United States v. Philadelphia Nat'l Bank, 374 U.S. 321, 359 (1963) (quoting Tampa Elec. Co. v. Nashville Coal Co., 365 U.S. 320, 327 (1961)) (internal quotations omitted) (emphasis omitted).

45. SmithKline Corp. v. Eli Lilly & Co., 575 F.2d 1056, 1063 (3d Cir. 1978).

46. *See, e.g.*, Philadelphia Nat'l Bank, 374 U.S. at 360 n.37 (noting that "fuzziness" is inherent in the determination of a relevant geographic market); Pitofsky, *supra* note 13, at 1812 (arguing that market definition should be seen "as an array of estimates with no market description being exactly right").

47. *See*, *e.g.*, James A. Keyte, *Market Definition and Differentiated Products: The Need for a Workable Standard*, 63 ANTITRUST L.J. 697, 699 (1995) (noting "the lack of any clear standard for defining the relevant product market"); Pitofsky, *supra* note 13, at 1807 (noting the "persistent and unreconciled conflicts of approach [to market definition] in important judicial decisions").

duced — price, use, and qualities considered."[48] Phillip Areeda's treatise describes market definition as contingent on a "critical policy choice": namely, the extent and duration of market power that will be considered legally problematic in the context of any particular antitrust case.[49] This Article does not purport to provide a definitive standard for reasonable interchangeability or to resolve this critical policy choice. Rather, by identifying the attributes of and relationships among software products that are relevant to the product market inquiry, this Article will provide an analytical framework for determining such standards and choices in the context of any particular case.

This Article will only address the problem of defining product markets in the software industry and not the issue of defining geographic markets. The definition of software product markets warrants particular attention as a discipline in antitrust practice because it is the part of the market definition analysis that requires technology-specific methods.[50] Product market analysis in the software industry needs to consider the specific legal rights and technological capabilities that comprise a particular software product, so that similar products capable of "tak[ing] significant amounts of business away"[51] from it can be identified.

B. Product Markets Generally

The determination of a product market begins by identifying the defendant's product[52] as the initial product in a "provisional market."[53] The relevant product market is then defined as the market in

48. United States v. E.I. du Pont de Nemours & Co., 351 U.S. 377, 399 (1956).

49. AREEDA, *supra* note 1, ¶¶ 530b–c, at 152–54.

50. In contrast, geographic market analysis in the software industry focuses on the physical locations where producers and consumers can find each other to deal in a software product or its substitutes. In this respect, such geographic markets are identified using the same methods as in any other industry. Even though software and other information products are distinctive in that they may be distributed over the Internet, geographic market analysis does not examine any technological aspect of the software product itself. Mischaracterizations of software technology and intellectual property concepts are therefore more likely to lead to errors in product market definition than in geographic market definition. For an earlier appreciation of these difficulties, see Robert H. Lande & Sturgis M. Sobin, *Reverse Engineering of Computer Software and U.S. Antitrust Law*, 9 HARV. J.L. & TECH. 237, 252–53 (1996) (describing software product market definition as "an extremely complex and intensely fact-dependent area of law").

51. *See* SmithKline Corp. v. Eli Lilly & Co., 575 F.2d 1056, 1063 (3d Cir. 1978).

52. The product market definition analysis thus begins by taking the defendant's product as it is actually sold, without regard to the distinct question of whether the defendant's product is a "single product" under tying doctrine.

53. *See* AREEDA, *supra* note 1, ¶ 560, at 251. Antitrust liability may be based on harms to competition not only in product markets consisting of the economic substitutes for one product, but also in "cluster markets" that aggregate markets for numerous products sold by the defendant even though they may not be economic substitutes for each other. This approach is for administrative convenience and may not be undertaken where separate treatment of the products would result in a different conclusion regarding the existence or cause

which this initial product competes. The analysis proceeds by iteratively extending the boundaries of the provisional market to include additional products that may be significant substitutes for the products already found to be in the provisional market.[54] The provisional market is recognized as the relevant product market when no more such substitutes can be added. Substitution may occur on both the demand side (when consumers are able to switch from using one product to using another)[55] and the supply side (when producers are able to switch from making one product to making another).[56] If product *A* is in the relevant market, and a significant price increase beyond the competitive level in the price of *A* would induce customers of *A* to buy product *B* instead, or induce producers of *B* to make and sell *A* instead, then *B* should also be included in the relevant market.[57] In either case, products *A* and *B* "have the ability — actual or potential — to take significant amounts of business away from each other,"[58] and are deemed to be in effective competition with each other.[59]

The definition of a product market thus calls for a careful analysis of demand and supply substitutability. As the Supreme Court explained in *Brown Shoe Co. v. United States*, courts are to perform this analysis by examining the available evidence relating to (1) "reasonable interchangeability of use or the cross-elasticity of demand between the product itself and substitutes for it," and (2) seven "practical indicia," namely "industry or public recognition of the [product market] as a separate economic entity, the product's peculiar characteristics and uses, unique production facilities, distinct customers, distinct prices, sensitivity to price changes, and specialized vendors."[60]

---

of monopoly power. *See generally* United States v. AT&T, 524 F. Supp. 1336, 1375–77 (D.D.C. 1981) (aggregating markets for 200,000 products sold by defendant into a single cluster market).

54. *See* AREEDA, *supra* note 1, ¶ 560, at 251.

55. *See id*. ¶ 562, at 258–66.

56. *See id*. ¶ 561, at 252–58.

57. *See id*. ¶ 561, at 252.

58. SmithKline Corp. v. Eli Lilly & Co., 575 F.2d 1056, 1063 (3d Cir. 1978).

59. *See supra* text accompanying notes 42–45.

60. 370 U.S. 294, 325 (1962). While *Brown Shoe* identifies these indicia as relevant specifically in connection with the determination of "submarkets," courts and commentators have widely recognized their applicability to the delineation of product markets in general, and it is doubtful whether there remains any meaningful distinction between the identification of submarkets and product markets. *See generally* Rothery Storage & Van Co., v. Atlas Van Lines, 792 F.2d 210, 219 (D.C. Cir. 1986) (concluding that "submarket indicia" are best viewed as "proxies for cross-elasticities" of supply and demand); AREEDA, *supra* note 1, ¶ 533c, at 170–73 ("Only 'markets' are relevant.").

1. Demand Substitutability

The analysis of demand substitutability looks to "the reasonable interchangeability of use or the cross-elasticity of demand between the [initial] product itself and substitutes for it."[61] Although the cross-elasticity of demand between two products is a precise quantity,[62] in practice courts rarely consider precise cross-elasticity data.[63] Instead, most courts use the term more generally as a synonym for "reasonable interchangeability of use," as discerned from the qualitative tendency of an increase in the price of one product to result in an increase in the demand for a second product within a reasonably short time.[64]

Two products are said to exhibit reasonable interchangeability of use if (1) they are functionally interchangeable and (2) purchasers have a significant propensity to switch from one to the other in response to a change in price.[65] Strictly speaking, however, only the second of these criteria must be met: "The ultimate determinant of whether products belong in the same market is whether customers are willing to substitute one product for the other."[66] Functional interchangeability is, however, a necessary (but not sufficient)[67] condition

---

61. *Brown Shoe*, 370 U.S. at 325.

62. The cross-elasticity of demand is the percentage change in demand for one good attributable to a percentage change in the price of another good. *See* E. THOMAS SULLIVAN & JEFFREY L. HARRISON, UNDERSTANDING ANTITRUST AND ITS ECONOMIC IMPLICATIONS 31 (3d ed. 1998).

63. *See* ABA SECTION OF ANTITRUST LAW, *supra* note 6, at 503–05; *see also* AREEDA, *supra* note 1, ¶ 531, at 187 (noting that if the defendant's own elasticities of supply and demand were known, it would be possible to infer market power directly, and therefore unnecessary to infer it from market share and market definition).

64. *See* AREEDA, *supra* note 1, ¶ 531, at 187.

65. *See* FTC v. Staples, Inc., 970 F. Supp. 1066, 1074 (D.D.C. 1997) ("[T]he general question is 'whether two products can be used for the same purpose, and if so, whether and to what extent purchasers are willing to substitute one for the other.'" (citing Hayden Pub. Co. v. Cox Broad. Corp., 730 F.2d 64, 70 n.8 (2d Cir. 1984))).

66. ABA SECTION ON ANTITRUST LAW, *supra* note 6, at 505. Similar reasoning appears in other sources:

> The [*du Pont*] Court's product market inquiry into reasonable interchangeability of use or the cross-elasticity of demand between the product itself and substitutes for it subsumes both the functional interchangeability of products and the actual propensity of buyers to switch from product A to product B in response to changes in price.

Joshua A. Newberg, *Antitrust for the Economy of Ideas: The Logic of Technology Markets*, 14 HARV. J.L. & TECH. 83, 89 (2000) (internal quotation marks omitted).

67. *See, e.g.*, U.S. Anchor Mfg., Inc. v. Rule Indus., Inc., 7 F.3d 986, 995–99 (11th Cir. 1993) (finding brand-name anchors functionally interchangeable with generic anchors, but finding that there was insufficient evidence of demand substitutability between them); United States v. Archer-Daniels-Midland Co., 866 F.2d 242, 246 (8th Cir. 1988) (accepting finding that sugar and high fructose corn syrup are functionally interchangeable, but concluding that "they are not reasonably interchangeable because of the price differential between the two products"); United States v. Charles Pfizer & Co., 246 F. Supp. 464, 468 n.3 (E.D.N.Y. 1965) ("While a finding of functional interchangeability must precede that of reasonable (reactive) interchangeability, it is not determinative. For products to be classified in the same market they must be both functionally and reasonably interchangeable.").

for consumers to be able to switch between two products, and serves as a useful heuristic filter to identify possibly competing products. Thus, in the standard formulation of the reasonable interchangeability inquiry, functional interchangeability is considered first.[68]

a. Functional Interchangeability

When a product can be used for only one purpose, the functional interchangeability inquiry is relatively straightforward: another product either serves the same purpose or it does not. For products that can be used for multiple purposes, however, there does not appear to be a bright-line test for functional interchangeability. On the one hand, "functional interchangeability does not require complete identity of use."[69] For example, in *United States v. E.I. du Pont de Nemours & Co.*, a finding that cellophane "has to meet competition from other materials in every one of its uses" was sufficient for the Supreme Court to conclude that "a very considerable degree of functional interchangeability exists between these products,"[70] even though no single material was a significant competitor to cellophane in all of cellophane's uses.[71] On the other hand, it may sometimes be proper to draw a product market boundary that distinguishes a group of buyers who are interested in a product only for certain purposes.[72] For example, the Seventh Circuit upheld a product market definition that included sales of new components for automotive electrical units to rebuilders who used them in production-line work, but excluded such sales to rebuilders who used them in custom or retail work.[73]

Given the indeterminacy that arises when there is competition with respect to some but not all of the purposes served by the defendant's product, it should be noted that the functional interchangeability inquiry is neither intended nor suited to resolve these complexities. It seems prudent in such cases to stop at identifying the group of products that are separately functionally interchangeable with the de-

---

68. *See Pfizer*, 246 F. Supp. at 468. The court stated:

> To determine whether [products] are in competition in a particular industry it is first necessary to decide whether they can be used for the same purpose — whether they are functionally interchangeable; and functional interchangeability does not require complete identity of use. Having found one or more products functionally interchangeable with [the product] in a particular use, the next question to be resolved is one of purchaser reaction — the willingness or readiness to substitute one for the other.

Id. (citation omitted).

69. *Id*.

70. 351 U.S. 377, 399 (1956).

71. *See id*. at 407 (showing market shares of different wrapping materials for various end uses of cellophane).

72. *See infra* text accompanying notes 86–120 (describing price discrimination markets).

73. *See* Avnet, Inc. v. FTC, 511 F.2d 70, 77–79 (7th Cir. 1975) (Stevens, J.).

fendant's product in each of its relevant uses,[74] and to defer the ultimate question of which products are functionally interchangeable for purposes of defining the relevant product market until the propensity of purchasers to switch can be examined.[75]

b. Propensity to Switch

The inquiry into the propensity of purchasers to switch between products is directed to "whether buyers would respond to a significant increase in the price of *A* [from the competitive level to a supracompetitive level] by so shifting to product *B* as to make that price increase unprofitable to the *A* producers."[76] Observed shifts between products,[77] correlation in the prices or price movements of products,[78] or "the factors that normally determine the choice or preference of the user"[79] demonstrate a willingness to make such a shift between products.

c. Product and Price Differentiation

Courts often define product markets broadly enough to encompass differences that are material in the minds of buyers.[80] Even substantial differences in product features may "wash out," either when a particular product has both wanted and unwanted features or when different buyers have opposite preferences for a particular feature.[81]

Notwithstanding any differences in price and features between two products, if preferences with respect to such factors show that consumers are willing to switch between them, then a court will find that the products are reasonably interchangeable.[82] Generally, "a price differential, even a substantial one, is irrelevant for purposes of determining reasonable interchangeability."[83] This is because price dif-

74. *See supra* note 68 (describing functional interchangeability inquiry as directed to finding "one or more products functionally interchangeable with [the product] in a particular use").

75. *See infra* text accompanying notes 104–108.

76. AREEDA, *supra* note 1, ¶ 562, at 258.

77. *See id*.

78. *See id*.

79. *Pfizer*, 246 F. Supp. at 468.

80. *See, e.g.*, United States v. E.I. du Pont de Nemours & Co., 351 U.S. 377 (1956) (various flexible packaging materials); United States v. Cont'l Can Co., 378 U.S. 441, 456–57 (1964) (glass jars and metal cans); Cable Holdings v. Home Video, Inc., 825 F.2d 1559, 1563 (11th Cir. 1987) (cable television, satellite television, videocassettes, and free broadcast television); FTC v. PPG Indus., 798 F.2d 1500, 1504–06 (D.C. Cir. 1986) (glass and plastic aircraft transparencies).

81. *See supra* note 80.

82. *See* ABA SECTION OF ANTITRUST LAW, *supra* note 6, at 508–16 (reviewing cases involving differences in product type, differences in grade or quality, price differences and trends, and differences in product condition or availability).

83. United States v. Archer-Daniels-Midland Co., 866 F.2d 242, 246 (8th Cir. 1988).

ferentials between functionally interchangeable products, absent a structural barrier to entry into the product market, are usually offset by differences in quality or other preferred attributes, thereby allowing the prices of more and less expensive products to constrain one another.[84] Courts have been particularly reluctant to define product markets based on differences in price or quality where a group of functionally interchangeable products forms a continuous spectrum of choices for consumers.[85]

d. Price Discrimination Markets

When user preferences regarding product characteristics vary enough to raise the possibility of price discrimination, this may justify the delineation of additional, narrower markets around groups of "captive" or "inframarginal" buyers to whom a significant price increase could be profitably targeted.[86] While such a "price discrimination market" is predicated on the theory that price discrimination against the captive buyers is possible, its valid use is not limited to cases involving an actual or alleged practice of price discrimination.[87] For purposes of market share/market power analysis, a price discrimination market stands on equal footing with any other relevant product market.[88]

To succeed with a price discrimination strategy, a seller must be able to identify and discriminate in price against a group of buyers who would not switch to other products, or find other sources, in sufficient numbers to make a "small but significant and nontransitory" price increase unprofitable.[89] In particular, other customers who can

84. *See* AREEDA, *supra* note 1, ¶ 563, at 267–68.

85. *See, e.g.*, *In re* Super Premium Ice Cream Distrib. Antitrust Litig., 691 F. Supp. 1262, 1268 (N.D. Cal. 1988) ("Courts have repeatedly rejected efforts to define markets by price variances or product quality variances. Such distinctions are economically meaningless where the differences are actually a *spectrum* of price and quality differences." (emphasis in original) (citations omitted)); *but see* United States v. Gillette Co., 828 F. Supp. 78, 83 (D.D.C. 1993) (defining a market for *premium* writing instruments in a retail price range between $50 and $400); Keyte, *supra* note 47, at 722 (describing *Gillette* as "arguably breath[ing] some life back into carving out submarkets along a continuous price continuum").

86. Commentators have likened groups of captive buyers to the "distinct customers" referred to as one of the *Brown Shoe* indicia. *See, e.g.*, Jonathan B. Baker, *Stepping Out in an Old Brown Shoe: In Qualified Praise of Submarkets*, 68 ANTITRUST L.J. 203, 207–08 & 208 n.20 (2000).

87. *See* 15 U.S.C. § 13(a) (1985) (prohibiting seller from "discriminat[ing] in price between different purchasers of commodities of like grade and quality" when such discrimination adversely affects competition).

88. *See, e.g.*, United States v. Eastman Kodak Co., 63 F.3d 95, 106–07 (2d Cir. 1995) (considering a price discrimination market proposed to resolve a monopolization claim with a consent decree); HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.12 (defining price discrimination markets for use in merger review).

89. *See id*.

buy at a lower price must not be able to engage widely in arbitrage; i.e., purchasing the product for resale to disfavored buyers.[90]

The courts have recognized the ability to price-discriminate as relevant evidence of market power,[91] and have acknowledged support for price discrimination markets in the agency guidelines and in academic commentary.[92] Thus far, however, they have provided only scattered precedent for a price discrimination approach to market definition.[93]

For example, in *U.S. Anchor Manufacturing., Inc. v. Rule Industries, Inc.*,[94] an Eleventh Circuit case, U.S. Anchor alleged that Rule had attempted to monopolize a market for fluke anchors that encompassed generic and economy anchors as well as Rule's exclusive "Danforth" brand anchors.[95] Rule argued that the relevant product market consisted of generic and economy anchors only.[96] At trial, the district court denied Rule's motion for a directed verdict, and the jury found Rule liable on the attempted monopolization claim.[97] On appeal, the Eleventh Circuit considered whether U.S. Anchor had intro-

---

90. *See id*.; Pitofsky, *supra* note 13, at 1814.

91. *See, e.g.*, U.S. Steel Corp. v. Fortner Enters., Inc., 429 U.S. 610, 617 (1977) ("[I]f, as some economists have suggested, the purpose of a tie-in is often to facilitate price discrimination, such evidence would imply the existence of power that a free market would not tolerate."); Eastman Kodak Co. v. Image Technical Servs., Inc., 504 U.S. 451, 475–78 (1992) (citing Kodak's ability to price-discriminate against unsophisticated, small-volume, and locked-in customers as supporting Image's allegations of market power); Coal Exps. Ass'n of U.S., Inc. v. United States, 745 F.2d 76, 91 (D.C. Cir. 1984) ("[I]t is well established that the ability of a firm to price discriminate is an indicator of significant monopoly power.").

This principle is not uncontroversial. Some commentators have recently argued that the practice of "economic price discrimination," in which the packaging of items is used to "meter" differing consumer valuations of different products, is consistent with vigorous competition and therefore does not imply market power. *See* Benjamin Klein & John Shepard Wiley Jr., *Competitive Price Discrimination As An Antitrust Justification for Intellectual Property Refusals to Deal*, 70 ANTITRUST L.J. 599, 624–29 (2003); Michael E. Levine, *Price Discrimination Without Market Power*, 19 YALE J. ON REG. 1, 8–21 (2002). *But see* Jonathan Baker, *Competitive Price Discrimination: The Exercise of Market Power Without Anticompetitive Effects*, 70 ANTITRUST L.J. 643, 649–54 (2003) (replying to Klein and Wiley). These commentators, however, have neither questioned the practice of defining price discrimination markets, nor suggested that it is unnecessary to direct antitrust scrutiny to the possible anticompetitive exercise of market power against identifiable groups of inframarginal consumers to whom a price increase could be profitably directed.

92. *See* United States v. Eastman Kodak Co., 63 F.3d 95, 106–07 (2d Cir. 1995) (citing AREEDA, *supra* note 1, ¶ 534d, at 183–85; Pitofsky, *supra* note 13).

93. *See* LAWRENCE A. SULLIVAN, HANDBOOK OF THE LAW OF ANTITRUST § 17, at 62 (1977) (noting that the Supreme Court has never explicitly articulated a price discrimination approach to market definition).

94. *See* U.S. Anchor Mfg., Inc. v. Rule Indus., Inc., 7 F.3d 986, 998 (11th Cir. 1993) (noting that the ability to price-discriminate against a distinct group of customers "demonstrates the existence of market power with respect to that group" and "may, as a practical matter, remove the higher priced product from the broader market composed of its functional substitutes" (citations omitted)).

95. *See id*. at 989–91.

96. *See id*. at 991.

97. *See id*. at 992.

duced sufficient evidence to raise a jury question on the inclusion of Danforth anchors in the relevant product market.[98] After examining the *Brown Shoe* indicia,[99] the court concluded that there was insufficient evidence for a reasonable juror to find significant cross-elasticities of demand and supply between Danforth and the less expensive anchors.[100] The court then observed that "[t]he fluke anchor industry presented the unusual circumstance of severe price discrimination" against consumers loyal to Danforth, and that this brand loyalty may have been sufficient to justify finding a separate market for the Danforth anchors.[101] The court noted that such a finding, without more, would not necessarily imply that Danforth anchors were to be excluded from the relevant product market.[102] In the absence of "demonstrable empirical evidence" of supply and demand substitution between Danforth and the other anchors, however, the court concluded as a matter of law that the Danforth anchors should have been excluded from the relevant product market.[103]

A clearer case for price discrimination markets is presented when consumer groupings are based not on brand loyalty or personal tastes, but on the buyers' utilities for the various purposes that a product may serve.[104] Accordingly, the agencies define a product market as "consisting of a particular use or uses by groups of buyers of the product" that could be profitably discriminated against by a hypothetical mo-

---

98. *See id*. at 994.

99. *See supra* note 60 and accompanying text.

100. *See U.S. Anchor Mfg*., 7 F.3d at 996–97.

101. *Id*. at 997.

102. *See id*. at 998.

103. *Id*. at 998–99.

104. *See* Keyte, *supra* note 47, at 741. Keyte observes that:

> Identifying inframarginal consumers becomes much more complex when . . . a consumer's reluctance to switch products reflects brand preferences or purely personal tastes rather than the utility of the product itself. In these circumstances some courts have found that it is unrealistic to attempt to define an inframarginal group of consumers around any particular product characteristic . . . .

