Apple v. Franklin: An Essay on Technology and Judicial Competence

Robert P. Merges*

Commenting on the close relationship between technological and legal developments, Grant Gilmore once wrote: "The reason for the dramatic change in English law during the second half of the eighteenth century is not far to seek. We know it as the industrial revolution."[1] His insight, although directed to trends in commercial law in eighteenth century England, is not limited to a particular legal discipline or a single historical epoch. Indeed, the many new activities and products stemming from recent technological innovations are provoking changes in American law comparable in magnitude to those experienced during the industrial revolution.

When legal rules change because of new technologies, they are usually responding to the *effects* of those technologies, not the technical developments themselves. For example, the invention of electronic funds transfer networks has brought substantial changes in the regulation of banking, somewhat akin to the developments in negotiable instrument and sale of goods law discussed by Gilmore in connection with the passage quoted above.[2] Such changes in the law require of the courts only an appreciation of a new, perhaps more rapid, means of transacting business or communicating; an understanding of the underlying technology is not necessary.

There are, however, a number of important areas where changes in the law require that judges have some appreciation of the actual technology involved. One such area is the regulation of the nuclear power industry. Here the new technology has created a class of problems fundamentally different from those posed by the regulation of conventional utilities. Accordingly, courts must devote some attention to nuclear power technology to deal effectively with these problems.[3] Another ex-

* B.S., Carnegie-Mellon University, 1981; J.D. Candidate, Yale Law School, 1985.

1. GILMORE, THE AGES OF AMERICAN LAW 5-6 (1977).

2. *Id.*

3. *See, e.g.*, Power Reactor Development Co. v. International Union of Electrical, Radio & Machine Workers, 367 U.S. 396 (1961). One commentator has stated, "The question before the court in *Power Reactor* was whether the record supported granting a construction permit for a reactor . . . at that moment in the evolution of nuclear technology. That oversight task necessitates an appreciation of the nuances of the project plan, and of the state of reactor safety." Yellin, *High Technology and the Courts: Nuclear Power and the Need for Institutional Reform*,

ample comes from attempts to protect computer programs, or software, from being wantonly copied. The courts have evaluated competing claims as to whether any existing bailiwick of the law—e.g., patents, copyrights, or trade secrets—should protect software. In the process, they have been forced to come to terms with the nature and operation of computer technology.[4]

Various commentators have questioned whether courts are competent to deal fairly and effectively with technology-related problems. Some have suggested sweeping institutional reforms such as the creation of a special court to resolve legal disputes arising out of complex technologies.[5] Others have criticized particular decisions for their technical naivete; in so doing, they have implied that courts can and should come to terms with technological issues.[6] All agree in one respect, however: disputes concerning technologies themselves will only be effectively resolved through an understanding of technical issues.

A recent case dealing with computer software suggests that the judiciary is equal to the task. That case, *Apple Computer, Inc. v. Franklin Computer Corp.*,[7] held that computer software is copyrightable. The *Apple* case is useful for more than its holding, however; it is also an instructive paradigm for an appropriate and effective judicial role in technology-centered cases.

I. *Background of the* Apple *Case*

The *Apple* case resulted from the intense competition between manufacturers of personal computers. Franklin Computer Company, an upstart challenger to the relatively well-established Apple Computer Company, began to sell an "Apple compatible" computer in the early 1980's. Franklin's chief selling point was that buyers could use software written for the popular Apple II computer with Franklin's Ace 100 model. To make this possible, Franklin employees made copies of sev-

94 HARV. L. REV. 489, 511 (1981). *See generally*, Bazelon, *Risk and Responsibility*, in SCIENCE, TECHNOLOGY, AND NATIONAL POLICY 356 (Kuehn and Porter eds. 1981).

4. For an exhaustive list of cases, *see* Davidson, *Protecting Computer Software: A Comprehensive Analysis*, 23 JURIMETRICS J. 337 (1983).

5. *See, e.g.*, Yellin, *supra* note 3, at 555 (calling for the establishment of "a committee of scientists, engineers, and lawyers to act as standing masters in complex environmental cases [footnotes omitted]"). *See generally* Tribe, *Technology Assessment and the Fourth Discontinuity: The Limits of Instrumental Rationality*, 46 S. CAL. L. REV. 617 (1973); Note, *The Role of the Courts in Technology Assessment*, 55 CORNELL L. REV. 861 (1970).