Id. Such groupings are characteristic of markets for software products and other information goods in particular. As at least one court and numerous commentators have observed, intellectual property rights serve in part as legal guarantees of an owner's ability to price-discriminate based on end-use segments. *See, e.g.*, ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1449–50 (7th Cir. 1996); Yochai Benkler, *An Unhurried View of Private Ordering in Information Transactions*, 53 VAND. L. REV. 2063, 2067–72 (2000); James Boyle, *Cruel, Mean, or Lavish? Economic Analysis, Price Discrimination and Digital Intellectual Property*, 53 VAND. L. REV. 2007, 2027–35 (2000); Julie E. Cohen, *Copyright and the Perfect Curve*, 53 VAND. L. REV. 1799, 1801–08 (2000); William W. Fisher III, *Property and Contract on the Internet*, 73 CHI.-KENT L. REV. 1203, 1234–40 (1998); Wendy J. Gordon, *Intellectual Property as Price Discrimination: Implications for Contract*, 73 CHI.-KENT L. REV. 1367, 1369 (1998); Louis Kaplow, *The Patent-Antitrust Intersection: A Reappraisal*, 97 HARV. L. REV. 1813, 1878–81 (1984); Michael J. Meurer, *Price Discrimination, Personal Use and Piracy: Copyright Protection of Digital Works*, 45 BUFF. L. REV. 845, 877–80 (1997); Michael J. Meurer, *Copyright Law and Price Discrimination*, 23 CARDOZO L. REV. 55, 80–90 (2001).

nopolist.[105] Courts have most commonly defined price discrimination markets by identifying one or more segments of consumers, each associated with one or more of the product's distinct "end uses."[106] For example, in *Illinois ex rel. Hartigan v. Panhandle Eastern Pipe Line Co.*, a district court analyzing a monopolization claim against Panhandle reasoned that the natural gas market needed to be "narrowed by reference to the capabilities of different types of end-users to take advantage of either alternative fuel or energy conservation methods or both."[107] After a bench trial, the court found that residential and commercial end-users had "much more restricted" abilities to conserve their consumption of natural gas or switch to other fuels than industrial end-users, and concluded that sales of natural gas to residential and commercial end-users constituted the relevant product market.[108] Although Panhandle was shown to have market power in this market,[109] the court ultimately concluded that Panhandle's conduct in most instances did not constitute willful acquisition or maintenance of monopoly power.[110]

For an end use to serve as the basis for a price discrimination market, it must *specifically* account for some significant part of the consumer demand for the product. Such an end use therefore must be complete, meaningful, and well-defined in the eyes of consumers, and must not be functionally interchangeable with any other end use or combination of end uses. For example, in *Nobel Scientific Industries v. Beckman Instruments, Inc.*,[111] the defendant Beckman was one of several companies that made blood analyzing machines and reagents.[112] Nobel alleged that Beckman had monopolized or attempted to monopolize the market for machines capable of performing seven particular tests simultaneously on a "stat" (high priority) basis, as well as the market for reagents to be used in such machines.[113] On Beckman's motion for summary judgment, the court rejected Nobel's market definition, citing uncontradicted evidence that the need to perform the seven specified stat tests simultaneously on one machine was not a complete, meaningful, and well-defined end use in the eyes of hospitals and laboratories.[114] Expert witnesses testified that hospitals base decisions to purchase analyzing machines on the cost and availability

105. HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.12.
106. *See* Keyte, *supra* note 47, at 740–41.
107. Illinois *ex rel*. Hartigan v. Panhandle E. Pipe Line Co., 730 F. Supp. 826, 900 (C.D. Ill. 1990).
108. *Id*.
109. *See id*. at 902–06.
110. *See id*. at 910.
111. 670 F. Supp. 1313 (D. Md. 1986).
112. *See id*. at 1315–16.
113. *See id*. at 1317–19.
114. *See id*. at 1319.

of the reagents and of other services needed to run an analyzer[115] and the need to perform "routine" (normal priority) tests and tests for other chemicals.[116] The evidence also showed that Beckman's machine was functionally interchangeable with other individual analyzers and combinations of analyzers for the purpose of performing the seven specified tests,[117] and that Beckman's reagents were functionally interchangeable with reagents sold by others for conducting the tests on Beckman's and other companies' machines.[118] The court concluded that it would be "overly restrictive" to define the product market by attributing consumer demand specifically to the seven specified tests where "few, if any, of the analyzers available [were] specifically limited to doing the seven named tests"[119] and where consumers valued the analyzers and reagents for many other features and purposes.[120]

To summarize, a product that has multiple uses may be found to face competition in two or more relevant product markets, each involving a significant group of consumers who are specifically interested in some subset of uses. A precise definition of these markets, however, requires an equally precise characterization of a "use"; one will be supplied for software products in Part II.C.

e. Illustration: Product Differentiation and Price Discrimination in the Cellophane Case

In *United States v. E.I. du Pont de Nemours & Co.*, the government charged du Pont with monopolizing the manufacture and sale of cellophane in violation of Section 2 of the Sherman Act. The Supreme Court, on direct appeal, reviewed the district court's determination that the "relevant market for determining the extent of du Pont's market control" was not cellophane, but all flexible packaging materials.[121] Noting that physical characteristics do not necessarily serve to

---

115. *See id*.
116. *See id*. at 1321.
117. *See id*. at 1320.
118. *See id*. at 1320–22.
119. *Id*. at 1320.
120. *See id*. The court noted:

> Some analyzers are valued for the number of tests they can do, some for their speed, some for their cost, and some for other features. All of the machines, however, compete for the same contracts and business. Therefore, one cannot separate out the competition to sell reagents for only these seven tests. Reagent competition is for selling reagents for any of the tests that the machines can run.

Id.

121. 351 U.S. 377, 380 (1956). A different aspect of the Supreme Court's market definition analysis in the Supreme Court's market definition analysis in the "Cellophane case," namely the Court's approach to the calculation of demand cross-elasticity, has long been criticized, but is not materially relevant to the present discussion. For criticism of the "Cel-

distinguish one material from another for purposes of the market definition inquiry,[122] a 4–3 majority of the Court held that "[i]n determining the market under the Sherman Act, it is the use or uses to which the commodity is put that control."[123] Turning to the trial record, the Court noted differences among the physical characteristics and prices of cellophane and other flexible packaging materials, but found that cellophane "has to meet competition from other materials in every one of its uses" and that "a very considerable degree of functional interchangeability exists between these products."[124] In the case of Pliofilm, a more expensive alternative to cellophane, the Court found that its superior physical characteristics, which made it preferable for use in wrapping meat, "apparently offset cellophane's price advantage," thereby making the price of Pliofilm a constraint on the price of cellophane in the eyes of consumers.[125] The Court concluded that the relevant market "is composed of products that have reasonable interchangeability for the purposes for which they are produced — price, use and qualities considered."[126] Therefore, the relevant market included at least the packaging materials that were shown at trial to be functionally interchangeable with cellophane.[127] Given cellophane's "competition and interchangeability with other wrappings," du Pont was not liable for monopolization.[128]

The dissenters objected that du Pont, by monopolizing cellophane, could price-discriminate against certain end-use segments, such as buyers engaged in wrapping cigarettes, who required cellophane in part for properties that other flexible packaging materials did not have.[129] Commenting on the case, Robert Pitofsky answers that any such discrimination would have been defeated by arbitrage,[130] but observes that arbitrage opportunities in general do not follow immediately from a price differential:

> To be effective in the arbitrage business, the customers must know the identity of the other customers

---

lophane fallacy," see, for example, RICHARD POSNER, ANTITRUST LAW 128 (1976); Donald F. Turner, *Antitrust Policy and the Cellophane Case*, 70 HARV. L. REV. 281, 309 (1956).

122. 351 U.S. at 394.

123. *Id*. at 395–96.

124. *Id*. at 399.

125. *Id*. at 399–400.

126. *Id*. at 404.

127. *See id*.

128. *Id*.

129. *See id*. at 424–25 (Warren, C.J., dissenting).

130. *See* Pitofsky, *supra* note 13, at 1814; *accord* SBC Comm'ns Inc. v. FCC, 56 F.3d 1484, 1493–94 (D.C. Cir. 1995) (affirming the FCC's determination that relevant market was for all interexchange service rather than interexchange service to cellular customers, *inter alia*, because of the California attorney general's finding that "arbitrage activities would defeat any attempt by AT&T/McCaw to raise cellular interexchange rates above existing levels").

> who are being discriminated against, undertake the expenses of buying, storing, reselling, and reshipping the product, and do so at a scale that would make an impact on the discriminating sellers. Finally, the arbitrageurs must be willing to go into this new business at whatever investment level is required, knowing that they could be frustrated completely in their initiative if the seller abandons its discriminatory scheme.[131]

f. Quality Restraints

Non-price competition among functionally interchangeable products, particularly those "in which differences in features are important (and in which improvement is possible)," is especially vital to consumers in the software industry.[132] To the extent that such non-price competition is recognized as a concern of antitrust law,[133] the practice of defining markets based on price discrimination should account for the ability of a seller with market power to discriminate against a particular end-use segment by reducing the quality of the product signifi-

---

131. Pitofsky, *supra* note 13, at 1848–49.

132. Glen Holly Entm't, Inc. v. Tektronix, Inc., 100 F. Supp. 2d 1073, 1081 (C.D. Cal. 1999); *see also* Robert H. Lande, *Consumer Choice as the Ultimate Goal of Antitrust*, 62 U. PITT. L. REV. 503, 517 (2001) (noting that nonprice competition is most likely to be necessary to protect consumer choice "with respect to certain kinds of intellectual property, some of which can play a competitive role only in an environment of organizational independence").

Microsoft chairman Bill Gates has acknowledged that non-price competition can predominate over price competition in a software market:

> With intellectual property, the upfront costs are what it's all about . . . . Say a piece of software costs $10 million to create and the marginal costs, because it's going to be distributed electronically, are basically zero. Once the costs of development have been recouped, every single additional unit is pure profit. But if someone comes along with a significantly superior product, your demand can literally almost drop to zero.

Alan Murray, *Intellectual Property: Old Rules Don't Apply*, WALL ST. J., Aug. 23, 2001, at A1 (quoting Gates).

133. *See, e.g.*, Standard Oil Co. v. United States, 221 U.S. 1, 52 (1911) (identifying "[t]he danger of deterioration in quality of the monopolized article" as one of the three "evils" of monopoly); Weit v. Cont'l Ill. Nat'l Bank & Trust Co., 641 F.2d 457, 477 (7th Cir. 1981) (citing C-O-Two Fire Equip. Co. v. United States, 197 F.2d 489, 493 (9th Cir. 1952)) ("[I]n an oligopoly . . . non-price competition is valuable, and anything tending to standardize non-price terms harms competition."); In re Microsoft Corp. Antitrust Litig., 127 F. Supp. 2d 702, 711 (D. Md. 2001) ("Since businesses compete through both lower prices and superior performance, a firm's stifling of innovative products would cause antitrust injury."); Douglas H. Ginsburg, *Nonprice Competition*, 38 ANTITRUST BULL. 83, 83 n.1 (1993) (citing Catalano, Inc. v. Target Sales, Inc., 446 U.S. 463 (1980)) (noting Court's "appreciat[ion] that an agreement to fix a nonprice term of trade is analytically indistinguishable from an agreement to fix price").

cantly below a competitive level with respect to that end use only.[134] Since a reduction in the quality of a product constitutes an increase in the product's quality-adjusted price,[135] such a practice is equivalent to quality-adjusted price discrimination against the end-use segment in question. A price discrimination market should be defined accordingly. Even though this form of discrimination against a group of buyers may not be cognizable as price discrimination under the Robinson-Patman Act,[136] it provides an appropriate criterion for identifying a market in which non-price competition may be harmed by the exercise of market power.[137]

Quality-adjusted price discrimination markets of this kind are more likely to involve information goods than the physical goods that typically have been the subjects of price discrimination theories of market definition. The basic fact that physical goods are fully characterized by their physical properties is likely to constrain a seller's ability to reduce quality with respect to only one end use. For example, Robert Pitofsky's observation that arbitrage would defeat price discrimination in *du Pont*[138] implicitly relies on the reasonable assumption that any attempt to modify the physical properties of cellophane (e.g., heat-sealability, printability, clarity, tear and burst strength, and resistance to oils)[139] to make it less useful for wrapping cigarettes

---

134. A seller with market power may find it profitable to reduce product quality in the eyes of a captive group of consumers if the seller can thereby reduce production costs or, more generally, if the seller's interests are adverse in some way to the consumers' preferences.

135. *See* Timpinaro v. SEC, 2 F.3d 453, 457 (D.C. Cir. 1993) (Ginsburg, J.) (citing Ginsburg, *supra* note 133) (noting that non-price discounts "have the same pro-competitive effect as a price discount").

Quality and price may not be fully commensurable in quantitative terms. *See* Lande, *supra* note 132, at 516 (noting that "[s]ome elements of non-price competition might be captured through use of the concept of 'quality-adjusted price,'" but that "'quality-adjusted price' may be a difficult concept to apply in concrete situations where the non-price components of competition are particularly important, or where they take subtle or complex forms"). The point here is a qualitative one; i.e., that a reduction in the quality of a product raises the same antitrust concerns as a corresponding increase in the product's price. *See id*.; Peter J. Hammer, *Questioning Traditional Antitrust Presumptions: Price and Non-Price Competition in Hospital Markets*, 32 U. MICH. J.L. REFORM 727, 759 n.85 (1999) ("It is simply wrong, however, to conclude that there are no antitrust issues when one observes constant prices in the face of falling quality.").

136. *See* 15 U.S.C. § 13(a) (1997) (prohibiting price discrimination between purchasers of commodities of "like grade [and] quality").

137. *See supra* notes 87–88 and accompanying text; *cf.* United States v. Microsoft Corp., 253 F.3d 34, 83 (D.C. Cir. 2001) (en banc) (noting that plaintiffs needed to establish that "Microsoft would have the power to raise the price of its browser above, or reduce the quality of its browser below, the competitive level" to show the existence of entry barriers into the "browser market"); Miller v. Ind. Hosp., 814 F. Supp. 1254 (W.D. Pa. 1992) ("A defendant possesses monopoly power if it has the ability to change the competitive variables of a product to the disadvantage of consumers without causing effective competitors to enter the relevant market." (citation omitted)).

138. *See supra* text accompanying note 130.

139. *See du Pont*, 351 U.S. at 411.

would also reduce its quality with respect to wrapping various foods.[140]

In contrast, digital information goods are highly susceptible to a vendor's legal and technological controls over individual end uses, as demonstrated by the burgeoning field of digital rights management.[141] An arbitrageur might be able to defeat these controls technologically by altering the product so that it supports new uses or better supports existing uses,[142] but license terms usually prohibit such activities.[143] More generally, intellectual property rights powerfully reinforce a vendor's ability to price-discriminate against particular end uses.[144]

An emerging body of literature has recognized the ability of intellectual property licensors to restrict the market output of quality through a related but different practice known as "quality discrimination."[145] Quality discrimination occurs whenever a seller "discriminate[s] among consumers with different tastes for quality . . . by offering an array of qualities."[146] Except in situations involving a reduction in quality targeted at a specific end-use segment, quality discrimination appears to be similar to product differentiation in its implications for product market definition.[147] Quality-adjusted price discrimination and quality discrimination are distinct factual predicates, and only the former is proposed here as a possible basis for

---

140. To the extent that du Pont's cellophane monopoly was derived in part from patent exclusivity, *see id*. at 382–84, du Pont was also constrained from modifying the physical properties of cellophane by the scope of the relevant patent claims.

141. *See, e.g.*, Meurer, *supra* note 104, at 878 & n.160 (1997) ("Digital technology will directly facilitate price discrimination by allowing low cost metering of the usage of digital works."). For a survey of digital rights management technologies, see, for example, BILL ROSENBLATT ET AL., DIGITAL RIGHTS MANAGEMENT: BUSINESS AND TECHNOLOGY (2001); Stephen M. Kramarsky, *Copyright Enforcement in the Internet Age: The Law and Technology of Digital Rights Management*, 11 DEPAUL-LCA J. ART & ENT. L. & POL'Y 1 (2001).

142. *See, e.g.*, Universal City Studios, Inc. v. Reimerdes, 111 F. Supp. 2d 294, 308 (S.D.N.Y. 2000), *aff'd sub nom.* Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2d Cir. 2001) (describing the DVD-descrambling software utility known as "DeCSS"); *cf.* Meurer, *supra* note 104, at 86 (2001) (describing arbitrage against software quality discrimination by modifying software to supply missing functionalities). See *infra* text accompanying notes 145–148 for an explanation of quality discrimination.

143. *See* Meurer, *supra* note 104, at 86 (noting that software modifications for the purpose of arbitrage "violate the derivative rights of the copyright owner"); Darren C. Baker, Note, ProCD v. Zeidenberg*: Commercial Reality, Flexibility in Contract Formation, and Notions of Manifested Assent in the Arena of Shrinkwrap Licenses*, 92 NW. U. L. REV. 379, 391 (1997) (describing prohibitions on reverse engineering and modification as "standard or typical terms" in shrinkwrap licenses).

144. *See supra* note 104.

145. *See, e.g.*, Meurer, *supra* note 104, at 73–74; Hal R. Varian, *Versioning Information Goods*, *in*, INTERNET PUBLISHING AND BEYOND: ECONOMICS OF DIGITAL INFORMATION AND INTELLECTUAL PROPERTY (Brian Kahin & Hal R. Varian eds, 1997).

146. JEAN TIROLE, THE THEORY OF INDUSTRIAL ORGANIZATION 149–50 (1988).

147. *See generally* Shubha Ghosh, *Gray Markets in Cyberspace*, 7 J. INTELL. PROP. L. 1 (1999) (characterizing practices of quality discrimination in cyberspace as forms of product differentiation).

product market definition.[148] The literature on quality discrimination in intellectual property licensing is worth noting in the present context, however, because it highlights another common situation in the software industry wherein licensors have wide discretion over product quality.

2. Supply Substitutability

Although courts have tended to focus more on demand substitutability than on supply substitutability in determining the relevant product market,[149] supply substitutability considerations have been found to be materially relevant in enough cases that it would be erroneous to define a market on the basis of demand substitutability alone.[150]

Recall that the goal in defining a product market is to identify "*producers* which, because of the similarity of their products, have the ability — actual or potential — to take significant amounts of business away from each other."[151] The demand substitutability inquiry, on the other hand, identifies *products* that have reasonable interchangeability of use.[152] The supply substitutability inquiry serves to complete the analysis by identifying firms that are actual or potential producers of these products.

The supply substitutability inquiry focuses on "[c]ross-elasticity of supply, or production flexibility among sellers"[153] or, equivalently, "the ability of firms in a given line of commerce to turn their productive facilities toward the production of commodities in another line because of similarities in technology between them."[154] As with cross-elasticity of demand, the cross-elasticity of supply between two prod-

148. *See supra* text accompanying notes 80–85 for a discussion of the analysis of product differentiation as it relates to product market definition.

149. *See* ABA SECTION ON ANTITRUST LAW, *supra* note 6, at 516.

150. *See, e.g.*, Rebel Oil Co. v. Atl. Richfield Co., 51 F.3d 1421, 1436 (9th Cir. 1995) ("[D]efining a market on the basis of demand considerations alone is erroneous . . . . A reasonable market definition must also be based on 'supply elasticity.'"); Virtual Maint., Inc. v. Prime Computer Inc., 11 F.3d 660, 664 (6th Cir. 1993) ("Defining a market, or 'submarket,' on the basis of demand considerations alone is erroneous because such an approach fails to consider the supply side of the market."); In re Mun. Bond Reporting Antitrust Litig., 672 F.2d 436, 441 (5th Cir. 1982) (finding that plaintiff's proposed market definition "fails to give due accord to the significance of elasticity of supply"); United States v. Empire Gas Corp., 537 F.2d 296, 303 (8th Cir. 1976) ("The cross-elasticity of supply would seem to be as important as the demand factor in determining relevant product market.").

151. SmithKline Corp. v. Eli Lilly & Co., 575 F.2d 1056, 1063 (3d Cir. 1978) (emphasis added).

152. *See supra* text accompanying notes 61–68.

153. Kaiser Aluminum & Chem. Corp. v. FTC, 652 F.2d 1324, 1330 (7th Cir. 1981).

154. Twin City Sportservice, Inc. v. Charles O. Finley & Co., 512 F.2d 1264, 1271 (9th Cir. 1975).

ucts is a precise quantity,[155] but the issue has usually been formulated less precisely in antitrust decisions.[156] Courts have placed products in the same product market if they could be produced interchangeably from the same production facilities,[157] but have declined to do so where there were sufficient barriers, such as large research and development costs,[158] to make a shift in production unprofitable.[159]

The 1992 Horizontal Merger Guidelines provide a theoretically accurate, if difficult to administer, approach to the analysis of supply substitution.[160] Specifically, the Guidelines include within the relevant market all firms that currently produce or sell the identified products and any other firms whose "inclusion would more accurately reflect probable supply responses."[161] Supply response is deemed probable if it is "likely to occur within one year and without the expenditure of significant sunk costs of entry and exit, in response to a 'small but significant and nontransitory' price increase."[162] In determining the likelihood of supply response, the agencies will consider "technological capability," as well as any "difficulties in achieving product acceptance, distribution, or production."[163]

C. Software Product Markets

1. Consumer Demand for Software Products

Software is code.[164] Software is *used* by installing and running it on a system, thereby producing *system behavior*. Consumers desire to

---

155. The cross-elasticity of supply is the percentage change in supply for one good attributable to a percentage change in the price of another good. *See* AREEDA, *supra* note 1, ¶ 507, at 108.

156. *See generally* ABA SECTION ON ANTITRUST LAW, *supra* note 6, at 517–19 & nn. 102–08 (reviewing cases).

157. *See, e.g.*, Yoder Bros. v. Cal.-Fl. Plant Corp., 537 F.2d 1347, 1367–68 (5th Cir. 1976) (finding that growers could easily switch production from other flowers to chrysanthemums); Telex Corp. v. IBM, 510 F.2d 894, 916 (10th Cir. 1975) (finding that manufacturers could switch production from non-IBM-compatible peripherals to IBM-compatible peripherals).

158. *See, e.g.*, United States v. Ivaco, Inc., 704 F. Supp. 1409, 1417 (W.D. Mich. 1989); In re B.A.T. Indus., 104 F.T.C. 852, 932 (1984).

159. *See, e.g.*, U.S. Anchor Mfg. v. Rule Indus., 7 F.3d 986, 997 (11th Cir. 1993); Ansell, Inc. v. Schmid Lab., 757 F. Supp. 467, 475–76 (D.N.J. 1991).

160. *See* Pitofsky, *supra* note 13, at 1860–61 (opining that the Guidelines "handle these supply substitution questions well," but noting that they fail to explain "what sort of evidence properly can be relied upon to establish supply substitution").

161. HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.32.

162. *Id*.

163. *Id*.

164. Software code may be *source code* or *object code*. Source code is "[h]uman-readable program statements written by a programmer or developer in a high-level or assembly language that are not directly readable by a computer" and "needs to be compiled into object code before it can be executed by a computer." MICROSOFT CORP., MICROSOFT COMPUTER DICTIONARY 418 (1999). Object code is "[t]he code, generated by a compiler or

use software for producing system behavior that supports various *tasks* (which are sometimes also referred to as *functionalities*). System behavior of the kind that supports a task occurs in the form of an *interaction* between the user and the system.

In response to these consumer desires for user-system interactions, producers market *software products*. A software product is defined by reference to accompanying software and *documentation*, and consists essentially of the legal rights and technological capabilities necessary to install and run the software on a system according to the documentation; it does not include any of the software or documentation itself, in which the vendor retains copyright.[165] The documentation describes legal and technological *preconditions* for using the software product, and tasks that may be supported by using the software product subject to such preconditions.

The use of a software product may require a system to run not only the software that accompanies the software product, but also other software that has previously been installed on the system. For example, the use of *application software*[166] requires the use of preinstalled *operating system software*.[167] It may therefore be a precondition for using one software product that another software product has previously been acquired and its accompanying software preinstalled on the system. In such a case, the two products are recognized as complements, not substitutes.[168] Any required preinstalled software is

---

an assembler, that was translated from the source code of a program," usually by the software developer or vendor prior to the distribution of the software that accompanies a software product. *Id*. at 317. Throughout this Article, the term "code" will generally be used to refer to object code.

165. *See infra* text accompanying note 445. For example, in a suit by Microsoft for copyright infringement, a defendant who had legitimately obtained a single user license for Microsoft Office could not validly claim to own the accompanying software code itself. *See* Microsoft Corp. v. Software Wholesale Club, Inc., 129 F. Supp. 2d 995, 1007–08 (S.D. Tex. 2000) (holding defendant liable for copyright infringement because, *inter alia*, defendant was unable to show that counterfeit copies had good chain of title).

This distinction between a software product and its accompanying software is a fundamental one. To preserve this distinction, this Article will employ the somewhat unwieldy terminology "software that accompanies a software product" throughout this Article. As a side benefit, the distinction also permits the discussion of the "sale" or "purchase" of a software product without implying that the accompanying software is sold or purchased.

166. An application is a "software program[ ] . . . that perform[s] specific user-oriented tasks." United States v. Microsoft Corp., 84 F. Supp. 2d 9, 12 (D.D.C. 1999).

167. An operating system ("OS") is a "software program that controls the allocation and use of computer resources (such as processing time, main memory space, disk space, and input/output channels)." *Id*.

168. This distinction is especially significant in the context of product market definition. A properly defined relevant market includes goods that are reasonably close substitutes for one another, but not complementary goods. *See* AREEDA, *supra* note 1, ¶ 565a–b, at 329–32.

referred to as *platform software*. Platform software is often installed on the system's hard drive in the form of *library files*.[169]

In some systems, platform software may be installed in multiple *layers*, where the use of each layer requires the use of the previously installed layers. The term *middleware* refers to platform software that itself requires other platform software in this way.

A software product specifies which software is to run on the system when the software product is used, even though not all such software necessarily accompanies the software product.[170] For example, a program may instruct the system to run specific routines in preinstalled platform software by using the conventions, or *calls*, defined in the platform software's *programming interface* (which is part of the documentation accompanying the platform software).[171] A software product is said to *support a task* if it specifies which software is to run on the system in order to produce behavior that supports the task, and (subject to its documented preconditions) confers sufficient legal rights and technological capabilities to do so.

In summary, a consumer may wish to acquire a software product because of some of the tasks it supports, or because of its complementarity to some other desired software products that require its acquisition as a precondition. This Article will use the terms *consumer purpose* and *end use* interchangeably and generically to refer to any such supported task or complementarity relationship.

Some examples of preconditions for the use of and consumer purposes served by the software products Microsoft Windows, Netscape Navigator for Windows, and Microsoft Word for Windows are:

Microsoft Windows — Precondition: The system is an Intel-based personal computer ("PC"). Consumer purposes: Platform software for Netscape Navigator for Windows; platform software for Microsoft Word for Windows; view the contents of directories on the system's hard drive.

Netscape Navigator for Windows — Precondition: Microsoft Windows software is preinstalled. Consumer purposes: Platform software for Web-based applications; perform Web transactions.

---

169. *See* FREE ON-LINE DICTIONARY OF COMPUTING, *at* http://foldoc.doc.ic.ac.uk/foldoc/ (last visited Dec. 4, 2004) (defining "library" as "[a] collection of subroutines and functions stored in one or more files, usually in compiled form, for linking with other programs").

170. Specifically, the software that accompanies a software product may make procedure calls to previously installed software, as when an application makes calls to the application programming interfaces of an operating system. *See id*. For a more detailed description of this process, *see infra* Part III.B.2; *see also* JOHN R. LEVINE, LINKERS & LOADERS 187–227 (2000) (describing linking of code using shared libraries, including Windows dynamically linked libraries). *Cf.* United States v. Microsoft Corp., 84 F. Supp. 2d 9, 50 (D.D.C. 1999) (describing "knitting" together of different software layers).

171. *See* FREE ON-LINE DICTIONARY OF COMPUTING, *supra* note 169 (defining "application program interface" as "[t]he interface (calling conventions) by which an application program accesses operating system and other services").

Microsoft Word for Windows — Preconditions: Microsoft Windows software is preinstalled; document is a file in Word .DOC format. Consumer purposes: Edit document.

2. Tasks and Essential Use Cases

The procedure for defining product markets described above, particularly price discrimination markets, calls for the consideration of end uses that may be degraded or withheld at a vendor's discretion. In determining a relevant product market in which a particular software product competes, it is therefore necessary to identify any consumer purposes that may be cognizable as captive end-use segments under a price discrimination theory. Any such consumer purpose must be characterized in terms that are complete, meaningful, and well-defined from the user's perspective, so that the resulting end-use segment represents a well-defined group of users who are interested in the software product for that consumer purpose.[172] The characterization of a consumer purpose should also be in terms that are simple, general, abstract, technology-free, and implementation-independent, so that the corresponding end-use segment avoids drawing false distinctions between different technological approaches to supporting what is essentially the same task from a user's perspective.[173]

Computer scientists and software engineers have considerable experience with the specification of software use, and have developed many models and methodologies to describe software behavior at various levels of abstraction.[174] Of particular relevance for present purposes is a highly abstract software modeling construct known as an *essential use case*, which was introduced in Larry Constantine and Lucy Lockwood's groundbreaking software engineering textbook, *Software for Use*:[175]

> An *essential use case* is a structured narrative, expressed in the language of the application domain

172. *See supra* text accompanying note 111.

173. Such variations in technological implementation are more appropriately analyzed as a kind of product differentiation rather than a division of a product market into end-use segments. *See* AREEDA, *supra* note 1, ¶ 563a, at 310 ("Products are differentiated when many buyers regard them as different even though the products still perform the same essential function.").

174. For highly formal models of software behavior, see, for example, JOHN COOKE, CONSTRUCTING CORRECT SOFTWARE (1998); D.C. INCE, AN INTRODUCTION TO DISCRETE MATHEMATICS, FORMAL SYSTEM SPECIFICATION, AND Z (1993). For highly abstract models, see, for example, STEPHEN M. MCMENAMIN & JOHN F. PALMER, ESSENTIAL SYSTEMS ANALYSIS (1984). For intermediate approaches, see, for example, ALI BEHFOROOZ & FREDERICK J. HUDSON, SOFTWARE ENGINEERING FUNDAMENTALS (1996); GRADY BOOCH, OBJECT-ORIENTED DESIGN WITH APPLICATIONS (1991).

175. LARRY L. CONSTANTINE & LUCY A.D. LOCKWOOD, SOFTWARE FOR USE: A PRACTICAL GUIDE TO THE MODELS AND METHODS OF USAGE-CENTERED DESIGN (1999).

> and of users, comprising a simplified, generalized, abstract, technology-free and implementation-independent description of one task or interaction that is complete, meaningful, and well-defined from the point of view of users in some role or role in relation to a system and that embodies the purpose or intentions underlying the interaction.[176]

Given this definition and the foregoing discussion, no background in software engineering is needed to appreciate that the concept of an essential use case is applicable to product market definition as a way of characterizing the tasks supported by a software product.

A full overview of the techniques necessary to construct essential use cases is presented in Chapter 5 of *Software for Use* and is beyond the scope of this article.[177] A concrete example taken from that chapter, however, will serve to illustrate the suitability of essential use cases for identifying cognizable end-use segments.

Figure 1: A Use Case for the Task of Getting Cash From an ATM[178]

gettingCash	
User Action	**System Response**
insert card	read magnetic stripe
enter PIN	request PIN
	verify PIN
press key	display transaction option menu
press key	display account menu
enter amount	prompt for amount
press key	display amount
take card	return card
take cash	dispense cash

176. *Id*. at 103 (emphasis in original).
177. *Id*. at 97–123.
178. *Id*. at 102.

To understand what is meant by an essential use case, it is helpful to be familiar with the more general concept of a *use case*. Invented in the late 1960's by software engineer Ivar Jacobson,[179] use cases are a methodology for narrating user-system interactions commonly used by software developers (and increasingly, customers) to describe required system behavior.[180] In particular, use cases play a central role in object-oriented software design techniques using the Unified Modeling Language.[181] Figure 1 is a use case depicting the process of getting cash from an automatic teller machine ("ATM").

Recently, proponents of use case-based design methods have emphasized the importance of describing the user-system interaction at a high level of abstraction, avoiding any implementation-specific language that assumes particular choices on the part of the designer regarding the system's behavior and user interface.[182] For example, the use case in Figure 1 presupposes that the user identification mechanism is a card with a magnetic stripe, that the system provides information to the user via a visual display, and that the user provides information to the system via a keypad.[183] The use case limits the choices available to the designer as to how the system will support the task of getting cash from an ATM.[184] Unless tasks are specified in an implementation-independent form, software designers may be constrained from choosing the design that best serves the purposes of the user.[185]

Figure 2 presents an essential use case for the same task. Note that all implementation-specific language has been abstracted away, and the narrative of the user-system interaction is expressed solely from the perspective of a user who has assumed a particular role in relation to a system (an account holder) and has a particular purpose in using the system (getting cash). This essential use case fully captures "the purpose or intentions underlying the interaction": that is, for any system to support the task of getting cash from an ATM, it is necessary and sufficient for the system to support each of the interaction steps shown in Figure 2. Beyond the requirement that the system serve this specified user purpose, the essential use case does not con-

---

179. *See* ALISTAIR COCKBURN, WRITING EFFECTIVE USE CASES, at xx (2001).

180. *See id*. at 1–3 (describing a use case as "a contract between the stakeholders of a system about its behavior"); DARYL KULAK & EAMONN GUINEY, USE CASES: REQUIREMENTS IN CONTEXT 50 (2000) (suggesting that use cases be used in requests for proposals to specify desired software behavior).

181. *See, e.g.*, JIM ARLOW & ILA NEUSTADT, UML AND THE UNIFIED PROCESS: PRACTICAL OBJECT-ORIENTED ANALYSIS AND DESIGN (2001); GRADY BOOCH ET AL., THE UNIFIED MODELING LANGUAGE USER GUIDE (1998).

182. *See, e.g.*, CONSTANTINE & LOCKWOOD, *supra* note 175, at 102–03; KULAK & GUINEY, *supra* note 180, at 36–37.

183. *See* CONSTANTINE & LOCKWOOD, *supra* note 175, at 103.

184. *See id*.

185. *See id*. at 102–03.

strain the design and implementation of the system in any way.[186] For example, user identification may be implemented with voice recognition, thumbprint analysis, or a retinal scan; and choices might be offered through voice synthesis, or conveniently arranged so that the customer's usual withdrawal amount is listed most prominently.[187] Thus, compared with a use case for a given task, the corresponding essential use case "is closer to a purely problem-oriented, rather than solution-oriented, view of the task . . . ."[188] Specifically, an essential use case provides "an abstract, idealized, and technology-free description of a problem with minimal intrusion of assumptions about particular solutions."[189]

Figure 2: An Essential Use Case for the Task of Getting Cash From an ATM[190]

gettingCash	
User Intention	**System Responsibility**
identify self	verify identity
choose	offer choices
take cash	dispense cash

Essential use cases abstract away implementation details and specific technological solutions, not purposes and problems that are of intrinsic interest to the user.[191] For example, the essential use case of Figure 2 includes the user self-identification step because such a step is necessary to determine that a user is the owner of an account before permitting the user to withdraw cash from that account.

Importantly, an essential use case must describe enough of the interaction to be "complete, meaningful, and well-defined" from the user's perspective. It should address at least the following questions regarding the user's purpose and intentions in interacting with the system:

- What does the user need to be able to do?

186. *See id*.
187. *See id*.
188. *Id*.
189. *Id*. at 108.
190. *Id*. at 105.
191. *See id*. at 105 (stating that the essential use case in Figure 2 "includes only those steps that are essential and of intrinsic interest to the user").

- What capabilities are required to support whatever the user is trying to accomplish?
- What information will the user need to examine, create, or change?
- What will the user need to be informed of by the system?
- What will the user need to inform the system about?[192]

Because an essential use case completely captures a user purpose without restricting design, it provides antitrust analysis with an appropriate criterion for deciding whether two software products "can be used for the same purpose"[193] and which software products, in supporting a task that is also supported by the defendant's product, may thereby compete with the defendant's product within the corresponding end-use segment.

Of course, these inquiries into functional interchangeability and end-use segments represent only some of the relevant considerations for the delineation of a product market. Product and price differentiation among functionally interchangeable products, and supply substitution by current producers and probable market entrants should also be examined. Software engineering can provide frameworks for these analyses as well.

3. Competitive Variables, Metrics and Preconditions

Within the parameters of an essential use case, a task can be implemented by a virtually unlimited variety of design approaches, thereby giving rise to significant differentiation among functionally interchangeable software products. Even with respect to differentiated products, however, if shifts in demand or correlations between prices or price movements are observed,[194] then such products should be seen as reasonably interchangeable. Otherwise, the product market definition analysis should examine the products' "competitive variables"[195] — i.e., "the factors that normally determine the choice or preference of the user."[196]

Software engineers have considerable experience with the measurement of software performance and quality, and have identified those metrics that play significant roles in a user's evaluation of soft-

192. This is a simplified version of a list of questions appearing in CONSTANTINE & LOCKWOOD, *supra* note 175, at 116.

193. United States v. Charles Pfizer & Co., 246 F. Supp. 464, 468 (E.D.N.Y. 1965).

194. *See supra* notes 77–78 and accompanying text.

195. *See* HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.11 (1992) (stating that the DOJ and FTC will consider buyer and seller "response to relative changes in price or other competitive variables" in defining the relevant product market).

196. *Pfizer*, 246 F. Supp. at 468.

ware products.[197] Constantine and Lockwood have identified two categories of software metrics that are relevant to the differentiation of software products: "preference metrics" (based on subjective user evaluations of user-system interactions) and "performance metrics" (based on controlled, systematic testing of user-system interactions).[198]

Table 1: Aspects of Software Use That May Be Measured By Preference and Performance Metrics

Preference metrics[199]	**Performance metrics**[200]
Affect	Completeness
Efficiency (subjective)	Correctness
Helpfulness	Effectiveness
Control	Efficiency (objective)
Learnability	Proficiency
	Productiveness

A full survey of software metrics is presented in Chapters 17 and 18 of *Software for Use* and is beyond the scope of this Article.[201] Table 1 lists some of the many aspects of software use for which metrics have been developed. Of course, not all of these aspects and metrics will factor into every market definition analysis, and other metrics may be found relevant to the valuations of software products.[202] Courts and parties should examine the relevant evidence to identify those particular aspects of software use that are material to "the choice or preference of the user" for a software product to serve a particular

197. *See, e.g.*, TOM GILB, SOFTWARE METRICS (1977); STEPHEN H. KAN, METRICS AND MODELS IN SOFTWARE QUALITY ENGINEERING (2002).

198. *See* CONSTANTINE & LOCKWOOD, *supra* note 175, at 419. A third category of metrics, referred to as either "predictive metrics" or "design metrics," is used by software developers to evaluate prototypes of software products early in the development process, rather than finished software products in consumer markets. *See id*. at 423–42.

199. *See* CONSTANTINE & LOCKWOOD, *supra* note 175, at 421.

200. *See id*. at 454.

201. *Id*. at 417–62.

202. For example, the emerging theory of "value sensitive design" has identified a number of ethical values that may be considered in evaluating alternative software designs. *See* Batya Friedman et al., Value Sensitive Design: Theory and Methods, 4–6 (June 2003) (listing human welfare, ownership and property, privacy, freedom from bias, universal usability, trust, autonomy, informed consent, accountability, identity, calmness, and environmental sustainability as "frequently implicated values that we suggest have a distinctive claim on resources in the design process"), *at* http://www.ischool.washington.edu/vsd/vsd-theory-methods-draft-june2003.pdf. These values expand on traditional usability concerns, *inter alia*, by accounting for the interests of stakeholders in the design of a computer system who do not use the system themselves (e.g., patients whose histories are stored on a medical records system, or citizens who are impacted by their fellow citizens' use of a voting machine). *See id*. at 3–4.

purpose, and then to identify those software products that effectively compete with the defendant's product with respect to these aspects of software use.[203]

Software products that support the same task (as defined by an essential use case) may also vary with respect to their preconditions. In deciding whether two software products are reasonably interchangeable, courts and parties should determine whether any overlap between their preconditions is broad enough to permit effective competition between them.[204] Where two software products have mutually exclusive preconditions (e.g., incompatible system hardware requirements), they should be deemed both reasonably and functionally non-interchangeable, even though both may support the same task.

Relevant documentary evidence for the analysis of metrics and preconditions may be found in software product marketing studies, published reviews of the software products, other descriptions of user experiences with software products, bug reports, software patches, documentation accompanying software products, and the general computer science, software engineering, and software consumer literature. Testimonial evidence from computer science and software engineering experts and software vendors, developers and users, and demonstrative evidence (e.g., verifying system behavior in the presence of the court) may also be relevant.

4. Price Discrimination Markets

As we have seen, essential use cases can be used to identify end-use segments that are possible targets for (quality-adjusted) price discrimination.[205] If a particular end use specifically accounts for some significant part of the consumer demand for the defendant's software product,[206] and a hypothetical monopolist of software products supporting the end use would have the legal and technological ability to reduce the quality of its products significantly below a competitive level with respect to that end use only,[207] then the respective end-use segment should be deemed a relevant product market. Discrimination against that end-use segment would be expected to succeed, as a would-be arbitrageur would be unable to alter the monopolist's software product so as to restore the product's quality to a competitive level with respect to the end use.[208]

---

203. *See supra* text accompanying note 59.
204. *See supra* text accompanying note 59.
205. *See supra* Part II.C.2.
206. *See supra* text accompanying note 111.
207. *See supra* note 134 and accompanying text.
208. *See supra* notes 141–144 and accompanying text.

To prove the technological feasibility of such a strategy, a party could develop and demonstrate a prototype software product that removes or significantly degrades the ability of the defendant's product to support the relevant end use without affecting its performance with respect to all other end uses. Other relevant evidence would address the presence or absence of functional or logical relationships among the software product's various end uses that would impede discrimination against only one of them. For example, it may be the case that every service provided by a particular software product's platform software is relied upon by multiple complementary application software products, so that any degradation in the platform software relied upon by one complementary product would entail a similar degradation in service to other complementary products. Also, tasks (as represented by essential use cases) may be interrelated in various ways that preclude their independent degradation, including by classification, by extension, by composition, or by affinity.[209]

5. Supply Substitutability

Once a group of products having reasonable interchangeability of use has been determined, current producers of these products can be identified for inclusion in the relevant product market. Under the Merger Guidelines' approach to supply substitution, the product market should also include firms that would probably begin producing these products in response to a price increase, taking into account "significant sunk costs of entry and exit," "technological capability," and "difficulties in achieving product acceptance, distribution or production."[210]

In the software industry, which is generally characterized by high fixed costs (mostly in research and development to design the product) and near-zero marginal costs of production and distribution,[211] there are two principal structural barriers to entry. They are the legal and technological impediments to designing a product that is functionally and reasonably interchangeable with the products already in the market ("incumbent products") and difficulties in achieving product acceptance. In general, the need to commit significant sunk costs

209. *See* CONSTANTINE & LOCKWOOD, *supra* note 175, at 109. For a formal description of these relationships, see *id*. at 109–15.

210. HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.32.

211. *See, e.g.*, Ronald A. Cass & Keith N. Hylton, *Antitrust Intent*, 74 S. CAL. L. REV. 657, 725 (2001) ("Virtually all the costs of production are in the design of the software and therefore independent of the amount sold, so that marginal costs are virtually zero.") (quoting Declaration of Kenneth J. Arrow, United States v. Microsoft Corp., 980 F. Supp. 537 (D.D.C. 1997) (No. 94-1564)); Einer Elhauge, *Why Above-Cost Price Cuts to Drive Out Entrants Are Not Predatory — And the Implications for Defining Costs and Market Power*, 112 YALE L.J. 681, 710 (2003) (describing software's marginal cost of production as "near zero").

of research and development will deter entry unless it is expected that such costs would be recouped through supracompetitive pricing.[212] Some other examples of structural impediments are:

a. Interference From Preinstalled Software

An incumbent product that supports a particular task may be designed to interfere with the ability of other software products to support the same task. This may be a rational strategy if the incumbent product has a sufficiently large installed base that its software has often been preinstalled on a system even when the user has chosen a different software product to support the task.

In particular, when preinstalled software is designed to support a task despite a user's choice of a different software product for that purpose, the effect of such preinstalled software is to prevent all subsequently acquired software products from supporting the task according to their documented specifications.[213] If sufficiently frequent and severe, the resulting frustration of the user's intentions may significantly diminish the product quality of the chosen software product, conferring a structural advantage on the preinstalled software product.

b. Proprietary Platform Software

Where incumbent products serve the purpose of preinstalling platform software for one or more complementary products, an entrant will typically be able to design a product that serves the same purpose only if it is legally permissible and technologically possible to emulate the programming interfaces provided by the platform software. Even when this is possible, it can be a risky, costly, and difficult undertaking.[214]

c. Exclusionary Preconditions

An incumbent product may have preconditions that require the system to use proprietary complementary technologies, and may therefore be incompatible with preconditions of other software products that support the same task. Depending on the existence and extent of any remaining overlap among the products' preconditions, proprietary technological requirements can warrant a determination of func-

212. *See* HORIZONTAL MERGER GUIDELINES, *supra* note 42, § 1.32 (excluding from the product market any firm that would face "[a] significant sunk cost . . . which would not be recouped within one year of the commencement of the supply response").