6. *See, e.g.*, Brooks, *Interrelationship of Copyright and Trade Secrets*, in SOFTWARE PROTECTION: CURRENT DEVELOPMENTS IN COPYRIGHT AND PATENT AND THEIR RELATIONSHIP TO TRADE SECRET 309, 322-328 (Brooks and Keplinger eds. 1982) (arguing that several federal trial courts have confused various issues relating to computer program copyrightability).

7. 714 F.2d 1240 (3d Cir. 1983), *rev'g* 545 F. Supp. 812 (E.D. Pa. 1982).

eral of Apple's operating system programs.[8] Apple filed a suit in federal court alleging in part that Franklin infringed its copyrights in these programs. The district court denied Apple's motion for a preliminary injunction.[9] Apple appealed on the copyright issue.

In legal terms, the case hinged on whether object code and operating system programs are copyrightable. What was really at issue, however, was the cost to Franklin of marketing a computer that could use software written for the Apple II. Franklin was trying to avoid the costs incurred by Apple in developing an Apple II operating system. By holding that Franklin had infringed Apple's copyrights, the court rewarded Apple for being a technological innovator and penalized Franklin for not developing its own operating system programs.

II. *Object Code Is Copyrightable*

The object code version of a program is a more mathematical, machine-oriented version than the original, source code version. Source code is a series of quasi-English instructions. Examples of source code programming languages include BASIC, Fortran, and Pascal. But programs are routinely translated from source code into object code,[10] so source code copyrightability alone does not help those who, like Apple, wish to protect their programs from infringement.

The *Apple* court recognized that object code must be copyrightable for software to be meaningfully protected. Accordingly, it rejected Franklin's two major arguments to the contrary.

Judge Sloviter, writing for a unanimous Third Circuit panel, rejected Franklin's first contention—that object code is uncopyrightable because it is unintelligible to humans. Relying heavily on the language and legislative histories of the 1976 Copyright Act and the 1980 amendment,[11] Sloviter declined to follow the case on which the intelligibility argument was built, *White-Smith Music Publishing Co. v. Apollo*.[12] In *White-Smith*, decided in 1908, the Supreme Court held that a piano roll was an uncopyrightable embodiment of a musical composition, since it was

8. 714 F.2d at 1245. ("Franklin did not dispute that it copied the Apple programs. Its witness admitted copying each of the works in suit from the Apple programs.")

9. 545 F. Supp. 812 (E.D. Pa. 1982), *rev'd* 714 F.2d 1240 (3rd Cir. 1983).

10. *See generally* V. HAMACHER, Z. VRANESIC & S. ZAKY, COMPUTER ORGANIZATION 301 (1978) (explaining the program translation process).

11. Copyright Act of 1976, 17 U.S.C. § 101, *et seq.*, Pub. L. No. 94-553, 90 Stat. 2541 (1978). The 1976 Act was a substantial reworking of copyright law; it was amended by the Software Copyright Act of 1980, Pub. L. No. 96-517, § 10(a), 94 Stat. 3028 (Supp. 1983), which provided that owners of copyrighted programs could make backup copies and included a general definition of a computer program.

12. 209 U.S. 1 (1908).

unintelligible to humans. Judge Sloviter addressed the intelligibility doctrine by citing a passage from the legislative history of the 1976 Act showing that the Act "was intended to obliterate distinctions engendered by *White-Smith*."[13] In addition, Judge Sloviter pointed out the logical inconsistency between the intelligibility argument and section 101 of the Copyright Act of 1976.[14] The court stated:

> [T]he definition of "computer program" adopted by Congress in the 1980 amendments [to the 1976 Act] is "sets of statements or instructions to be used *directly* or *indirectly* in a computer in order to bring about a certain result." 17 U.S.C. § 101 (emphasis added). As source code instructions must be translated into object code before the computer can act upon them, only instructions expressed in object code can be used "directly" by the computer.[15]