213. *See supra* text accompanying note 171.

214. *See* United States v. Microsoft Corp., 84 F. Supp. 2d 9, 22 (D.D.C. 1999) (describing IBM's unsuccessful effort to clone the Windows platform in 1994 at a cost of "tens of millions of dollars").

tional or reasonable non-interchangeability. For example, a word processing software product may require as a precondition for use that any input file be in a certain proprietary document format. A would-be competitor may be legally and technologically precluded from developing another word processing software product that works with the same document format.

D. A First Principles Approach to Market Definition

This section serves to summarize the procedure this Article has described for defining a relevant product market.

1. Define the Defendant's Product

A software product is defined by reference to accompanying software and documentation, and consists essentially of the necessary legal rights and technological capabilities to install and run the software on a system according to the documentation.[215]

2. List Relevant Consumer Purposes for the Defendant's Product

The list should consist of consumer purposes for the defendant's product that are relevant to the challenged practice and are complete, meaningful, and well-defined from the user's perspective.[216] Consumer purposes may include (1) tasks supported by the defendant's product, and (2) the satisfaction of preconditions for running other software products by the acquisition of the defendant's product and the preinstallation of its accompanying platform software.[217]

The list need not include all consumer purposes served by the defendant's product. Since "functional interchangeability does not require complete identity of use,"[218] the list need not be comprehensive, but might be limited to the product's primary end use or uses. Alternatively, it may consist of a single end use that could be targeted for price discrimination where the challenged practice has been alleged to affect competition among products serving that end use.[219] Such price discrimination is possible, for example, if it specifically accounts for some significant part of the consumer demand for the product,[220] and if a hypothetical monopolist would have the ability to discriminate

215. *See supra* Part II.C.1.
216. *See supra* text accompanying notes 111–171.
217. *See supra* text accompanying note 171.
218. United States v. Charles Pfizer & Co., 246 F. Supp. 464, 468 (E.D.N.Y. 1965).
219. *See supra* text accompanying notes 104–106.
220. *See supra* text accompanying note 106.

against the end use by reducing the quality of the product significantly below a competitive level with respect to that end use only.[221]

3. Represent Any Relevant Tasks as Essential Use Cases

Each relevant task should be characterized in the form of an essential use case: a structured narrative, expressed in the language of the application domain and of users, comprising a simplified, generalized, abstract, technology-free, and implementation-independent description of the user-system interaction that supports the task.[222]

4. Identify Products That Are Functionally Interchangeable with the Defendant's Product for the Relevant Consumer Purposes

A product should be deemed functionally interchangeable with the defendant's product if it serves any of the consumer purposes identified in step 2, as characterized in step 3.[223]

5. List Relevant Competitive Variables

Competitive variables include material preference and performance metrics with respect to each relevant task, and material preconditions for using the defendant's product.[224] A factor is material if it would normally determine the user's choice or preference of a software product for the relevant end use.[225]

6. Identify Products That Are Reasonably Interchangeable with the Defendant's Product for the Relevant Consumer Purposes

The reasonable interchangeability analysis begins with a provisional market consisting of the defendant's product, and proceeds by iteratively extending the boundaries of the provisional market to include additional products that are reasonably interchangeable with the products already found to be in the provisional market.[226] A product identified in step 4 as functionally interchangeable with the defendant's product is reasonably interchangeable if, given consumer preferences with respect to the competitive variables identified in step 5, consumers would respond to a quality-adjusted price increase above a competitive level by a hypothetical monopolist of the provisional market by switching to the functionally interchangeable product in

221. *See supra* text accompanying note 134.
222. *See supra* Part II.C.1.
223. *See supra* notes 205–210 and accompanying text.
224. *See supra* Part II.C.3.
225. *See supra* text accompanying notes 195–196.
226. *See supra* text accompanying notes 52–57.

sufficient volume so as to make such a price increase unprofitable.[227] This iterative process should continue until no more reasonably interchangeable products can be added to the provisional market.[228]

7. Identify Structural Barriers to Entry

The software product market definition procedure concludes by identifying producers that could respond to a price increase above a competitive level by a hypothetical monopolist of the provisional market by making and selling any of the incumbent products identified in step 6, or a reasonably interchangeable new product, in sufficient volume so as to make such a price increase unprofitable.[229] This analysis should account for structural barriers to entry into the product market that may arise from the technological difficulty of designing a functionally and reasonably interchangeable new product, such as exclusionary preconditions, proprietary platform software, and interference from preinstalled software,[230] as well as difficulties in achieving product acceptance.

E. Well-Functioning Software Product Markets

Quality competition in a software product market has been described here as being located within a space defined by preference and performance metrics.[231] To be precise, the quality of a software product is measured by the degree to which the system on which the product is used is seen to perform its responsibilities satisfactorily in response to user intentions, as specified by the essential use case(s) embodying the user purpose(s) that the software product was purchased to support. This description of software product quality coincides with economic conceptions of consumer welfare. A software product's full economic cost includes the time and effort required to use the product for the purpose(s) for which it was purchased. Ease of use is an economic benefit to consumers.

One of the world's leading authorities on human-centric product design, Donald A. Norman, refers to the fundamental problems presented by difficult-to-use systems as "the gulf of execution" and "the gulf of evaluation."[232] A gulf of execution exists when the user must expend considerable effort before the system will respond to the user's intentions as expected.[233] A gulf of evaluation exists when the

227. *See supra* note 65 and accompanying text.
228. *See supra* text accompanying note 54.
229. *See supra* Part II.C.5.
230. *See supra* Part II.C.5.
231. *See supra* notes 195–204 and accompanying text.
232. DONALD A. NORMAN, THE DESIGN OF EVERYDAY THINGS 49–52 (1988).
233. *See id*. at 51.

user must expend considerable effort before the user is able to understand that that the system has responded to the user's intentions as expected.[234] Drawing on examples from door handles to VCRs, Norman asserts that a failure to address these two problems is responsible for design defects that impede the usability of a vast number of everyday devices.[235] Applying Norman's concepts to software products, it is apparent that innovation focused on preference and performance metrics embodies the pursuit of ease of use.[236] When, as the result of innovation, a software product offers improvements with respect to one or more preference or performance metrics, the system on which the product is used will exhibit reductions in the gulfs of execution and evaluation. Consumers can perceive this ease of use and then make choices accordingly as to which software products to use for their desired purposes.[237]

The ability of the vendor of a software product to respond to consumer demand for quality through design innovation resides in the vendor's freedom to choose the code (which may include platform software) that the system executes in fulfillment of its responsibilities whenever a consumer chooses to use the software product for any of the user purposes for which it is sold. This Article will use the term "well-functioning" to describe a software product market in which every software vendor has this freedom to innovate in response to consumer demand. This freedom is necessary not only for consumer demand to drive competition among software vendors to improve product quality, but also for software development to proceed according to standard industry practice.

Table 2 outlines the standard development process for a software product. Within this framework, the specification of user purposes with essential use cases corresponds to the requirements-gathering stage (documentation of functions to be performed that are "in the

234. *See id*. at 51–52.

235. *See id*. at 34–104 *passim*.

236. *See* MICHAEL L. DERTOUZOS, THE UNFINISHED REVOLUTION 20–24 (2001) (arguing that the first step toward human-centric computing will be the implementation of natural interaction with machines, wherein "machine actions [will] match our human intent" and where the system will "let us carry out our intent at our level and with little effort").

237. This description of consumer behavior assumes that the vendor will provide the consumer with accurate information about the software product's performance for the consumer's desired purposes, which is not always the case. *See* 2 L.J. KUTTEN, COMPUTER SOFTWARE: PROTECTION/LIABILITY/LAW/FORMS § 8.01[2], at 8-6 (2001) ("The goal of a salesman is to sell the software as quickly as possible. And unfortunately the product has to be sold whether or not it fits the user's needs."). To the extent that the choice of a software product may be made on the basis of material misinformation, however, the appropriate remedy would normally be furnished by commercial law rather than antitrust. *See generally* CEM KANER & DAVID L. PELS, BAD SOFTWARE: WHAT TO DO WHEN SOFTWARE FAILS (1998) (describing consumer's remedies under the Uniform Commercial Code for defective software products).

users' language and from the users' perspective").[238] By definition, the essential use cases that are the products of the requirements-gathering stage are independent of the software developer's design and implementation decisions.

Table 2: Activities in the Standard Software Development Process[239]

Stage	**Description**
Requirements gathering	Gather and document the functions that the application should perform for the users in the users' language and from the users' perspective.
Analysis	Build a logical solution that satisfies the requirements but does not necessarily take hardware constraints into account.
Design	Adapt the logical solution to satisfy system constraints.
Construction	Write, compile, debug, and test code in a programming environment.
Testing	Test code in a complete working system. Fix problems and obtain user acceptance.
Deployment	Deliver code and documentation. Install code on machines and train users.
Maintenance	Make changes to the working system to fix new problems and adapt to ongoing changes in technology and customer needs.

In subsequent stages, the developer is called upon to make increasingly specific decisions about the design and implementation of the software product, culminating in the creation of deliverable code and documentation. Each of these stages is predicated on the expectation that the developer will be free to choose the code that is to be executed when the software product is chosen. Thus, in the analysis and design stages, the developer expects that the logical solution being created (including the system's user interface as specified by an implementation-specific use case)[240] will be amenable to implementation through construction, testing, deployment, and maintenance. In the construction, testing, and deployment stages, the developer expects that the system being created and installed will respond to user requests by executing the code exactly as the developer has written and compiled it. Finally, in the maintenance stage, the developer expects that any fixes and adaptations to the code will be fully reflected in changes to the behavior of the system.

238. *See* KULAK & GUINEY, *supra* note 180, at 34–38 (explaining the role of "context-free" (i.e., implementation-independent) use cases in requirements gathering).

239. *See id*. at 5; *see also* Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 JURIMETRICS J. 33, 38 & n.8 (1987) (describing a similar "process of program creation" performed by software engineers).

240. *See supra* notes 179–190 and accompanying text.

The code that is executed when the software product is used may, at the developer's option, incorporate platform software that has been preinstalled on the system prior to the deployment stage.[241] Among the decisions to be made by the developer during the design stage is the determination of which platform software, if any, is assumed to have been preinstalled. During the construction and testing stages, the developer then has the option of incorporating some of this platform software by reference into the code that is to be executed when the software product is used — i.e., by making calls to the appropriate programming interfaces in the platform software. It bears emphasizing that the developer's freedom to choose which code is to be executed is not affected in any way by the provenance of platform software. Although the developer typically is not the author of the platform software, the developer defines the context and scope of its incorporation into the software, and thereby retains ultimate responsibility and control over which code is to be executed when the software product is used.

The foregoing discussion demonstrates the need for careful consideration of the distinctive role and structure of the software development process as a basis for competition in a well-functioning software product market. Antitrust analysis should specifically recognize the potential anticompetitive effects of any legal or technological impediments to a developer's freedom to choose which code is to be executed when a consumer chooses to use the developer's product. If sufficiently severe, such impediments should be regarded as a barrier to entry in defining the relevant product market.[242]

The antitrust analysis of alleged injuries to competition in software product markets requires a rigorous examination of another distinctive characteristic of software products. Namely, the sale of a software product does not confer fee simple title to the accompanying software code, but instead allocates various limited legal rights between the vendor and the consumer under the terms of a software license.[243] As I will explain in Part III, statutory and judicially-created limitations on copyright exclusivity serve in this context to support well-functioning software product markets.

III. COPYRIGHT EXCLUSIVITY IN SOFTWARE PRODUCT MARKETS

Software licensing, like other contracting activities, may constitute exclusionary practices that trigger antitrust scrutiny. In such cases, antitrust liability can turn on whether the practices in question

241. *See supra* note 170 and accompanying text.
242. *See supra* text accompanying note 214.
243. *See infra* Part III.

are deemed to be a legitimate exercise of rights that were lawfully acquired under the federal copyright laws.[244] This issue is especially significant in software product markets because mass market software licenses frequently purport to extend the vendor's rights beyond the scope of the copyright grant.[245]

The Copyright Act of 1976[246] defines the scope of copyright protection in two respects. First, it identifies the elements of a work that are eligible subject matter for copyright protection. Second, it identifies and limits the exclusive and exclusionary rights that are granted to the owner of copyright in a work with respect to these elements. These rights and limitations constitute the basic framework within which authors and other rights holders control and exploit their works.

Increasingly, however, software products are marketed to consumers under terms and conditions that purport to extend the vendor's rights beyond the scope defined by the Copyright Act.[247] As many commentators have noted, these transactional practices may have the effect of overriding the balance of public policy interests embodied in the federal copyright statute.[248] Of particular concern in the antitrust

244. *See, e.g.*, United States v. Loew's, Inc., 371 U.S. 38, 47–48 (1962) (condemning the block booking of separately copyrighted motion pictures for television exhibition as a tying arrangement under § 1 of the Sherman Act); In re Indep. Serv. Orgs. Antitrust Litig., 203 F.3d 1322, 1329 (Fed. Cir. 2000) (rejecting unilateral refusal to license claim for lack of evidence that copyrights were obtained by unlawful means or were used to gain monopoly power beyond the statutory copyright granted by Congress); Data Gen. Corp. v. Grumman Sys. Support Corp., 36 F.3d 1147, 1187 (1st Cir. 1994) ("[W]hile exclusionary conduct can include a monopolist's unilateral refusal to license a copyright, an author's desire to exclude others from use of its copyrighted work is a presumptively valid business justification for any immediate harm to consumers."); Montgomery County Ass'n of Realtors, Inc. v. Realty Photo Master Corp., 878 F. Supp. 804, 817 & n.23 (D. Md. 1995), *aff'd*, 91 F.3d 132 (4th Cir. 1996) (holding that a copyright owner's refusal to license the copying of its database was authorized by § 106 of the Copyright Act, and was therefore a "legitimate business purpose" negating the claim of concerted refusal to deal claim); *see generally* 1 HOVENKAMP ET AL., IP AND ANTITRUST: AN ANALYSIS OF ANTITRUST PRINCIPLES APPLIED TO INTELLECTUAL PROPERTY LAW § 13.3, at 13-10 (2003) (stating that antitrust law generally imposes a duty to license intellectual property only in cases where "an intellectual property owner has sought to expand the scope of its right beyond what the intellectual property laws grant it").

245. *See, e.g.*, ProCD, Inc. v. Zeidenberg, 86 F.3d 1447 (7th Cir. 1996) (holding that a "shrinkwrap" license agreement that overrode limitations on the copyright owner's rights under the Copyright Act was enforceable under Wisconsin contract law).

246. 17 U.S.C. §§ 101–1332 (2004).

247. *See ProCD*, 86 F.3d 1447.

248. There is an extensive literature on the preemption of state contract law by federal copyright law and the enforceability of shrinkwrap and clickwrap license agreements; i.e., standard form license agreements that assert that the act of opening a box, or downloading files, containing software signifies the consumer's assent to the license terms. For commentary on preemption, see, for example, Maureen A. O'Rourke, *Drawing the Boundary Between Copyright and Contract: Copyright Preemption of Software License Terms*, 45 DUKE L.J. 479 (1995). For commentary on shrinkwrap and clickwrap agreements, see, for example, Garry L. Founds, Note, *Shrinkwrap and Clickwrap Agreements: 2B or Not 2B?*, 52 FED. COMM. L.J. 99 (1999); Dennis S. Karjala, *Federal Preemption of Shrinkwrap and On-Line Licenses*, 22 U. DAYTON L. REV. 511 (1997); Mark A. Lemley, *Intellectual Property and Shrinkwrap Licenses*, 68 S. CAL. L. REV. 1239 (1995); Mark A. Lemley, *Shrinkwraps*

context is the balance between the constitutional purpose to "promote the Progress of Science . . . by securing for limited Times to Authors . . . the exclusive Right to their respective Writings"[249] and the Sherman Act's "general prohibition on unreasonable restraints of trade."[250]

In addressing this balance, antitrust doctrine draws a fundamental distinction between trade restraints that inhere in the rights conferred by the intellectual property laws, and trade restraints that result from the contractual or technological exploitation of those rights. While intellectual property rights themselves may restrain competition by subjecting competing suppliers to civil liability for certain kinds of productive activities (i.e., those involving infringement), the legitimate acquisition and enforcement of rights under the intellectual property laws are generally not subject to antitrust scrutiny.[251] Transactions involving intellectual property rights, however, may be subject to antitrust challenge based on the owner's conduct in exploiting those rights through contractual or technological means.[252] In *United States v. Microsoft Corp*., the D.C. Circuit provided perhaps the most vivid statement of this distinction.[253] In affirming the district court's dismissal of Microsoft's copyright counterclaim, the appellate court reasoned that Microsoft's contention that the exercise of lawfully acquired intellectual property rights cannot give rise to antitrust liability "is no more correct than the proposition that use of one's personal property, such as a baseball bat, cannot give rise to tort liability."[254]

A comprehensive survey of the antitrust analysis of specific intellectual property transactional practices is beyond the scope of this Article.[255] The more limited purposes of Part III are to clarify the role

---

in Cyberspace, 35 JURIMETRICS J. 311 (1995); Apik Minassian, *The Death of Copyright: Enforceability of Shrinkwrap Licensing Agreements*, 45 UCLA L. REV. 569 (1997).

249. U.S. CONST. art. 1, § 8, cl. 8.

250. Times-Picayune Pub. Co. v. U.S., 345 U.S. 594, 614 (1953).

251. *See generally* HOVENKAMP ET AL., *supra* note 244 §§ 13.1–13.2, at 13-2 to 13-10 (2003) (explaining that the antitrust laws generally permit the lawful owners of intellectual property to enforce and refuse to license their rights); *cf.* Feb. 22, 2000 A.M. Session Trial Transcript at 28, United States v. Microsoft. Corp., 87 F. Supp. 2d 30 (D.D.C. 2000) (No. 98-1232) (statement by Jackson, J.) (stating that Copyright Act "gives you, for all practical purposes, fee simple control over that code"), *available at* 2000 WL 215541.

252. *See, e.g.*, Interstate Circuit v. U.S., 306 U.S. 208, 230 (1939) ("An agreement illegal because it suppresses competition is not any less so because the competitive article is copyrighted."); Data General Corp. v. Grumman Sys. Support Corp., 36 F.3d 1147, 1185 n.63 (1st Cir. 1994) ("It is in any event well settled that concerted and contractual behavior that threatens competition is not immune from antitrust inquiry simply because it involves the exercise of copyright privileges."); *cf.* Eastman Kodak Co. v. Image Technical Servs., 504 U.S. 451, 479 n.29 (1992) ("The Court has held many times that power gained through some natural and legal advantage such as a patent, copyright, or business acumen can give rise to liability if 'a seller exploits his dominant position in one market to expand his empire into the next.'" (citations omitted)).

253. 253 F.3d 34 (D.C. Cir. 2001).

254. *Id*. at 63.

255. The leading treatise in this area is HOVENKAMP ET AL., *supra* note 244.

of copyright in defining the legal rights that comprise a software product and to derive generally applicable principles for determining whether an alleged restraint on competition in software product markets exceeds the scope of the exclusionary rights granted by the Copyright Act. In Parts III.A and III.B, respectively, this Article will explain how the Copyright Act defines the scope of copyrightable expression in software and the scope of the copyright owner's substantive rights with respect to that copyrightable expression.

A. The Scope of Copyrightable Subject Matter

It is settled law that the Copyright Act extends protection at least to the software code[256] that accompanies a software product. Section 102(a) expressly provides copyright protection to "original works of authorship fixed in any tangible medium," including literary, musical, dramatic, choreographic, pictorial, graphic, sculptural, audiovisual, recorded audio, and architectural works.[257] A work is fixed in a tangible medium when its embodiment in a copy "is sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration."[258] Literary works are defined as "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia."[259]

Software code is expressed in "verbal or numerical symbols or indicia," and is therefore considered a "literary work" within the meaning of the Copyright Act.[260] The software code that accompanies a software product is fixed in a tangible medium insofar as it is distributed in a form that at least permits its reproduction in the random access memory ("RAM") of the user's computer.[261] Copyright law thus prohibits at least the unauthorized literal copying of software code associated with simple forms of software piracy, such as the duplication of software CD-ROMs or diskettes.

The Copyright Act is less clear, however, regarding the eligibility of nonliteral elements of software for copyright protection. Section 102(b) of the Copyright Act expressly denies copyright protection to

---

256. *See supra* note 164.

257. 17 U.S.C. § 102(a) (2004).

258. *Id.* § 101.

259. *Id.*

260. *See* Atari Games Corp. v. Nintendo of Am., Inc., 975 F.2d 832, 838–39 (Fed. Cir. 1992) (finding that computer programs fall within the terms of the Copyright Act, but noting that § 102(b) limits protection to "the expression adopted by the programmer" and excludes "the actual processes or methods embodied in the program").

261. *See* MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 517–18 (9th Cir. 1993) (finding that copies of software in random access memory can be "perceived, reproduced, or otherwise communicated for a period of more than transitory duration" within the meaning of § 101).

ideas, procedures, processes, systems, methods of operation, concepts, principles, and discoveries.[262] In general, this provision serves to codify the longstanding rule that "copyright does not protect ideas, but only expression of ideas,"[263] also known as the "idea/expression dichotomy."[264] The purpose of the idea/expression dichotomy is to balance competing constitutional values by placing the original expressions of authors within the scope of copyrightable subject matter while preserving the public's First Amendment interest in the free communication of ideas.[265]

Although the idea/expression dichotomy is well-settled as a matter of principle,[266] it has been difficult for courts to apply in practice, particularly in cases involving functional works.[267] In such cases, the aim is to preserve "the balance between competition and protection reflected in the patent and copyright laws."[268] With respect to software, the idea/expression dichotomy stands for the principle that copyright in a computer program should not prevent another developer from writing different code that performs the same functions as the copyrighted program when executed. Thus, § 102(b) serves "to make clear that the expression adopted by the programmer is the copyrightable element in a computer program, and that the actual processes or methods embodied in the program are not within the scope of copyright law."[269]

---

262. 17 U.S.C. § 102(b) (2004).

263. Whelan Assocs. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1234 (3d Cir. 1986).

264. *See* H.R. REP. NO. 94-1476, at 57 (1976), *reprinted in* 1976 U.S.C.C.A.N. 5659, 5670 ("Section 102(b) in no way enlarges or contracts the scope of copyright protection under the present law. Its purpose is to restate . . . that the basic dichotomy between expression and idea remains unchanged.").

265. *See* Harper & Row Publishers, Inc. v. Nation Enters., 471 U.S. 539, 556–57 (1985) (noting that the dichotomy "strike[s] a definitional balance between the First Amendment and the Copyright Act by permitting free communication of facts while still protecting an author's expression" (citation omitted)); Religious Tech. Ctr. v. Netcom On-Line Communications Servs., Inc., 907 F. Supp. 1361, 1377 (N.D. Cal. 1995) (noting that the dichotomy "balance[s] the important First Amendment rights with the constitutional authority for 'promoting the progress of the science and useful arts'" (citations omitted)).

266. *See* Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 350 (1991) (stating that the dichotomy "applies to all works of authorship"); Baker v. Selden, 101 U.S. 99 (1879).

267. *See, e.g.*, John Shepard Wiley, Jr., *Copyright at the School of Patent*, 58 U. CHI. L. REV. 119, 121–29 (1991) (arguing that the dichotomy has become an "incoherent" doctrine that "announces results but does not determine or justify them").

268. Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1253 (3d Cir. 1983) (quoting Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971)).

269. H.R. REP. NO. 94-1476, at 57 (1976), *reprinted in* 1976 U.S.C.C.A.N. 5659, 5670; S. REP. NO. 94-473, at 54 (1975). Several of the federal circuit courts have referred to this legislative history in discerning the purpose of § 102(b). *See, e.g.*, Gates Rubber Co. v. Bando Chem. Indus., 9 F.3d 823, 836–37 (10th Cir. 1993); Atari Games Corp. v. Nintendo of Am. Inc., 975 F.2d 832, 838–39 (Fed. Cir. 1992); Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 703 (2d Cir. 1992); M. Kramer Mfg. Co. v. Andrews, 783 F.2d 421, 434–35 (4th Cir. 1986); *Apple Computer*, 714 F.2d at 1252–53.

In effect, the § 102(b) inquiry requires courts to attach legal significance to the subtle distinction between a programmer's expression in the "verbal or numerical symbols or indicia" of a computer program[270] and the "processes or methods embodied in the program."[271] As a result, courts have historically taken widely divergent approaches toward copyright infringement cases in which the defendant has not literally copied the plaintiff's code, but has duplicated other elements of the plaintiff's software, such as its structure, sequence, organization, hardware interfaces, programming interfaces, user interfaces, and "look and feel."

1. *Whelan v. Jaslow*

A 1986 decision by the Third Circuit, *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*,[272] provided the first thorough analysis of this issue. In the case, the defendant Jaslow had hired Whelan to develop a custom computer program for dental laboratory record-keeping that required the particular platform software that had been preinstalled on Jaslow's machines.[273] Jaslow subsequently developed and marketed another program for use by other dental laboratories that served essentially the same function as Whelan's program, but was based on more widely available platform software.[274] Whelan sued Jaslow for copyright infringement. After a bench trial, the district court credited the testimony of Whelan's expert, who had found similarities between the file structures used, the screen outputs produced, and some of the subroutines called by both programs.[275] Based on these substantial similarities and Jaslow's prior access to Whelan's program, the district court inferred that Jaslow had actually copied the common elements from Whelan's program.[276] Noting that the copyrightable expression in a computer program includes not only its code, but also "the manner in which the program operates, controls and regulates the computer," the district court concluded that Jaslow had committed copyright infringement by copying expression from Whelan's program.[277]

On appeal to the Third Circuit, Jaslow argued that the two programs were not substantially similar as a matter of law, because the district court did not find any similarity between their versions of

270. 17 U.S.C. § 101.

271. H.R. REP. NO. 94-1476, at 57 (1976), *reprinted in* 1976 U.S.C.C.A.N. 5659, 5670; S. REP. NO. 94-473, at 54 (1975).

272. 797 F.2d 1222 (3d Cir. 1986).

273. *Id*. at 1225–26.

274. *Id*. at 1226.

275. Whelan Assocs. v. Jaslow Dental Lab., Inc., 609 F. Supp. 1307, 1321–22 (E.D. Pa. 1985).

276. *Id*.

277. *Id*. at 1320–22.

code.[278] Rejecting this argument, the appeals court drew an analogy between the "structures" of computer programs and the copyrightable "plot[s] or plot devices" of literary works,[279] and held that this analogy governed the scope of copyright in computer programs.[280] Applying the idea/expression dichotomy to software, the court concluded that the unprotectable idea of a program is simply "the purpose or function" of the program, and that the protectable expression in a computer program includes not only its code, but "everything that is not necessary to . . . [its] purpose or function."[281] In this analysis, the copyrightability of a software element turns on the availability of functionally equivalent alternatives. The court stated, "Where there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea."[282] The appellate court found that the purpose of Whelan's program was "the efficient organization of a dental laboratory,"[283] and that the particular "detailed structure" of Whelan's program, not being necessary to that purpose, was protectable expression.[284] Accordingly, the Third Circuit affirmed the district court's judgment of copyright infringement based on the substantial similarity between the "structure, sequence, and organization" of the two programs.[285]

Even though the elements of software structure at issue in *Whelan* were found to be copyrightable, the Third Circuit noted in a footnote that in other contexts, elements of software structure and organization might be found to be unprotectable ideas:

> We do not mean to imply that the idea or purpose behind *every* utilitarian or functional work will be precisely what it accomplishes, and that structure and organization will therefore always be part of the expression of such works. The idea or purpose behind a utilitarian work may be to accomplish a certain function *in a certain way*, and the structure or function of a program might be essential to that task. There is no suggestion in the record, however, that the purpose of the [Whelan] program was anything so refined; it

278. 797 F.2d at 1233.
279. *Id*. at 1234.
280. *Id*. at 1238.
281. *Id*. at 1236.
282. *Id*.
283. *Id*. at 1240; *see also id.* at 1236 n.28 ("the idea . . . was the efficient management of a dental laboratory"); *id*. at 1238 ("the purpose . . . was to aid in the business operations of a dental laboratory").
284. *Id*. at 1238–39.
285. *Id*. at 1248.

> was simply to run a dental laboratory in an efficient way.[286]

Despite this dictum, *Whelan* has generally been read as standing for the proposition that "only one 'idea,' in copyright law terms, underlies any computer program, and that once a separable idea can be identified, everything else must be expression."[287] As a result, the *Whelan* court's approach to the scope of copyrightable subject matter in software has been widely rejected by most courts outside the Third Circuit[288] and numerous commentators.[289] Even so, *Whelan* remains good law in the Third Circuit,[290] and continues to be cited elsewhere for the settled proposition that some non-literal elements of software structure, sequence, and organization may be copyrightable subject matter.[291]

2. *Computer Associates v. Altai*

In a 1992 case, *Computer Associates International, Inc. v. Altai, Inc.*, the Second Circuit took a contrasting approach to the idea/expression dichotomy.[292] The plaintiff in this case, Computer

---

286. *Id*. at 1238 n.34 (citation omitted).

287. Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 705 (2d Cir. 1992) (quoting 3 MELVIN B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 13.03[F], at 13-62.34) (2003).

288. *See, e.g.*, Gates Rubber Co. v. Bando Chem. Indus., 9 F.3d 823, 840 (10th Cir. 1993); *Computer Assocs.*, 982 F.2d at 705; Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1524–25 (9th Cir. 1992); Plains Cotton Co-Op Ass'n v. Goodpasture Computer Serv., Inc., 807 F.2d 1256, 1262 (5th Cir. 1987).

289. *See, e.g.*, NIMMER, *supra* note 287; Michael A. Jacobs, *Copyright and Compatibility*, 30 JURIMETRICS J. 91, 103 (1989); Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 STAN. L. REV. 1045, 1074 & 1082–83 (1989); David Nimmer et al., *A Structured Approach to Analyzing the Substantial Similarity of Computer Software in Copyright Infringement Cases*, 20 ARIZ. ST. L.J. 625, 629–30 (1988); Cary S. Kappel, *Copyright Protection of SSO: Replete with Internal Deficiencies and Practical Dangers*, 59 FORDHAM L. REV. 699, 707–08 (1991); Steven R. Englund, Note, *Idea, Process or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Computer Programs*, 88 MICH. L. REV. 866, 881 (1990); Thomas M. Gage, Note, Whelan Associates v. Jaslow Dental Laboratories*: Copyright Protection for Computer Software Structure – What's the Purpose?*, 1987 WIS. L. REV. 859, 860–61 (1987); Amaury Cruz, Comment, *What's the Big Idea Behind the Idea-Expression Dichotomy? – Modern Ramifications of the Tree of Porphyry in Copyright Law*, 18 FLA. ST. U. L. REV. 221, 246–47 (1990); Mark T. Kretschmer, Note, *Copyright Protection for Software Architecture: Just Say No!*, 1988 COLUM. BUS. L. REV. 823, 824–27 (1988).

290. *See, e.g.*, Dun & Bradstreet Software Servs., Inc. v. Grace Consulting, Inc., 307 F.3d 197, 206 & 214–16 (3d Cir. 2002) (citing *Whelan* as principal authority for copyright infringement test, but acknowledging defenses available under the *Computer Associates* line of cases); Lotus Dev. Corp. v. Borland Int'l, Inc., 799 F. Supp. 203, 215 (D. Mass. 1992) ("*Whelan* remains good authority in the Third Circuit and may provide guidance for this court.").

291. *See, e.g.*, *Gates Rubber Co*., 9 F.3d at 840–41; *Computer Assocs.*, 982 F.2d at 702–03.

292. 982 F.2d 693 (2d Cir. 1992).

Associates ("CA"), had developed an application program, CA-SCHEDULER, for use with three operating system platforms for IBM System 370 mainframe computers.[293] Rather than develop three separate and distinct versions of the entire program for each operating system, CA divided CA-SCHEDULER into two components: "a first component that contain[ed] only the task-specific portions of the program" and "a second component that contain[ed] all the interconnections between the first component and the operating system."[294] CA developed three versions of the second component, one for each IBM operating system, to serve as an interface between the first component and the different IBM operating systems.[295] This second component, called ADAPTER, served as a middleware layer[296] between the first component and each of the three different operating system platforms.[297] Thus CA needed to develop only one version of the first component to complete the development of CA-SCHEDULER.

Altai, a competitor of CA, had developed a similar application program for use with only one of the IBM operating systems.[298] Seeking to adapt the program for use with other operating systems, Altai undertook to develop a component called OSCAR that, like ADAPTER, would provide a common programming interface to different operating systems.[299] Altai hired a CA employee to develop the component, who later admitted to copying approximately thirty percent of OSCAR's code from ADAPTER.[300] After Altai began licensing the OSCAR component in its software products, CA sued Altai for copyright infringement.[301]

Altai responded to the lawsuit by developing a revised version of OSCAR in which all of the copied sections of code were rewritten by programmers who did not have access to ADAPTER and had not been involved in the development of the original version of OSCAR.[302] Altai admitted that the original version of OSCAR infringed CA's copyright in ADAPTER,[303] but argued at trial that CA had failed to show a substantial similarity between the revised version of OSCAR and the copyrightable elements of ADAPTER.[304] The district court

293. *See id*. at 698.
294. *Id*. at 699.
295. *See id*.
296. *See supra* Part II.C.1.
297. *See Computer Assocs.*, 982 F.2d at 699.
298. *See id*.
299. *See id*. at 699–700.
300. *See id*.
301. *See id*. at 700.
302. *See id*.
303. *See* Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp. 544, 560 (E.D.N.Y. 1991).
304. *See id*. at 561–62.

agreed, and concluded that the revised version of OSCAR did not infringe CA's copyright in ADAPTER.[305]

In its appeal to the Second Circuit, CA argued that the district court should have found copyright infringement due to substantial similarities between the second version of OSCAR and certain protected structural elements of ADAPTER.[306] In addressing these arguments, the Second Circuit agreed with the *Whelan* court that copyright protection may extend to some non-literal elements of a computer program,[307] but rejected the *Whelan* court's approach to the idea/expression dichotomy in favor of a three-step analytical procedure:[308]

1. Abstraction: "[D]issect the allegedly copied program's structure and isolate each level of abstraction contained within it. This process begins with the code and ends with an articulation of the program's ultimate function."[309]

2. Filtration: "[S]eparat[e] protectable expression from nonprotectable material. This process entails examining the structural components at each level of abstraction to determine whether their particular inclusion at that level was 'idea' or was dictated by considerations of efficiency, so as to be necessarily incidental to that idea; required by factors external to the program itself; or taken from the public domain and hence is nonprotectable expression."[310] Filtering the "structural components . . . required by factors external to the program"[311] may exclude from copyright protection those elements in which the "freedom of design choice"[312] available to the developer of the allegedly infringed program was "circumscribed by extrinsic considerations such as (1) the mechanical specifications of the computer on which a particular program is intended to run; (2) compatibility requirements of other programs with which a program is designed to operate in conjunction; (3) computer manufacturers' design standards; (4) demands of the industry being serviced; and (5) widely accepted programming practices within the computer industry."[313]

3. Comparison: Determine "whether the defendant copied any aspect of [the remaining] protected expression," and assess "the copied

---

305. *See id*. at 562.
306. *See Computer Assocs.*, 982 F.2d at 702.
307. *See id*.
308. *See id*. at 706.
309. *Id*. at 707.
310. *Id*. (citing 3 NIMMER, *supra* note 287, § 13.03[F]; Kretschmer, *supra* note 289, at 844–45).
311. *Id*.
312. *Id*. at 709.
313. *Id*. at 709–10 (citing 3 NIMMER, *supra* note 287, § 13.03[F][3], at 13-66 to 13-71).

portion's relative importance with respect to the plaintiff's overall program."[314]

Table 3: The Second Circuit's Analysis of the Similarities Between Elements of ADAPTER and the Revised Version of OSCAR

Level of Abstraction	**Similar Elements**	**Rationale for Filtration**
Program code	"Virtually no lines of code"[315]	N/A
Programming interface	"Parameter lists and macros"[316]	All but a few "were either in the public domain or dictated by the functional demands of the program"[317]
Calls to operating system	"Overlap . . . between the list of services required" for the operating system [318]	List of services was "determined by the demands of the operating system and of the application program to which it [was] to be linked through ADAPTER or OSCAR"[319]
General organization	Similarities between the programs' organizational charts that were "simple and obvious to anyone exposed to the operation of the program[s]"[320]	Similar elements "follow naturally from the work's theme rather than from the author's creativity," and therefore belong to the public domain[321]

Reviewing the trial record, the appeals court found that the district court had made sufficient factual findings to provide a rationale for filtering out (i.e., finding uncopyrightable) each of the similar elements between ADAPTER and the revised version of OSCAR.[322] The district court had identified four levels of abstraction: the program code,[323] the programming interface,[324] the calls made to the op-

314. *Id*. at 710 (citing 3 NIMMER, *supra* note 287, § 13.03[F][5]; Data East USA, Inc. v. Epyx, Inc., 862 F.2d 204, 208 (9th Cir. 1988)).

315. *Id*. at 714 (quoting *Computer Assocs.*, 775 F. Supp. at 561).

316. *Id*.

317. *Id*. (quoting *Computer Assocs.*, 775 F. Supp. at 562).

318. *Id*. at 715.

319. *Id*. (quoting *Computer Assocs.*, 775 F. Supp. at 562).

320. *Id*.

321. *Id*. (quoting 3 NIMMER, *supra* note 287, § 13.03[F][3], at 13-65).

322. *See id*. at 714–15.

323. *See id*. at 714 (identifying "object code" and "source code" as abstractions); *see also supra* note 164. The distinction between object code and source code was not of significance to the analysis in *Computer Associates* and so will not be discussed here.

324. *See Computer Assocs.*, 982 F.2d at 714 (identifying "parameter lists"). As used here, parameter lists refer to the calling conventions that are specified by the programming interface of a software component.

erating system,[325] and the general organization of the program.[326] As Table 3 summarizes, the district court had also determined that at each of these levels, all or almost all of the similar elements between ADAPTER and the revised version of OSCAR were either "required by factors external to the program" or "taken from the public domain."[327] Accordingly, the Second Circuit concluded that the district court properly found that there was no substantial similarity between the revised version of OSCAR and the protectable expression in ADAPTER, and affirmed the denial of CA's copyright infringement claim with respect to the revised version of OSCAR.[328]

Although *Whelan* remains good law in the Third Circuit,[329] *Computer Associates* has been favored throughout the federal courts since 1992.[330] In contrast to the widespread criticism of *Whelan*,[331] most commentators have praised the abstraction-filtration-comparison approach of *Computer Associates*,[332] and some have come close to describing the Second Circuit's approach as superseding that of the Third Circuit.[333] It is therefore reasonable to expect that, for the fore-

325. *See id*. (identifying "services required"). As used here, services required refer to the calls made by ADAPTER to other software components; i.e., the operating system.

326. *See id*. (identifying "general outline").

327. These characteristics of elements of a program are discussed in step two of the court's three-step analytic procedure for determining whether copyright extends to non-literal elements of a computer program. *See id*. at 702.

328. *See id*. at 715.

329. *See supra* note 291.

330. *See, e.g.*, Softel, Inc. v. Dragon Med. & Scientific Communications, Inc., 118 F.3d 955 (2d Cir. 1997); MiTek Holdings, Inc. v. Arce Eng'g Co., Inc., 89 F.3d 1548 (11th Cir. 1996); Bateman v. Mnemonics, Inc., 79 F.3d 1532 (11th Cir. 1996); Eng'g Dynamics, Inc. v. Structural Software, Inc., 26 F.3d 1335 (5th Cir. 1994); Gates Rubber Co. v. Bando Chem. Indus., Ltd., 9 F.3d 823 (10th Cir. 1993); Autoskill Inc. v. National Educ. Support Sys., Inc., 994 F.2d 1476 (10th Cir. 1993); Sega Enters., Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992); Atari Games Corp. v. Nintendo of Am., Inc., 975 F.2d 832 (Fed. Cir. 1992).

331. *See supra* note 290.

332. *See, e.g.*, Brief of Amicus Curiae Copyright Law Professors, Lotus Dev. Corp. v. Borland Int'l, Inc., 49 F.3d 807 (1st Cir. 1995) (endorsing *Computer Associates* approach), *reprinted in* 16 HASTINGS COMM. & ENT. L.J. 657 (1994); David Bender, Computer Associates v. Altai*: Rationality Prevails*, COMPUTER LAWYER, Aug. 1992, at 1; Douglas Derwin, *It is Time to Put "Look and Feel" Out to Pasture*, 15 HASTINGS COMM. & ENT. L.J. 605 (1993); Aram Dobalian, Comment, *Copyright Protection for the Non-Literal Elements of Computer Programs: The Need for Compulsory Licensing*, 15 WHITTIER L. REV. 1019, 1073 (1994); Stephen H. Eland, Note, *The Abstraction-Filtration Test: Determining Non-Literal Copyright Protection for Software*, Computer Associates International, Inc. v. Altai, Inc., 39 VILL. L. REV. 665, 699 (1994). *But see, e.g.*, Jack E. Brown, *"Analytical Dissection" of Copyrighted Computer Software — Complicating the Simple and Confounding the Complex*, 25 ARIZ. ST. L.J. 801 (1993); Anthony L. Clapes & Jennifer M. Daniels, *Revenge of the Luddites: A Closer Look at* Computer Associates v. Altai, COMPUTER LAWYER, Nov. 1992, at 11.

333. *See, e.g.*, JONATHAN BAND & MASANOBU KATOH, INTERFACES ON TRIAL: INTELLECTUAL PROPERTY & INTEROPERABILITY IN THE GLOBAL SOFTWARE INDUSTRY 130 (1995) (stating that *Computer Associates* "buried *Whelan*'s . . . analysis"); Peter S. Menell, *Envisioning Copyright Law's Digital Future*, 46 N.Y.L. SCH. L. REV. 63 (stating that the approach "has been universally adopted by the courts since 1992").

seeable future, *Computer Associates* will control the § 102(b) inquiry in most if not all determinations as to whether a defendant's software copyright provides a warrant for conduct challenged under the antitrust laws.

3. Toward a Marketplace of § 102(b) "Idea[s]"

In its *Computer Associates* decision, the Second Circuit aspired to develop a "pragmatic" procedure for identifying copyrightable elements in software "which also keeps in consideration 'the preservation of the balance between competition and protection.'"[334] To the extent that this conception of competition is understood as disfavoring legal barriers to entry into the software product markets described in Part II, the court has largely succeeded. By denying copyright protection to program elements where "a programmer's freedom of design choice is . . . circumscribed by extrinsic considerations,"[335] the *Computer Associates* filter supports a well-functioning software product market by removing legal impediments to quality competition in every product market in which a software product competes, while still allowing the expressive elements in the product's accompanying software to obtain full protection.

To engage in quality competition against an incumbent software product, vendors must be legally and technologically able to develop products that can achieve a demand substitution response by providing functional interchangeability and desirable preference and performance features.[336] To provide functional interchangeability, a competing software product needs to support the same tasks, and needs to satisfy, and be satisfied by, the same preconditions as its counterpart.[337] For purposes of competition among software products, tasks are deemed equivalent if they can be characterized by the same essential use case — i.e., the same simplified, generalized, abstract, technology-free, and implementation-independent description of the user-system interaction supporting the task.[338]

Computer Associates assures vendors that developing a competing, functionally interchangeable software product that offers desirable performance features, without more, will not give rise to copyright liability. Under the Second Circuit's analysis, the mere fact that a defendant's software product supports the same essential use cases, satisfies and is satisfied by the same preconditions, and exhibits (at least) the same performance metrics as the plaintiff's product is not

334. *Computer Assocs.*, 982 F.2d at 711 (quoting Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1253 (3d Cir. 1983)).

335. *Id*. at 709.

336. *See supra* Part II.C.3.

337. *See supra* text accompanying note 217.

338. *See supra* text accompanying note 222.

a sufficient basis for a finding of copyright infringement. First, any similarities between the essential use cases supported by the two programs should be analyzed at the highest level of abstraction and, inasmuch as they describe the general organization of the programs' behavior in a manner that is "simple and obvious to anyone exposed to the operation of the program[s]," should be deemed to belong to the public domain.[339] Second, any similarities between the programs' satisfying and satisfied preconditions should be analyzed at the programming interface level, and deemed to be "dictated by the functional demands of the program[s]."[340] Finally, to the extent that any similarities in the completeness, correctness, effectiveness, (objective) efficiency, proficiency and/or productiveness of the two programs are attributable to structural components at any level of abstraction, they should be deemed to be dictated by either "the functional demands of the program[s]" or "considerations of efficiency."[341]

While software elements dictated by performance metrics should be considered "ideas" under § 102(b), software elements dictated by preference metrics may, in some circumstances, qualify as copyrightable "expressions." The *Computer Associates* filter thus immunizes software quality competition with respect to objective but not subjective measures of usability. This distinction represents a pragmatic "balance between competition and protection."[342] On one hand, an expressive software element will not be denied copyright protection merely because many consumers subjectively see it as attractive. On the other hand, copyright will not impede the competition among software developers to improve the objectively measured usability of functionally interchangeable software, which is the primary challenge in the development of a marketable software product.[343] In short, the idea/expression dichotomy, as codified in § 102(b) of the Copyright Act and interpreted by the Second Circuit in the software development setting, is fully compatible both with the grant of exclusivity over expressive software elements and with competition in a well-functioning marketplace of § 102(b) "idea[s]."