Judge Sloviter also rejected the defendant's second major argument, that object code is uncopyrightable because it has a utilitarian purpose.[16] Her opinion distinguished a copyrightable work such as a program from the useful functions which that work may serve. The court used the example of an instruction booklet for a complex machine. It observed that although the booklet does something useful, i.e., explains how to operate the machine, this does not preclude copyright protection.[17]

This holding should help resolve a debate that has simmered for some time in the academic literature.[18] The debate concerns the "true" nature of object code and software in general. The two positions can be summarized as: (1) object code is just another means of expression, and (2) object code is like a machine part since a computer cannot work without it. The court sided with those who advance the first position. This is encouraging, not because the court made the "correct" choice,

---

13. 714 F.2d at 1248.

14. 17 U.S.C. 101 (1978 & Supp. 1983).

15. 714 F.2d at 1248.

16. Franklin argued that because object code has a utilitarian purpose, it is protectible, if at all, only by the patent laws. Patent protection for computer programs has been discussed for a number of years. *See, e.g.*, Schmidt, *Legal Proprietary Interests in Computer Programs: The American Experience*, 21 JURIMETRICS J. 345, 355-362 (1981). Under the leading case, only programs that are part of a more comprehensive process may be patented. Diamond v. Diehr, 450 U.S. 175 (1981). This is a largely unsettled area, however; the current trend in the Court of Appeals for the Federal Circuit (which used to be called the Court of Customs and Patent Appeals) is towards a liberalization of the *Diehr* holding. *Compare In re* Taner, 681 F.2d 787 (1982) *with In re* Pardo, 684 F.2d 912 (C.C.P.A. 1982) *and In re* Abele, 684 F.2d 902 (C.C.P.A. 1982) (computer algorithm for correcting seismic wave data held unpatentable in *Taner*, while patents for similar algorithms were upheld in *Pardo* and *Abele*).

17. 714 F.2d at 1252.

18. *See* Stern, *Another Look at Copyright Protection of Software: Did the 1980 Act Do Anything for Software?*, 3 COMPUTER/LAW J. 1 (1981); Stern, *The Case of the Purloined Object Code: Can It Be Solved?*, Parts I and II, BYTE MAGAZINE, Sept., 1982 at 420 and Oct., 1982 at 210. *Contra*: Davidson, *supra* note 4; Brooks, *supra* note 6.

but because the judges felt confident enough of their understanding of object code to rule decisively.

This holding is one example of why *Apple* should serve as a paradigm. The judges on the court drew on the testimony of expert witnesses in the record and the written technical summaries presented by both sides.[19] They used these facts to form a basic understanding of what object code is and how it works. Each adversary presented information that supported his case, but the court was able to take this into account when evaluating testimony. The court's sound conclusions are not only a tribute to its perspicacity; they also demonstrate the effectiveness of the adversarial process in resolving technology-based disputes.

III. *Operating System Programs Are Copyrightable*

Besides the copyrightability of object code, the *Apple* opinion effectively resolves another complex issue: the copyrightability of operating system programs. Operating system programs are essential components in any computer system. They allow the keyboard to supply data to the microprocessor chip that is the heart of the computer. These programs also coordinate the reading of program instructions and data from external media such as floppy diskettes. Most importantly, they allow "application" programs such as financial planning models and word processors to run on a given computer.

Franklin contended that Apple's operating system programs were not copyrightable under section 102(b) of the 1976 Copyright Act,[20] which codified *Baker v. Selden*.[21] In *Baker*, the Supreme Court held that defend-