---

339. *Computer Assocs.*, 982 F.2d at 715.

340. *Id*. at 714.

341. *Id*. at 700.

342. Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1253 (3d Cir. 1983) (quoting Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971)).

343. *See, e.g.*, CONSTANTINE & LOCKWOOD, *supra* note 175, at 518 ("It is almost always far easier to make a functional but unaesthetic system attractive than to take an attractive but impractical system and make it work. In other words, design for use comes first. Marketability and artistry are not necessarily of lesser import, but they are better achieved by dealing with them in turn.").

B. The Copyright Act's Allocation of Rights

The ability of a consumer to use a software product depends not only on the development of usable software,[344] but on the legal rights that constitute the software product itself, as defined by the terms of the accompanying software license. Most of these license terms track the Copyright Act's default allocation of rights, either by granting to the user certain nonexclusive rights within the scope of the copyright owner's exclusive rights or by leaving the default allocation unchanged.[345] Violations of such license terms may constitute both a contractual breach and a copyright infringement.[346] Other license terms, however, purport to enlarge the scope of the copyright owner's rights under the Copyright Act. These terms are enforceable, if at all, under state contract law, and may be subject to preemption by federal intellectual property law.[347] They may also be susceptible to challenge under the antitrust laws to the extent that they exceed the legitimate exercise of a copyright owner's exclusionary rights.[348] In addition, software license agreements usually contain terms other than those relating to rights in the software, such as warranties, limitations of liability, and choice of law.[349] Such terms are regarded as independent contractual covenants; they do not bear on the scope of the license and are enforceable only under contract law and not copyright law.[350]

Contractual variations aside, the Copyright Act's default allocation of rights supports the marketing of expressive works by providing copyright owners with various exclusive rights that can be licensed, including the right to copy. The value of a software product to a consumer, however, does not subsist primarily in the right to copy the accompanying software, but in the right to run it on a computer system. It turns out that this right to run software embraces an intricate combination of rights and defenses under the Copyright Act, which needs to be precisely understood.

---

344. *See supra* Part II.E.

345. *See* O'Rourke, supra note 248, at 490 (noting that "with few exceptions, . . . [software license] terms track those of the Copyright Act").

346. *See, e.g.*, S.O.S., Inc. v. Payday, Inc., 886 F.2d 1081, 1087 (9th Cir. 1989) ("A licensee infringes the owner's copyright if its use exceeds the scope of its license.").

347. *See* 17 U.S.C. § 301(a) (2004) (preempting the grant of rights under "the common law or statutes of any State" that are "equivalent to any of the exclusive rights within the general scope of copyright"); *see also* Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 269–70 (5th Cir. 1988) (holding prohibition against decompilation in a standard form software licensing agreement unenforceable despite contrary state statute because the state statute "touches upon an area" of federal copyright law).

348. *See supra* note 245 and accompanying text.

349 *See* O'Rourke, *supra* note 248, at 490 n.39.

350. *See, e.g.*, Sun Microsystems, Inc. v. Microsoft Corp., 188 F.3d 1115, 1121–23 (9th Cir. 1999).

1. The Statutory Grant

Section 106 of the Copyright Act enumerates a bundle of exclusive rights granted to the owner of copyright in a protected work. Of the six listed rights, three are applicable to software. Specifically, the owner of copyright in a computer program has the exclusive rights to reproduce the program himself and to authorize the reproduction of the program by others,[351] to prepare derivative works based upon the program,[352] and to distribute of copies of the program to the public.[353] Of these three rights, only the first two — the reproduction right and the derivative works right — are normally implicated by a consumer's use of a software product.

The reproduction right is implicated when a copy of a copyrighted work is created and "fixed" in a "material object[ ] . . . from which the work can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device."[354] A copy is "fixed" if it is "sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration."[355] The hardware devices and digital storage media from which software is read and reproduced by a computer system are facially "material object[s]" within the meaning of the statute.[356] Thus, the installation of copyrighted software onto a computer (typically by copying it from a CD-ROM or floppy disk onto a hard drive) necessarily creates and fixes a copy of the software in a machine-readable medium, and therefore implicates the reproduction right.[357]

A temporary copy of copyrighted code created in random access memory ("RAM") prior to its execution by a microprocessor, while often described as "ephemeral,"[358] has also been found to be sufficiently stable to constitute fixation. In *MAI Systems Corp. v. Peak*

---

351. *See* 17 U.S.C. § 106(1) (2000).

352. *See id*. § 106(2).

353. *See id*. § 106(3).

354. *Id*. § 101 (defining "copies").

355. *Id*. (defining "fixed").

356. *See, e.g.*, Victor F. Calaba, *Quibbles 'N Bits: Making a Digital First Sale Doctrine Feasible*, 9 MICH. TELECOMM. & TECH. L. REV. 1, 12 (2002) ("[T]here is little question as to whether a RAM chip is a 'material object.'").

357. *See, e.g.*, Stenograph L.L.C. v. Bossard Assocs., Inc., 144 F.3d 96, 100 (D.C. Cir. 1998) ("[I]f someone loads validly copyrighted software onto his or her own computer without the owner's permission, and then uses the software for the principal purposes for which it was designed, there can be no real doubt that the protected elements of the software have been copied . . . .").

358. *See, e.g.*, Advanced Computer Servs. of Mich., Inc. v. MAI Sys. Corp., 845 F. Supp. 356, 362–63 (E.D. Va. 1994) (holding RAM copy to be a fixation notwithstanding its "ephemeral or transient" nature); Dan L. Burk, *The Trouble With Trespass*, 4 J. SMALL & EMERGING BUS. L. 27, 40 (2000) ("[S]ome courts have moved away from the requirement of stable, tangible copies and instead affirmed the copyrightability of ephemeral, so-called 'RAM copies.'").

Computer, Inc., it was undisputed that the defendant Peak, a computer maintenance services firm, had executed MAI's operating system software on its customers' computers and had viewed the software's output ("the system error log"), and that these actions necessarily entailed loading the software into the computer's RAM without MAI's authorization.[359] The district court granted summary judgment in favor of MAI on its claims of copyright infringement,[360] and Peak appealed.[361] Describing the system error log as "part of the operating system,"[362] the Ninth Circuit reasoned that inasmuch as Peak had loaded the software into RAM and was then able to view the system error log, the RAM copy created by Peak was "fixed" within the meaning of the Copyright Act.[363] Accordingly, the court held that "the loading of software into the RAM creates a copy under the Copyright Act."[364]

This holding of *MAI* is questionable because the court's conclusion that the RAM copy of MAI's copyrighted program code was capable of being "perceived . . . for a period of more than transitory duration"[365] appears to have been based on the incorrect premise that the system error log was part of the copyrighted code. Since the court had found only that Peak had viewed the system error log, not the code itself, there was no predicate act of perceiving the RAM copy *of the copyrighted work* from which to infer that such acts were permitted "for a period of more than transitory duration."[366] Other commentators have also criticized the holding as an unwarranted departure from prior caselaw and copyright policy.[367]

Nevertheless, courts, usually without expressly addressing the fixation requirement, have generally held that the loading of code into a computer's memory prior to executing it creates an infringing "copy" within the meaning of § 106.[368] The Information Infrastructure

---

359. *See* MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 (9th Cir. 1993).

360. *See id*. at 517.

361. *See id*. at 513.

362. *Id*. at 518.

363. *See id*.

364. *Id*. at 519.

365. *Id*. at 518 (quoting 17 U.S.C. § 101).

366. *Cf.* Mark A. Lemley, *Dealing with Overlapping Copyrights on the Internet*, 22 U. DAYTON L. REV. 547, 550–51 (1997) (noting that *MAI* "does not discuss the 'transitory duration' prong of the fixation test" (emphasis in original)).

367. *See, e.g.*, Ronald S. Katz & Janet S. Arnold, MAI v. Peak*: An Unprecedented Opinion with Sparse Analysis*, COMPUTER LAWYER, May 1993, at 19; Jessica Litman, *The Exclusive Right to Read*, 13 CARDOZO ARTS & ENT. L.J. 29, 41–43 (1994); David Nimmer, *Brains and Other Paraphernalia of the Digital Age*, 10 HARV. J.L. & TECH. 1, 21–25 (1996).

368. *See, e.g.*, Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 260 (5th Cir. 1988) ("[T]he act of loading a program from a medium of storage into a computer's memory creates a copy of the program . . . ."); ISC-Bunker Ramo Corp. v. Altech, Inc., 765 F. Supp. 1310, 1332 (N.D. Ill. 1990) ("Loading a computer's memory requires copying of the program from a disk into memory, and that copy is a direct infringement of the copyright.");

Task Force[369] and the leading copyright treatise[370] have also endorsed the Ninth Circuit's approach to the fixation issue. The prevailing view, then, appears to be that the creation of a temporary copy of copyrighted code in RAM prior to its execution by a microprocessor implicates the reproduction right of § 106.

2. The § 117 Limitation

As originally enacted in 1976, the modern Copyright Act did not specifically provide for copyright protection of software. At the time of enactment, Congress was awaiting the report of the Commission on New Technological Uses of Copyrighted Works ("CONTU").[371] CONTU had been established in 1974 to study, *inter alia*, the creation, reproduction, and use of works of authorship "in conjunction with automatic systems capable of storing, processing, retrieving, or transferring information," and to recommend such statutory provisions as "may be necessary to assure for such purposes access to copyrighted works, and to provide recognition of the rights of copyright owners."[372] Accordingly, the original § 117 of the Copyright Act of 1976 provided that rights with respect to these technological uses of copyrighted works would be governed by the copyright laws in force just prior to the effective date of the Act.[373]

In 1978, CONTU issued a final report identifying four policy objectives concerning copyright protection of software works:

> 1. Copyright should proscribe the unauthorized copying of these works.
>
> 2. Copyright should in no way inhibit the rightful use of these works.

Apple Computer, Inc. v. Formula Int'l, Inc., 562 F. Supp. 775 (C.D. Cal. 1983) (holding that copying from a diskette into ROM is a copy); Micro-Sparc, Inc. v. Amtype Corp., 592 F. Supp. 33 (D. Mass. 1984) (holding that inputting a program into memory constitutes a copy).

369. *See* INFO. INFRASTRUCTURE TASK FORCE, INTELLECTUAL PROPERTY AND THE NATIONAL INFORMATION INFRASTRUCTURE: THE REPORT OF THE WORKING GROUP ON INTELLECTUAL PROPERTY RIGHTS 65 n.204 (1995) (describing *MAI* court's reasoning as "quite unexceptional").

370. 2 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 8.08[A][1], at 8-122.3 (2003) ("In *MAI v. Peak*, the fixation in RAM was evidently more than momentary, as it sufficed to permit a user 'to view the system error log and diagnose the problem with the computer . . . .'" (quoting MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 (9th Cir. 1993))).

371. *See* 17 U.S.C. § 117 (2004).

372. Act of Dec. 31, 1974, Pub. L. No. 93-573, §§ 201(b)(1)(A), 201(c), 88 Stat. 1873, 1873–74; *see also* NAT'L COMM'N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 4 (1978) (hereinafter CONTU REPORT).

373. *See* 17 U.S.C. § 117 (1976).

> 3. Copyright should not block the development and dissemination of these works.
>
> 4. Copyright should not grant anyone more economic power than is necessary to achieve the incentive to create.[374]

To attain these objectives, CONTU recommended the enactment of a new § 117, to provide in relevant part:

> Notwithstanding the provisions of § 106, it is not an infringement for the rightful possessor of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided . . . that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner . . . .[375]

In effect, § 117 serves both as a limitation on the exclusive rights granted under § 106 to the owner of copyright in a computer program, and as a guarantee of rights to certain other parties who may desire to use the program. In explaining the rationale for this recommendation, CONTU noted the concern that using a copyrighted program might entail creating copies or adaptations despite the contrary wishes of the copyright owner and explained that the proposed § 117 would ensure that a person who had rightfully acquired a copy of a program and who wished to use it would be assured of the benefit of the bargain — i.e., the right to use the program for the purposes contemplated by the parties to the acquisition:

> Because the placement of a work into a computer is the preparation of a copy, the law should provide that persons in rightful possession of copies of programs be able to use them freely without fear of exposure to copyright liability . . . . One who rightfully possesses a copy of a program, therefore, should be provided with a legal right to copy it to that extent which will permit its use by that possessor. This would include the right to load it into a computer . . . .
>
> Because of a lack of complete standardization among programming languages and hardware in the com-

374. CONTU REPORT, *supra* note 372, at 30.
375. *Id*.

> puter industry, one who rightfully acquires a copy of a program frequently cannot use it without adapting it to that limited extent which will allow its use in the possessor's computer. The copyright law . . . should no more prevent such use than it should prevent rightful possessors from loading programs into their computers. Thus, a right to make those changes necessary to enable the use for which it was both sold and purchased should be provided . . . . These rights would necessarily be more private in nature than the right to load a program by copying it and could only be exercised so long as they did not harm the interests of the copyright proprietor . . . . Should proprietors feel strongly that they do not want rightful possessors of copies of their programs to prepare such adaptations, they could, of course, make such desires a contractual matter.[376]

In its 1980 amendments to the Copyright Act, Congress enacted CONTU's legislative recommendations "almost verbatim."[377] There was one change: for the phrase describing the intended beneficiary of § 117, i.e., "the rightful possessor of a copy of a computer program," Congress substituted the language "the owner of a copy of a computer program."[378] No explanation is given for this change. More generally, the legislative history for the 1980 amendments is "scant," consisting only of the statement that the new § 117 "embodies the recommendations of [CONTU] with respect to clarifying the law of copyright of computer software."[379] Courts have therefore regarded the CONTU final report as providing the legislative history of § 117.[380]

a. "Owners" Eligible for Protection

It is not immediately apparent whether a consumer who acquires a software product is entitled to benefit from § 117 as "the owner of a copy of a computer program." The acquisition of a software product may involve a transfer both of "copyright rights in the software program (intellectual property rights)" and of "rights in the copy of the program through the material object that embodies the copyrighted

376. *Id*. at 31–33.

377. Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1252 (3d Cir. 1983).

378. 17 U.S.C. § 117 (1976).

379. Atari, Inc. v. JS & A Group, Inc., 597 F. Supp. 5, 9 (N.D. Ill. 1983) (quoting H.R. REP. NO. 96-1307, pt. 1, at 23 (1980)).

380. *See, e.g.*, *id*. at 9.

work (personal property rights)."[381] The Copyright Act expressly provides that these two kinds of rights are legally separate and distinct and that "[t]ransfer of ownership of any material object . . . does not of itself convey any rights in the copyrighted work embodied in the object; nor, in the absence of an agreement, does transfer of ownership of a copyright or of any exclusive rights under a copyright convey property rights in any material object."[382]

In cases outside of the mass-market setting, some courts have conflated the distinction between ownership of copyright interests in software and ownership of the material object containing a copy of the software. For example, in *MAI v. Peak*, discussed above, the Ninth Circuit stated in a footnote, without further discussion, that "[s]ince MAI licensed its software, the Peak customers do not qualify as 'owners' of the software and are not eligible for protection under § 117."[383] The court's cursory treatment of the "owner of a copy" determination failed to recognize that, in some cases, a licensee of intellectual property rights in software might be deemed to own a copy of the software through the physical media in which the software was embodied.[384] In effect, the Ninth Circuit's statement amounted to a holding that § 117 protected consumers only in those rare instances where they had rightfully acquired a copy of the software without entering into a "license agreement."[385] Although *MAI* is still good law in the Ninth Circuit,[386] this aspect of the decision has been widely criticized[387] and has subsequently been overruled with respect to machine

381. *See* Applied Info. Mgmt., Inc. v. Icart, 976 F. Supp. 149, 150–51 (E.D.N.Y. 1997).

382. 17 U.S.C. § 202 (2004).

383. MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 n.5 (9th Cir. 1993). In a separate case also involving MAI's software licenses, a district court reasoned that MAI customers were not eligible for § 117 protection because as licensees, they were "not 'owners' of *the copyrighted software*." *See* Advanced Computer Servs. of Michigan, Inc. v. MAI Sys. Corp., 845 F. Supp. 356, 367 (E.D. Va. 1994) (emphasis added). This was an erroneous reading of the crucial words "of a copy" out of § 117. *See* NIMMER, *supra* note 287, § 8.08[B][1], at 8-123.

384. *See, e.g.*, Vault Corp. v. Quaid Software, Ltd., 847 F.2d 255, 261 (5th Cir. 1988) (holding software licensee's copying of copyrighted software to be non-infringing under § 117); Foresight Res. Corp. v. Pfortmiller, 719 F. Supp. 1006, 1009–10 (D. Kan. 1989) (finding that defendant's client, who had obtained plaintiff's software subject to a license agreement, was a lawful "owner of a copy" under § 117).

385. *See* Lothar Determann & Aaron Xavier Fellmeth, *Don't Judge a Sale by Its License: Software Transfers Under the First Sale Doctrine in the United States and the European Community*, 36 U.S.F. L. REV. 1, 36–37 (2001).

386. *See, e.g.*, Triad Sys. Corp. v. Southeastern Exp. Co., 64 F.3d 1330, 1335 (9th Cir. 1995) (following *MAI*).

387. *See, e.g.*, NIMMER, *supra* note 287, § 8.08[B][1], at 8-122.8 to 8-123; Trinnie Arriola, *Software Copyright Infringement Claims After MAI Systems v. Peak Computer*, 69 WASH. L. REV. 405, 419–20, 422–23 (1994); Determann & Fellmeth, *supra* note 385, at 41; Katz & Arnold, *supra* note 367, at 19–20; Katrine Levin, Note, MAI v. Peak*: Should Loading Operating System Software Into RAM Constitute Copyright Infringement?*, 24 GOLDEN GATE U. L. REV. 649, 668–77 (1994); Carol G. Stovsky, Note, MAI Systems Corp. v. Peak Computer, Inc.*: Using Copyright Law to Prohibit Unauthorized Use of Computer Software*, 56 OHIO ST. L.J. 593, 601–04 (1995); *see also* DSC Communications Corp. v. Pulse Com-

maintenance and repair activities through an amendment to § 117 in the Digital Millennium Copyright Act of 1998.[388]

The prevailing view today appears to be that the mere fact that a software transaction takes the *form* of a license of rights derived from the vendor's copyright in the software is immaterial for purposes of determining the consumer's eligibility to benefit from § 117. Instead it is the *substance* of the consumer's rights that is determinative.[389]

For example, in *DSC Communications v. Pulse Communications, Inc.*, the plaintiff DSC licensed a telecommunications system to various regional Bell operating system companies ("RBOC's"), which included an interface card that was designed to download and run DSC's software.[390] Pulse designed a competing card that also downloaded DSC's software.[391] After some of DSC's licensees began using Pulse's cards, DSC sued Pulse for, *inter alia*, contributory infringement of the copyright in DSC's software.[392] Following a jury trial, Pulse moved for judgment as a matter of law on this claim.[393] Finding that the RBOC's were "owners" under § 117, the district court concluded that they were entitled to download DSC's software onto Pulse's cards, and granted Pulse's motion.[394]

On appeal to the Federal Circuit, DSC challenged the district court's conclusion that the RBOC licensees were "owners."[395] Applying Fourth Circuit law, the appeals court acknowledged that licensees can be "owners" under § 117 and rejected the Ninth Circuit's analysis in *MAI*.[396] The court went on, however, to examine the terms of DSC's software license agreements, and found that they "severely limit[ed] the rights of the [licensees] . . . in ways that are inconsistent with the rights normally enjoyed by owners of copies of software."[397]

---

munications, Inc., 170 F.3d 1354, 1360 (Fed. Cir. 1999) (noting Nimmer's criticism of *MAI*).

388. *See* 17 U.S.C. § 117(c) (2004) (exempting from infringement copies made for purposes of machine maintenance or repair).

389. *See, e.g.*, NIMMER, *supra* note 287, § 8.08[B][1], at 8-124 ("whether the software vendor calls its subject contract a 'license' or a 'bill of sale' is immaterial"); Dieterman & Fellmeth, *supra* note 385, at 41 (rejecting Ninth Circuit's approach of allowing "pro forma labels" to govern whether § 117 applies); *see also* Telecomm Technical Servs., Inc. v. Siemens Rolm Communications, Inc., 66 F. Supp. 2d 1306, 1325 & n.20 (N.D. Ga. 1998) (finding *DSC Communications* persuasive in view of Nimmer's criticism of *MAI*); Applied Info. Mgmt., Inc. v. Icart, 976 F. Supp. 149, 153 (E.D.N.Y. 1997) ("Determination of whether an agreement transfers ownership of a copy of a computer program requires interpretation of the contract between the parties.").

390. 170 F.3d 1354, 1357–58 (Fed. Cir. 1999).

391. *See id*. at 1358.

392. *See id*. at 1357.

393. *Id*.

394. *See id*. at 1359–60.

395. *See id*. at 1360.

396. *Id*. ("Plainly, a party who purchases copies of software from the copyright owner can hold a license under a copyright while still being an 'owner' of a copy of the copyrighted software for purposes of section 117.").

397. *Id*. at 1361.

In particular, each of the agreements limited the licensee's right to transfer copies of the software to third parties, a right that would have been guaranteed to an "owner of a copy" of the software under the "First Sale" doctrine of § 109.[398] Accordingly, the Federal Circuit concluded that the RBOC licensees in this case were not "owners" under § 117, and reversed the district court's judgment.[399]

Nimmer's copyright treatise has taken issue with the Federal Circuit's characterization of the *DSC* license limitations, arguing that they were directed to the scope of intellectual property in the copyrighted software, and not to the "incidents of ownership [that] could be exercised over the physical media in which that software was embodied."[400] Nimmer contends that the determination that a consumer is an "owner of a copy" under § 117 should be based solely on whether the material object containing the copy is the consumer's personal property.[401] Following this approach, if the terms of a software license agreement permit the consumer to "repurpose the physical media in which the software was delivered as 'door jambs, landfill, or (absent blank floppies in a pinch) deleting the software and re-using the disks to store vital company documents'" without incurring liability to the vendor, then the consumer should be deemed to have ownership of a copy of the software through ownership of the physical media in which it was embodied.[402]

According to Lothar Determann and Aaron Fellmeth, a consumer's eligibility both to make copies and adaptations of software under § 117 and to transfer copies of software under § 109 should be based on the overall "sales character" of the transaction under commercial law.[403] Thus, if the transaction does not involve either "an assignment of the copyright itself (sale of the copyrights) or a lease of the individual copy (lease of a software copy) . . . then the transaction seems to be properly characterized as a sale of a software copy, which should trigger the 'First Sale' doctrine of section 117."[404] In reviewing license terms, the appropriate focus under this approach is on any additional rights and restrictions that are not generally implied in a sales context.[405]

Other commentators have suggested that insofar as most mass market software license agreements are contracts of adhesion, restric-

398. *See id*. at 1361–62 (citing 17 U.S.C. § 109).

399. *See id*. at 1362.

400. NIMMER, *supra* note 287, § 8.08[B][1], at 8-125 n.39.10g.

401. *See id*. at 8-122.8 to 8-125.

402. *Id*. at 8-125 n.39.10f (*quoting* David Nimmer, *Brains and Other Paraphernalia of the Digital Age*, 10 HARV. J.L. & TECH. 1, 22 (1996)); *cf.* Determann & Fellmeth, *supra* note 385, at 41 ("[T]he transfer of perpetual user rights characterize a 'sale' of a copy of software.").

403. *See id*. at 41–42.

404. *Id*.

405. *See id*.

tive mass market license terms should generally be viewed with skepticism, particularly when they purport to deprive the licensee of protections provided by the federal copyright laws.[406] Such perspectives are part of an extensive ongoing debate over the enforceability and federal preemption of mass market, "shrinkwrap," and "clickwrap" license agreements.[407]

Developments since *MAI* have shown that, even though there remains considerable disagreement as to which rights constitute the *sine qua non* of "ownership of a copy," there is substantial agreement among courts and commentators that the § 117 eligibility inquiry requires a particularized examination of the consumer's rights under the terms of a software license.[408] For purposes of characterizing legal and technological impediments to software use as restraints of trade in antitrust analysis, it is sufficient to observe that under each of the post-*MAI* interpretational approaches described here, the only way a consumer of a software product can be legally disqualified from the protections of § 117 is by entering into an enforceable agreement with the vendor whose terms operate to remove some or all of those protections.[409] In other words, nothing in the Copyright Act operates to impede a consumer from creating copies and adaptations of a computer program that are essential to enable its use for the purposes contemplated by the vendor and consumer. Any such impediments are, therefore, either contractual or technological in nature.

---

406. *See, e.g.*, Thomas Lee Hazen, *Contract Principles as a Guide for Protecting Intellectual Property Rights in Computer Software*, 20 U.C. DAVIS L. REV. 105, 112, 149–51 (1986); Dennis S. Karjala, *Federal Preemption of Shrinkwrap and On-Line Licenses*, 22 U. DAYTON L. REV. 511, 531–33 (1996–97).

407. *See supra* note 248.

408. *See supra* text accompanying notes 389–407.

409. Notably, § 117 is an affirmative defense. Thus, even if a defendant is fully entitled to the protections of § 117, the burden remains on the defendant to prove its applicability. *See* In re Indep. Serv. Orgs. Antitrust Litigat., 964 F. Supp. 1469, 1474 (D. Kan. 1997); Allen-Myland v. Int'l Bus. Machs. Corp., 746 F. Supp. 520, 535–36 (E.D. Pa. 1990); Atari, Inc. v. JS & A Group, Inc., 597 F. Supp. 5, 10 (N.D.Ill. 1983); *see also* Am. Int'l Pictures, Inc. v. Foreman, 576 F.2d 661, 665 (5th Cir. 1978) ("[B]ecause copyright law favors the rights of the copyright holder, the person claiming authority to copy or vend generally must show that his authority to do so flows from the copyright holder.").

b. Loading Produces an "Adaptation"

CONTU's final report,[410] as well as many courts[411] and commentators,[412] treated the loading of software code into RAM as being synonymous with copying the code there. However, in most modern computer systems the loading of software code into RAM requires the creation not only of a "cop[y]," implicating the reproduction right of § 106, but also of one or more "adaptations" within the meaning of § 117. Whereas "'copy' and 'copying' are defined by the [Copyright] Act, [and] not by . . . [computer] industry usage,"[413] "loading" is a term of art in the computer industry. Thus, the existence of a discrepancy between the concepts of loading and copying is not surprising, but it is significant. Since the use of a software product entails the loading of software code into RAM, this analysis will imply that the scope of the § 117 adaptation exemption plays a more important role in defining the legal rights that constitute a software product than has been previously appreciated.

To see why "loading" implicates the § 117 adaptation exemption, it is first necessary to have a basic understanding of the concept of a "process," which represents the operational context in which software code is loaded into RAM. [414] A process may be described informally as an active computation, or defined more formally as a dynamic entity that executes code and processes data using computer system resources.[415] Processes are the entities responsible for generating all of the behavior of a computer system when it executes software.[416]

---

410. *See* CONTU REPORT, *supra* note 372, at 31 ("the placement of a [copyrighted] work into a computer is the preparation of a copy").

411. *See, e.g.*, Vault Corp. v. Quaid Software, Ltd., 847 F.2d 255, 260 (5th Cir. 1988) ("[T]he act of loading a program from a medium of storage into a computer's memory constitutes a copy of the program . . . ."); MAI Sys. Corp. v. Peak Computer, Inc., 991 F.2d 511, 517–19 (9th Cir. 1993) (discussing applicability of § 117 to "copies" made by loading software into RAM, but not to "adaptations"); Allen-Myland, Inc. v. Int'l Bus. Machs. Corp., 746 F. Supp. 520, 536 (E.D. Pa. 1990) (citation omitted) (stating that § 117 "permits only the copying of a program into a computer's memory in order to permit the computer to execute the program").

412. *See, e.g.*, Joseph P. Liu, *Owning Digital Copies: Copyright Law and the Incidents of Copy Ownership*, 42 WM. & MARY L. REV. 1245, 1256–57 (2001) ("In order to run a software program, a computer must copy portions of that software from the disk or other medium on which it is stored into the computer's RAM."); John E. Titus, Comment, *Right to Reverse Engineer Software: Is Japan Next and Does It Really Matter?*, 19 N.C. J. INT'L L. & COM. REG. 491, 497 n.44 (1994) ("[F]or the purchaser of software to run it on a computer, he must make a copy of it in the computer's random access memory (RAM).").

413. Advanced Computer Servs. of Mich., Inc. v. MAI Sys. Corp., 845 F. Supp. 356, 364 (E.D. Va. 1994).

414. *See* GARY NUTT, OPERATING SYSTEMS: A MODERN PERSPECTIVE 38 (2002).

415. *See id*.

416. *See id*. at 50 ("Processes are the fundamental schedulable unit of computation representing the execution of a program . . . . A process has a program to define its behavior, resources that are used to carry out the execution, and data on which to operate.").

In modern computer systems, the RAM and other system resources are allocated to the various processes that are running on the system.[417] Each process consists of code to be executed, data to be processed, system resources to be utilized during the computation, and a "process descriptor" that keeps track of the process's exact status, including its progress through the computation.[418] Each process has an "address space" that assigns a unique identifier, or "address," to each system resource (e.g., memory location, operating system service, file, mailbox or other object) to which it has access.[419] Wherever the code makes reference to an address within the process's address space, a process executing the code uses the system resource corresponding to that address.[420] Although each process' address space is used only by that process, system resources may be shared among several processes by being represented in each such process' address space.

To draw upon a commonly-used analogy from the culinary world,[421] if code is like a recipe, then a process is like a chef working in a designated space, during a designated time period, in a common kitchen. Within the chef's workspace are the ingredients (data) and utensils (system resources) to be used in executing the recipe (code). At any given point in time, the chef is aware of the status of all of his or her own work in progress and, if queried, can provide a precise

---

A process is an abstraction that represents "a coherent sequence of steps undertaken by a program," *see* MICROSOFT CORP., *supra* note 164, at 359 (1999), and that comes into existence only at runtime, when the software code has been linked together and loaded into memory. *See* United States v. Microsoft Corp., 84 F. Supp. 2d 9, 50 (D.D.C. 1999) ("The user who launches a program, however, is ultimately responsible for causing routines to be loaded into memory and executed together to produce the program's overall functionality."); *id*. at 55 ("For software code to provide any functionalities at all the code must be loaded into the computer's dynamic memory and executed."). Thus, to the extent that antitrust analysis may need to be grounded in an intuitive description of the "technological components" that constitute a software product, *see* United States v. Microsoft Corp., 253 F.3d 34, 81–82 (D.C. Cir. 2001) (criticizing plaintiffs' failure to identify "what are the technological components of . . . a browser"), a software product may be said to consist of the processes that are created pursuant to the user's legal rights when the product is used (as opposed to, for example, the lines of code in the accompanying software).

417. *See id*. at 164.

418. *See* FREE ON-LINE DICTIONARY OF COMPUTING, *supra* note 169 (defining "process"); NUTT, *supra* note 414, at 170–71 (describing the "process descriptor").

419. *See id*. at 166–67.

420. *See id*. at 167.

421. *See, e.g.*, Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 704 (2d Cir. 1992) (stating that computer programs are more closely similar to a utilitarian "recipe for scrambled eggs" than a creative "narration of Humpty Dumpty's demise"); Bernstein v. U.S. Dept. of State, 922 F. Supp. 1426, 1435 (N.D. Cal. 1996) (analogizing software to recipes and other "purely functional" forms of speech protected under the First Amendment); WILLIAM H. PRESS ET AL., NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (1993); Michael S. Bendel, Comment, *An Analysis That Is Not "Ad Hoc": The Bifurcated Uniform Analysis That the Federal Courts Should Follow to Determine Computer Program Copyright Nonliteral Infringement*, 12 J. MARSHALL J. COMPUTER & INFO. L. 485, 489 n.19 ("[T]hink of the computer program as a recipe for baking bread."). The recipe/cooking analogy is obviously imprecise. It is used here for expository purposes only and is not intended to lend any analytical support to the author's argument.

description of this status (process descriptor). Although the concept of an address space does not have an exact counterpart in the kitchen, it is possible to think of the chef as having a mental directory that translates each of the abstract terms in a recipe (e.g., "whisk") to the concrete utensils in the kitchen (e.g., "the copper egg whisk"), and consulting this directory while executing the recipe. Such a mental directory would play the same role as a process's address space. Although each chef has his or her own mental directory, multiple chefs can have the same copper egg whisk in mind when they read the word "whisk" in a recipe.

Just as a chef can consult (and thereby place into his or her own mental directory) a recipe book in another chef's workspace, in most computer systems a process can assign to its address space code that has already been assigned to another process's address space.[422] Also, a chef can work concurrently on multiple recipes, or multiple batches of the same recipe, all of which may be at varying stages of progress at any given time. Similarly, in most computer systems, a process can concurrently execute different sequences of code, or multiple instances of the same sequence of code, all of which may be at different stages of progress at any given time.[423] The process descriptor typically holds status information that tracks each of these concurrent computations, or "threads," as it progresses through its sequence of code.[424]

Code must be loaded into RAM in such a way that it appears within the address space of each process that needs to access and execute it. To accomplish this, the computer system performs the following actions.

First, the operating system allocates areas of RAM to the various processes, so that each process has a group of addresses in its address space that are assigned to specific physical locations in RAM.[425]

Next, the code that each process needs to execute is translated, or "linked," from the form in which it is stored on the hard drive into its final executable form.[426] In modern computer systems, linking often results in extensive modifications to the code,[427] which are too complex to describe in detail here. Linking serves in part to adjust the code to reflect the actual RAM locations where the code has been assigned to be loaded.[428] When a program is built from subprograms

422. *See* LEVINE, *supra* note 170, at 187–227 (discussing shared code libraries).

423. *See* FREE ON-LINE DICTIONARY OF COMPUTING, *supra* note 169 (defining "multithreading").

424. *See id*.

425. *See* NUTT, *supra* note 414, at 335.

426. *See id*. at 168.

427. *See* LEVINE, *supra* note 170, at 10 ("Modern computers . . . require considerably more complex code modification . . . . [T]he compiler and linker have to use complicated addressing tricks to handle data at arbitrary addresses.").

428. *Id*. at 5 (describing "relocation").

that are stored separately on the hard drive (e.g., in shared program libraries such as Windows DLL files), linking also serves to adjust any cross-references between the subprograms by identifying the RAM location of the referenced code and altering the referencing code so that it refers to that RAM location.[429]

Finally, the linked code is loaded into available blocks in RAM that correspond to locations within the process's address space.[430]

Linking and loading are usually performed by specialized computer programs known as "linkers" and "loaders," respectively.[431] Because linking and loading serve as the final means by which code is adapted to the machine environment in which it is to be executed, linkers and loaders must be "exquisitely sensitive to the architectural details, both the hardware architecture and the architecture conventions required by the operating system of their target computers."[432]

The legislative history of § 117,[433] as well as the plain meaning of the word "adaptation,"[434] provide strong support for the conclusion that loading software into RAM creates an "adaptation" of that software within the meaning of the statute. Linked and loaded code that has been "fixed" in RAM is most accurately described, in CONTU's words, as a version that has been "adapt[ed] . . . to that limited extent which will allow its use in the possessor's computer" in order to accommodate the "lack of complete standardization among programming languages and hardware in the computer industry."[435]

As a general matter, adaptations of copyrighted works may implicate the copyright owner's exclusive right to reproduce the work in copies.[436] To the extent that software loaded into RAM is substantially similar to copyrighted software so as to constitute a "copy" under § 106, it is appropriate, given CONTU's intent,[437] to continue the present practice of construing § 117 as immunizing "owners of a copy" who load software into RAM from liability for infringement of repro-

---

429. *Id*. (describing "symbol resolution").

430. *See* NUTT, *supra* note 414, at 168.

431. *See* LEVINE, *supra* note 170, at 5 ("Although there's considerable overlap between linking and loading, it's reasonable to define a program that does program loading as a loader, and one that does symbol resolution as a linker. Either can do relocation, and there have been all-in-one linking loaders that do all three functions.").

432. *Id*. at 19.

433. *See supra* notes 377–380 and accompanying text.

434. *See* WEBSTER'S NINTH NEW COLLEGIATE DICTIONARY 55 (1990) (defining "adaptation" as "adjustment to environmental conditions").

435. *See supra* note 372.

436. *See* NIMMER, *supra* note 287, § 8.09[A], at 8-138 ("[I]f the right to make derivative works, i.e., the adaptation right, has been infringed, then there is necessarily also an infringement of either the reproduction or performance rights.").

437. *See* CONTU REPORT, *supra* note 372, at 30 (stating that protection under § 117 should provide "a legal right to copy [software] to that extent which will permit its use by that possessor").

duction rights.[438] Since we have seen that software loaded into RAM is not merely a "copy," but also an "adaptation," the § 117 adaptation exemption should be recognized as an integral part of the bundle of rights that constitute a software product.

c. Rights in End Uses of a Software Product

A consumer may face considerable uncertainty in determining the applicability of the § 117 adaptation exemption to a particular use of copyrighted software. Since the exemption is a personal right,[439] its application may call for a fact-specific inquiry.[440] Also, some courts have read the exemption broadly to include uses other than those intended by the copyright owner,[441] while others have stressed the exemption's limitations.[442] It is clear that § 117 does not vitiate all use limitations in software licenses.[443] It is also clear, however, that at a minimum the exemption includes any adaptations that are "necessary to allow use of the program for the purpose for which it was purchased."[444]

---

438. *See supra* note 411.

439. *See* 17 U.S.C. § 117(b) (2004) ("Adaptations so prepared may be transferred only with the authorization of the copyright owner.").

440. *See* CONTU REPORT, *supra* note 372, at 32 ("These [adaptation] rights would necessarily be more private in nature than the right to load a program by copying it and could only be exercised so long as they did not harm the interests of the copyright proprietor.").

441. *See* Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 261 (5th Cir. 1988) ("Section 117(1) contains no language to suggest that the copy it permits must be employed for a use intended by the copyright owner, and, absent clear congressional guidance to the contrary, we refuse to read such limiting language into this exception."); *see also* Christian H. Nadan, *A Proposal to Recognize Component Works: How a Teddy Bears on the Competing Ends of Copyright Law*, 78 CAL. L. REV. 1633, 1659 (1990) ("The term 'essential' in the statute likely means essential to the buyer's utilization of the program. The buyer's desires should be the issue, and not the intention of the seller of the program.").

442. *See* Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1520 (9th Cir. 1993) (finding that defendant's use "went far beyond" that authorized by § 117, but declining to decide whether § 117 "protects only the use intended by the copyright owner"); Micro-Sparc, Inc. v. Amtype Corp., 592 F. Supp. 33, 35 (D. Mass. 1984) (holding that an owner of a printed copy of software code is not permitted under § 117 to authorize a third-party typing service to make a machine-readable disk copy); *see also* Steven Kyle Tapp & Daniel E. Wanat, *Computer Software Copyright Issues: Section 117 and Fair Use*, 22 MEM. ST. U. L. REV. 197, 213 & n.220 (1992) (noting that in *Vault*, the requirement that the copy or adaptation be created "as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner" just as easily could have been read to prohibit copying for uses contrary to those intended by the copyright owner).

443. *See, e.g.*, *Sega*, 977 F.2d at 1520; Expediters Int'l of Wash., Inc. v. Direct Line Cargo Mgmt. Servs., Inc., 995 F. Supp. 468, 478–79 (D.N.J. 1998) (finding that the defendant's use of copyrighted software beyond the scope of the license agreement's express use limitations did not fall within the § 117 exemptions); *see also* CONTU REPORT, *supra* note 372, at 33 (noting that software vendors who do not want their programs modified to support additional user purposes "could, of course, make such desires a contractual matter").

444. RAV Communications, Inc. v. Philipp Bros., Inc., 1988 WL 36174 (S.D.N.Y. Apr. 13, 1988), at *2–*3; *accord*, Aymes v. Bonelli, 47 F.3d 23, 25–27 (2d Cir. 1995).

The clarity of the § 117 safe harbor for the use of a software product for a consumer purpose "for which it was both sold and purchased," when contrasted with the unsettled status of other end uses, provides a basis for legally significant distinctions among the bundles of rights that permit the respective end uses of a software product. For example, consider a software product with two non-interchangeable end uses *A* and *B*. A vendor could offer the product under a license restricting the consumer to end use *A* only. Under the license terms and the § 117 adaptation exemption, the consumer would have a clear right to link, load and execute the accompanying software while using the product for end use *A* (the purpose for which it was purchased), but would face uncertainty as to the legality of using the product for end use *B*, even though there might be no technological impediment to doing so. While a distinction between clear and unclear legal authorization under the Copyright Act might not seem to be an intuitive approach to constituting a product for the mass market, this is essentially the same distinction that the vendors of all copyrighted works offer to consumers in urging them to purchase authorized copies of their works instead of making their own unauthorized copies and relying on the uncertain, fact-specific applicability of the fair use doctrine.

d. Significance of the Adaptation Exemption

As part of the default allocation of rights provided by the Copyright Act,[445] the § 117 adaptation exemption is an essential element of the paradigmatic software product contemplated by CONTU[446] and, by implication, Congress.[447] Within this framework, a software product consists essentially of technological access to the accompanying software through the ownership of a copy, together with legal immunity from copyright liability for acts of copying and adaptation in the due course of installing and running the software on a system according to the documentation.[448]

Although some software products may be licensed under enforceable agreements that waive the consumer's rights under § 117, both CONTU and the courts have recognized that for a consumer to use a software product for the purpose "for which it was both sold and purchased," a legal entitlement to adapt the software "to that limited ex-

445. *See, e.g.*, Tasini v. N.Y. Times Co., 206 F.3d 161, 170 (2d Cir. 2000) (holding that the Copyright Act's "default allocation and presumption of rights" applies where authors and publishers have not contracted around the statutory framework), *aff'd*, 533 U.S. 483 (2001).

446. *See* CONTU REPORT, *supra* note 372, at 33 ("[I]t is likely that many transactions involving copies of programs are entered into with full awareness that users will modify their copies to suit their own needs, and this should be reflected in the law.").

447. *See supra* notes 377–380 and accompanying text.

448. *See supra* note 165 and accompanying text.

tent which will allow its use in the possessor's computer" is often necessary and expected as a practical matter.[449] As the Second Circuit has explained, "Buyers should be able to adapt a purchased program for use on the buyer's computer because without modifications, the program may work improperly, if at all. No buyer would pay for a program without such a right."[450]

Beyond this, the foregoing analysis demonstrates that the § 117 adaptation exemption is practically necessary under far more general conditions than has been previously appreciated. While the CONTU Report cites the "conversion of a program from one higher-level language to another" and the "add[ition of] features" to a program as examples of exempted software adaptations,[451] we have seen that the mere loading of software into RAM prior to executing it also constitutes a § 117 adaptation.

The § 117 adaptation exemption serves alongside the *Computer Associates* filter[452] to ensure a well-functioning software product market, inasmuch as it guarantees that the consumer of a software product will be able to enjoy the benefit of the bargain — i.e., the ability to link, load, and execute the same code that the vendor chose to implement the product's intended purposes.[453] To impede competing software developers from determining which code is to be executed when consumers choose to use their products is to frustrate the Copyright Act's scheme for guaranteeing consumers the right to use every software product for the purpose "for which it was both sold and purchased." There is no warrant in the Copyright Act for the imposition of such a restraint.

C. Summary

The principles of copyright law and computer science discussed in this section can be summarized by reference to the detailed software-product definition that was described earlier:[454]

> A *software product* is defined by reference to accompanying software and documentation, and consists essentially of the necessary *legal rights*, and *technological capabilities*, to install and run the

449. CONTU REPORT, *supra* note 372, at 32.

450. Aymes v. Bonelli, 47 F.3d 23, 26–27 (2d Cir. 1995) (quoting CONTU REPORT, *supra* note 372, at 32). Although the citation is to the CONTU Report, the quotation actually appears to be taken from Robert A. Kreiss, *Section 117 of the Copyright Act*, 1991 B.Y.U. L. REV. 1497, 1518–19 (1991), which further states that "reasonable sellers who want to make a sale will readily agree to all the adaptations." *Id*. at 1519.

451. CONTU REPORT, *supra* note 372, at 33.

452. *See supra* Part III.A.2.

453. *See supra* Part II.E.

454. *See supra* text accompanying note 165.

> software on a system according to the documentation; it does not include any of the software code or documentation.
>
> The necessary *legal rights* consist essentially of a limited, nonexclusive license to make copies and adaptations of the software code on a computer's hard drive and in the computer's memory during the course of using the software product for the consumer purpose(s) for which it was sold and purchased. These rights are granted to the consumer by express contractual provisions (for example, by the terms of a software license agreement) and, where the consumer is an "owner of a copy" of the software, by the statutory adaptation exemption of § 117.
>
> The necessary *technological capabilities* refer essentially to an end user's ability, by installing and running the software on a system according to the documentation, to cause the creation of processes in RAM that generate system behavior for supporting the consumer purpose(s) for which the software product was sold and purchased.

IV. APPLICATIONS AND IMPLICATIONS

This Article has had two goals in mind in introducing new concepts and approaches for defining software products and the markets in which they compete: enhancing the factual accuracy and legal sufficiency of the resulting analysis and supporting innovation driven by competition in well-functioning markets. In the following concluding discussion, this Article will indicate how these goals may be achieved in practice through the adjudicative process.

A. Syncsort v. Sequential

A recent case, *Syncsort Inc. v. Sequential Software, Inc.*, illustrates the application of the first principles approach to the definition of software products and the markets in which they compete.[455] Syncsort, a vendor of a popular software product for sorting large data sets, sued Sequential, a small competitor, for misappropriation of trade secrets.[456] Sequential answered with a counterclaim alleging that Syncsort had engaged in various tactics "to monopolize and maintain

455. 50 F. Supp. 2d 318 (D.N.J. 1999).
456. *Id*. at 321–23.

its monopoly of the UNIX [s]orting [m]arket" in violation of § 2 of the Sherman Act.[457] Sequential also alleged that the UNIX sorting market "consists of primarily three competitors: Syncsort, Innovative Routines International, Inc. and IBM,"[458] and that Syncsort's position in this market "has allowed it to create an industry standard command structure for using computer sort programs on UNIX operating systems."[459]

Syncsort moved for judgment on the pleadings with respect to Sequential's antitrust counterclaim.[460] Noting that Sequential had the burden of defining the relevant product market in its pleadings,[461] the district court described Sequential's proposed market definition as "impermissibly narrow."[462] In particular, Sequential gave no reason for excluding certain other "programs which perform sorting operations under the UNIX operating system, including Ahlsort, Aps Sort, Nsort, NitroSort and OT Sort,"[463] or for restricting the market to UNIX-based sorting programs rather than "the broader sorting market comprised of all programs which perform sorting operations or operations equivalent to sorting outside of the UNIX operating system."[464] Citing *Queen City Pizza*,[465] a widely followed Third Circuit case, the court held that "[t]he failure of Sequential to define the market in terms of reasonable interchangeability or explain the rationale underlying its narrow proposed market definition is, in itself, grounds for dismissal."[466] Accordingly, the court dismissed Sequential's antitrust counterclaim.[467]

The facts presented in the district court's opinion do not provide all of the information necessary to define a relevant product market using this Article's first principles approach. By applying the product market definition procedure in Part II.D to the facts of the case and to publicly available technical information about some of the products at issue, however, it is possible to establish that many, but not all, of the district court's objections to Sequential's proposed product market definition were warranted.

---

457. *Id*. at 326–27.
458. *Id*. at 331.
459. *Id*. at 330.
460. *Id*. at 322 (citing Fed. R. Civ. P. 12(c)).
461. *Id*. at 331 (citing Schuylkill Energy Res., Inc. v. Penn. Power & Light Co., 113 F.3d 405, 415 (1997)).
462. *Id*.
463. *Id*. at 332.
464. *Id*. at 332–33.
465. Queen City Pizza, Inc. v. Domino's Pizza, Inc., 124 F.3d 430 (3d Cir. 1997).
466. *Syncsort*, 50 F. Supp. 2d at 333.
467. *See id*. at 340.

1. The Defendant's Product

Syncsort markets Syncsort/UNIX as a software product defined by reference to accompanying software and documentation, consisting essentially of sufficient legal rights and technological capabilities to install and run the software on a UNIX system.[468]

2. Relevant Consumer Purposes

Syncsort describes Syncsort/UNIX as a "full-function, high-performance sort product for UNIX systems."[469] While Syncsort/UNIX may be used for various consumer purposes,[470] two appear to be relevant to the practices Sequential challenged. First, Syncsort/UNIX enables a user to perform the task of sorting a data file.[471] Second, by providing "an industry standard command line structure" that can be invoked by COBOL, UNIX, and other applications,[472] Syncsort/UNIX may be valued for its accompanying platform software, which must be preinstalled as a precondition to running such applications.

Sequential's suggestion that Syncsort has the power to create and impose the sort command structure of its choice on the industry seems dubious, since Syncsort has positioned Syncsort/UNIX as a product that is compatible with standard COBOL and UNIX sort commands.[473] Even if this were the case the end use of Syncsort/UNIX as platform software would not be susceptible to quality-adjusted price discrimination, because any diminution in quality with respect to that use would also adversely affect Syncsort/UNIX's ability to perform sorting tasks through the UNIX command line interface.[474] Similarly, the end use of Syncsort/UNIX for performing sorting tasks could not

468. Syncsort, Inc., Sorting Software for UNIX Systems, *at* http://www.syncsort.com/infosu.htm(last visited Dec. 4, 2004).

469. *See id*.

470. *See id*. (noting that "SyncSort provides a full set of data manipulation functions," including record selection, reformatting, join, summarization, and multiple output).

471. *See id*. ("Specifically designed for sorting and data manipulation in commercial UNIX environments, SyncSort provides a combination of speed, efficiency, the ability to handle a variety of data and file types, and versatile data manipulation features."); Paul Boal, Syncsort Application Guidelines, at 9, *at* http://www.apo49.org/~pboal/papers/ (last visited Dec. 4, 2004) ("By its name, SyncSort's strong suit is sorting data files.").

472. *See id*. (describing "[i]nvoking SyncSort with a pre-written script"); Syncsort, Inc v. Sequential Software, Inc., 50 F. Supp. 2d 318 (D.N.J. 1999) ("COBOL applications frequently use flat or indexed files, which require a great deal of sorting. SyncSort makes these sorts run faster whether they are hidden inside programs using the COBOL SORT verb or are done in shell scripts using the UNIX sort command.").

473. *See id*. (describing Syncsort's Micro Focus COBOL Sort Accelerator and MVS or VSE to UNIX Sort Converters).

474. *See, e.g.*, KAARE CHRISTIAN, THE UNIX OPERATING SYSTEM 27–42 (1983) (explaining that the UNIX system shell permits commands to be typed at the terminal or executed within programs by background and/or foreground processes).

be targeted for quality-adjusted price discrimination. Therefore, the relevant product market definition should be based on both consumer purposes together, rather than either one taken separately.

3. Represent Any Relevant Tasks as Essential Use Cases

The basic command structure for making a call to Syncsort/UNIX to perform the task of sorting a file is illustrated by Figure 3.[475]

Figure 3: Command Structure to Sort a File in Syncsort/UNIX

```
$ syncsort << EOF
01> /INFILE /temp/sorted_by_id.dat FIXED 30
02> /FIELDS id 1 CHARACTER 5,
03> val 5 CHARACTER 25
04> /KEYS val
05> /REFORMAT val, id
06> /OUTFILE /temp/sorted_by_name.dat
07> EOF
```

In this example, command 01 specifies the input file, commands 02 and 03 define the layout of the fixed-width input file, command 04 specifies the sorting criterion by identifying the field "val" as the key to sort on, command 05 starts the sorting routine, and command 06 specifies the output file.

As a specification of the sorting task, this example is far too particularized and concrete (not to mention technical) to be useful in a reasonable interchangeability inquiry. However, the same user-system interaction can readily be expressed in a simplified, generalized, abstract, technology-free, and implementation-independent form as shown in the essential use case in Figure 4.

475. *See* Boal, *supra* note 471, at 9.

Figure 4: Essential Use Case for Sorting a File According to a Desired Sort Criterion

sort file	
User Intention	**System Responsibility**
choose file to be sorted	
choose sort criterion	
start sorting	
	input file to be sorted
	sort file according to sort criterion
	output sorted file
receive sorted file	

4. Identify Products That Are Functionally Interchangeable with the Defendant's Product for the Relevant Consumer Purposes

A reasonable interchangeability analysis of all of the software products mentioned in the case is beyond the scope of this Article.[476] For illustration, however, this Article will consider whether one of these products is functionally and reasonably interchangeable with Syncsort/UNIX for the relevant consumer purposes.

According to the Nsort User Guide,[477] Nsort can "sort data sets quickly" according to a wide range of sort criteria.[478] Specifically, Nsort responds to a command line whereby a user can choose, *inter alia*, the file to be sorted and the sort criterion.[479]

Nsort therefore supports the essential use case in Figure 4, and is functionally interchangeable with Syncsort/UNIX for the purpose of supporting the task of sorting files.

The manual does not mention COBOL compatibility, but does state that Nsort can "[a]ccept command lines for the POSIX sort utility included with most UNIX implementations."[480] Accordingly, Nsort may be deemed functionally interchangeable with Syncsort/UNIX for the purpose of preinstalling platform software for

476. *See, e.g.*, text accompanying note 463.

477. ORDINAL TECHNOLOGY, NSORT USER GUIDE: RELEASE 3.2 (May 17, 2002), *available at* http://www.ordinal.com/NsortUserGuide.pdf.

478. *Id*. at 3.

479. *Id*. at 19.

480. *Id*. at 3.

UNIX programs that utilize the UNIX/POSIX command line structure.

5. List Relevant Competitive Variables

The marketing literature for Syncsort and Nsort identifies high speed and high capacity as the most salient performance metrics in their competition with other software products.[481] Other cited performance metrics include "efficiency, the ability to handle a variety of data and file types, and . . . a variety of [data manipulation] functions such as selecting, joining, grouping and extracting."[482]

Other competitive variables are highlighted by the compatibility differences between Syncsort/UNIX and Nsort. By supporting standard COBOL and UNIX sort commands, Syncsort/UNIX offers greater versatility for the purpose of preinstalling platform software than does Nsort, which appears to support only UNIX programs that use the UNIX/POSIX command line structure. Depending on user preferences among the various command line structures, this difference may also be material with respect to the end use of Syncsort/UNIX and Nsort to perform the task of sorting files.

Finally, both Syncsort/UNIX and Nsort require as a precondition that the UNIX operating system software be preinstalled on the system. Nsort further requires that the UNIX installation include support for the POSIX sort utility. This difference in preconditions may be material and should be considered as a possibly relevant competitive variable.

6. Identify Products That Are Reasonably Interchangeable with the Defendant's Product for the Relevant Consumer Purposes

Even though detailed information is not available regarding the demand response to the various competitive variables identified, there does not seem to be any clear error in the district court's conclusion that Sequential needed to justify the exclusion of Nsort from the relevant market in which Syncsort/UNIX competes. Speed and capacity differences would presumably serve to locate these two functionally interchangeable products along a continuous spectrum of price/quality

481. Ordinal Technology, Nsort: the Choice for Sorting Very Large Data Sets, *at* http://www.ordinal.com/Nsort.pdf (last visited Dec. 4, 2004) ("Nsort is a high-speed, high-capacity sort program that supports timely processing of today's large data sets. As corporate databases continue to grow, the ability to process them quickly to gain business insight is a critical competitive advantage."); Syncsort, Inc., SyncSort UNIX Product Description, *at* http://www.syncsort.com/sort/infosu.htm (last visited Dec. 4, 2004) (providing a "performance comparison" of elapsed and CPU time between Syncsort and other methods of sorting data).

482. *See* Syncsort, Inc v. Sequential Software, Inc., 50 F. Supp. 2d 318 (D.N.J. 1999).

choices for consumers.[483] Both Syncsort/UNIX and Nsort offer considerable flexibility in handling a variety of file and data types and performing a variety of data manipulation functions, and there is nothing to suggest the existence of any product features unique to either product that could give rise to a group of captive buyers. Finally, the differences between the two products' preconditions for use are probably minor and certainly narrowing. The POSIX sort utility is already included on most UNIX installations, and is increasingly being adopted as an "industry standard UNIX implementation."[484] In contrast, COBOL is increasingly being seen as an antiquated programming language.[485]

The precondition that the UNIX operating system software be preinstalled on the system, on the other hand, is much more material with respect to consumer preferences between UNIX-based and non-UNIX-based software products. The use of a non-UNIX-based software product requires as a precondition that some other operating system software be preinstalled on the system. Because of the difficulty of porting applications software from one operating system to another[486] and the relative scarcity of "dual boot" systems on which multiple operating systems have been installed,[487] it is unlikely that many consumers would respond to a quality-adjusted price increase by a hypothetical monopolist of UNIX-based sorting software products by switching to non-UNIX-based products.

7. Identify Structural Barriers to Entry

Sequential's only apparent claim of a structural barrier to entry was in its suggestion that Syncsort has the power to create and impose the sort command structure of its choice on the industry. As explained above, this claim seems dubious.

---

483. *See supra* text accompanying note 85.

484. *See, e.g.*, CARL Corp. v. Dep't of Educ., 946 P.2d 1, 13 (Haw. 1997).

485. *See, e.g.*, Michael Daly, *City's $1B Pain in the Assessment*, N.Y. Daily News, May 14, 2003, at 6 (describing a letter alleging that New York City property tax assessor's office "uses programs written in antiquated COBOL"); Andrew Kukielka, *Comment: The Millstone of Mainframes*, BANKING TECH., Feb. 23, 2004, at 38 (noting that financial institutions have traditionally relied on "1970s technology and antiquated programming languages such as Cobol"). *But see* Tim McKenna, *COBOL, RPG Not Going Away*, COMPUTING CANADA, Nov. 16, 2001 ("COBOL is taught at many institutions, but its champions are retiring . . . . Remember how hot COBOL and RPG programmers were between 1996 and Dec. 31, 1999? It will happen again, albeit less dramatically, when those grey haired programmers retire.").

486. *See, e.g.*, United States v. Microsoft Corp., 84 F. Supp. 2d 9, 18 (D.D.C. 1999) (finding that "the porting of applications from one operating system to another is a costly process").

487. *See, e.g.*, James Coates, *Kissing Blue Screen of Death Goodbye Leaves User Seeing Red*, Chi. Trib., Sept. 21, 2003, at 4 (describing dual boot installations as "too much of a hassle for most" users).

To summarize, this Article's first principles approach to product market definition in *Syncsort* indicates that Sequential's exclusion of non-UNIX-based software products from its proposed market was most likely correct, as was its more general argument that "the market is clearly segregated by operating systems."[488] On the other hand, Sequential improperly failed to explain its exclusion of other UNIX-based sorting software products from the proposed relevant market, at least with respect to Nsort. The district court was therefore correct in dismissing Sequential's antitrust counterclaim.

B. The Peer-to-Peer Controversy

The pursuit of well-functioning software product markets implies that where a court states a legal rule that has the effect of regulating the use of software products, it should try to do so without prescribing any particular software design solution. The ongoing controversy over the use of peer-to-peer network ("P2P") software products to trade copyrighted files over the Internet illustrates one potential application of this principle.

In *A&M Records, Inc. v. Napster, Inc.*[489] and *Metro-Goldwyn-Mayer Studios, Inc. v. Grokster, Ltd.*,[490] two courts in the Ninth Circuit adjudicated claims of contributory copyright infringement against vendors who developed and distributed these P2P software products. Both of the software products at issue, Napster and Grokster, supported the same two user purposes: "downloading," by enabling a user to search for and retrieve files that have been designated as shared on another user's system, and "uploading," by enabling a user to designate which files are to be shared and serving search and retrieval requests from other users, as indicated in the essential use cases in Figure 5.[491]

As the *Grokster* court noted, there was a "critical distinction" between the ways Napster and Grokster implemented these functionalities.[492] Napster's servers "indexed files from, and passed search queries and results among, all Napster users."[493] Grokster, on the other hand, did not operate any "supernode" on the network or play any part in the relaying of information across the network of Grokster users.[494] The court noted that "[u]sers connect to the [network], select which files to share, send and receive searches, and download files, all with no material involvement of [Grokster]. If [Grokster] closed [its]

488. Syncsort, Inc v. Sequential Software, Inc., 50 F. Supp. 2d 318, 333 (D.N.J. 1999).
489. 239 F.3d 1004 (9th Cir. 2001).
490. 259 F. Supp. 2d 1029 (C.D. Cal. 2003).
491. *See id*. at 1032–33.
492. *See id*. at 1040.
493. *See id*.
494. *See id*. at 1039–40.

doors and deactivated all computers within [its] control, users of [its] products could continue sharing files with little or no interruption."[495] Thus, while the Ninth Circuit had previously upheld the grant of a preliminary injunction barring Napster from engaging in or facilitating the unauthorized trading of the plaintiffs' copyrighted works,[496] the District Court for the Central District of California declined to enjoin Grokster. Holding that no contributory infringement could be found "[a]bsent evidence of active and substantial contribution to the infringement itself," the district court concluded that Grokster's provision of P2P software did not constitute contributory infringement.[497]

Figure 5: Essential Use Cases for Downloading From and Uploading to a P2P Network

download file	
User Intention	**System Responsibility**
search for file	
	find shared file on another user's system
retrieve file	
	request file
	receive file
upload file	
User Intention	**System Responsibility**
share file	[continue while connected]
	respond to other users' search and retrieval requests for this file

If the prevailing doctrine concerning the scope of copyright protection in software promotes antitrust policy by ensuring a well-functioning software product market,[498] the contrasting results of *Napster* and *Grokster* place the doctrine of contributory infringement in tension with that policy.

Using first principles, it is clear that Napster and Grokster are competing products. In a well-functioning software product market, developers of these and other P2P software products would engage in

495. *Id*. at 1041.
496. *Napster*, 239 F.3d at 1011.
497. *Grokster*, 259 F. Supp. 2d at 1043.
498. *See supra* text accompanying notes 452–453.

full and free competition with respect to user-oriented preference and performance metrics, which would very likely include the user's interest in minimizing the risk of incurring direct copyright liability.[499] After *Napster* and *Grokster*, however, a software developer's design choice must also take into account a cost that is not reflected among any *user's* preference and performance metrics: namely, the cost to the *vendor* of the potential for contributory copyright liability. In implementing the essential use cases in Figure 5, a vendor will tend to avoid designs that require the vendor to make an "active and substantial contribution" to file trading on the network, for reasons that are independent of product quality.[500] Thus, even if it happens to be the case that the most usable P2P network for file trading requires a centralized indexing server, *Grokster* will have deterred the market from producing such a design.

In reaching its conclusions, the *Grokster* court observed that "[i]n a case like this, in which Congress has not plainly marked our course, we must be circumspect in construing the scope of rights created by a legislative enactment which never calculated such a calculus of interests."[501] If innovation driven by quality competition in a well-functioning software product market is cognizable as an interest in this calculus,[502] then courts in future contributory infringement cases should reconsider the wisdom of analytical approaches that attach liability to particular design choices. The determination of liability for contributory infringement should instead be based on an analysis of the infringing and non-infringing user purposes that are supported by the defendant's product.[503]

C. Human-centric Computing

While the foregoing case studies go some way toward illustrating the breadth of application of the first principles approach, the ap-

499. In September 2003, the Recording Industry Association of America filed 261 lawsuits against individual users of P2P software products. *See* Peter K. Yu, *The Copyright Divide*, 25 CARDOZO L. REV. 331, 332 & n.6 (2003).

500. *See supra* note 497 and accompanying text.

501. 259 F. Supp. 2d at 1046.

502. *See generally* Mark A. Lemley & R. Anthony Reese, Stopping Digital Copyright Infringement Without Stopping Innovation (2003) (draft presented at Telecommunications Policy Research Conference, 2004) (examining current and proposed copyright policies regarding P2P networks with the objectives of stopping the deterrence of P2P innovators and permitting cost-effective copyright enforcement in the digital environment), *at* http://tprc.org/papers/2003/210/Stopping_Copyright_Infringement_Without_Stopping_Innovation.htm.

503. *Cf.* Sony Corp. of Am. v. Universal City Studios, Inc., 464 U.S. 417, 442 (1984) (holding that "the sale of copying equipment, like the sale of other articles of commerce, does not constitute contributory infringement if the product is widely used for legitimate, unobjectionable purposes. Indeed, it need merely be capable of substantial noninfringing uses.").

proach can also bring clarity and rigor to much more analytically intensive questions, as this Article has demonstrated elsewhere in a detailed study of the *Microsoft* tying claim.[504]

In his final book, *The Unfinished Revolution*,[505] the late MIT computer scientist Michael Dertouzos predicted that "the first step toward human-centric computing" will be the implementation of "natural interaction with machines," wherein "machine actions [will] match our human intent" and where the system will "let us carry out our intent at our level and with little effort."[506] The pursuit of well-functioning software product markets advocated in this Article may advance Dertouzos's vision of human-centric computing by setting in motion a full and free competition to offer the software product that most satisfactorily enables a system to fulfill its responsibilities in response to a user's intentions.

Design for human-centric computing is a worthwhile teleological objective for the software industry, and one well suited to antitrust jurisprudence. Despite the D.C. Circuit's dicta regarding "the undesirability of having courts oversee product design"[507] and of putting "judges and juries in the unwelcome position of designing computers,"[508] it is unlikely that the appeals court ever intended to extend antitrust immunity to all business conduct that uses product design as its instrumentality. As various commentators have observed, product designs can be used to restrain trade just as effectively as contracts, combinations, and conspiracies.[509] Courts have explicitly condemned at least those failures of product design that have given rise to legal liability.[510] By situating the evaluation of product design in a well-functioning software product market while recognizing usability metrics as objects of competition, antitrust courts can promote market-driven advances in human-centric computing.

---

504. *See* Chin, *supra* note 41.

505. MICHAEL L. DERTOUZOS, THE UNFINISHED REVOLUTION (2001).

506. *Id*. at 20–24.

507. United States v. Microsoft Corp., 147 F.3d 935, 948 (D.C. Cir. 1998).

508. *Id*. at 950 (citing IX PHILLIP E. AREEDA, ANTITRUST LAW, ¶ 1700j, at 15 (1991)).

509. *See* Lessig, *supra* note 27, at 27 ("To the extent that software manufacturers seek to bundle software products, or software functionality, they can achieve that bundle either through contract or through the design of the software itself . . . ."); *cf.* Rachel V. Leiterman, Comment, *Smart Companies, Foolish Choices? Product Designs that Harm Competitors*, 15 SANTA CLARA COMPUTER & HIGH TECH. L.J. 159, 160–61 (1999) ("Some companies, hiding behind the strong policy favoring innovation, have made changes to their products that are at least as damaging to competitors as they are innovative.").

510. *See* Leon E. Wein, *Maladjusted Contrivances and Clumsy Automation: A Jurisprudential Investigation*, 9 HARV. J.L. & TECH. 375, 383 (1996).