19. The opinion refers several times to the testimony of both Apple's and Franklin's expert witnesses. *See, e.g.*, 714 F.2d at 1245. The Third Circuit in general and Judge Sloviter in particular were familiar with the general technical concepts presented by the case due to their involvement with a video game case involving software issues, Williams Electronics, Inc. v. Arctic International, Inc., 685 F.2d 870 (3d Cir. 1982), and from consultation of outside sources, e.g., Note, *Copyright Protection of Computer Program Object Code*, 96 HARV. L. REV. 1723 (1983). Attorneys for both sides included appendices to their briefs that explained technical aspects of software, such as how software is written and what object code looks like. According to one of Apple's attorneys, Judge Sloviter did not appear to use sources outside the record to resolve technical issues. Telephone interview with Lawrence Scarborough, Brown & Bain, P.A., Phoenix, Arizona (Feb. 6, 1984). It should be noted that in several cases where a judge has consulted a technical source outside the record, the result has been less than satisfactory. The trial court in *Apple*, for instance, mistakenly cited a passage from a book describing programmers who write microcode to support its position on object code. 545 F. Supp. at 822 n.14 (quoting from T. KIDDER, THE SOUL OF A NEW MACHINE (1981)). Microcode is the basic set of instructions in a computer. Each object code instruction is broken down into a series of microcoded instructions. Microcode is actually "burned into" the microprocessor. Unlike object code, therefore, microcode is arguably part of the "device" and thus patentable. *See generally* Toong, *Microprocessors*, 237 SCI. AM. 146, 152 (1977).

20. 17 U.S.C. § 102(b) (1978).

21. 101 U.S. 99 (1879).

ant's bookkeeping instruction manuals did not infringe the plaintiff's copyrights in similar accounting manuals. Defendant's blank account books, the court held, used only the method, and not the exact expression, of plaintiff's books.[22] *Baker* is enshrined in Section 102(b) of the copyright statute, which prohibits copyright protection for any "process, system, or method of operation"; in short, for any idea—as opposed to its expression.[23] Franklin argued that operating system programs are part of the "process, system, or method of operation" of a computer. Therefore, they should be precluded from copyright protection under the rationale of *Baker*.

The court flatly refused to adopt this expansive reading of section 102(b). Instead, Judge Sloviter pointed out that "Apple does not seek to copyright the method which instructs the computer to perform its operating functions but only the instructions themselves."[24] Thus, Franklin may re-create, but not copy, Apple's operating system programs. The programs' method of operation is not protected, but the instructions used as a particular expression of this method may be.

Franklin also claimed that Apple's operating system programs monopolized the ideas on which they are based. Consequently, the court next considered a line of cases holding that a particular expression may not be copyrighted if it is one of a very limited number of ways to express an idea.[25] Because of the sparse evidentiary record, the court felt constrained to remand the case for further consideration on this point. But it framed the issue so that Franklin had little chance for success by stating that "If other programs can be written or created which perform the same function as an Apple's operating system program, then that program [sic] is an expression of the idea and hence copyrightable."[26] The court gave the following example:

> The idea of one of the operating system programs is . . . how to translate source code into object code. If other methods of expressing that idea are not foreclosed as a practical matter, then there is no merger [of idea and expression]. Franklin may wish to achieve total compatibility with inde-

22. *Id.* at 100.

23. 714 F.2d at 1250 (citing 17 U.S.C. § 102(b)). This language alludes to the distinction between copyright and patent protection for computer programs. *See generally* note 16, *supra*.

24. 714 F.2d at 1251.

25. 714 F.2d at 1253 (citing Morrissey v. Proctor & Gamble Co., 379 F.2d 675 (1st Cir. 1967) and Freedman v. Grolier Enterprises, Inc., 179 U.S.P.Q. 476 (S.D.N.Y. 1973)). The *Morrissey* case involved an attempt by the plaintiff to copyright a set of contest rules; the court held that the rules could not be copyrighted, since this would amount to copyrighting the game itself. *Freedman* involved an attempt, similarly rejected, to copyright a set of bridge cards containing the numerical point total for each card. Note that the merger cases are but one illustration of the general copyright principle forbidding protection of an idea, as opposed to its expression. *See* Baker v. Selden, *supra* note 21.

26. 714 F.2d at 1253.

> pendently developed application programs written for the Apple II, but that is a commercial and competitive objective which does not enter into the somewhat metaphysical issue of whether particular ideas and expressions have merged.[27]

Subsequent events proved the court right. In its settlement agreement with Apple, Franklin conceded that it had developed its own set of operating system programs to take the place of Apple's.[28] This demonstrates that the ideas embodied in Apple's programs had indeed not merged with their expressions.

The merger issue is clothed in difficult technical arguments. The development of programs that "emulate" Apple's was no doubt a difficult and expensive task. At the time of the court's ruling, there was even considerable debate among programmers as to whether it was possible at all.[29] The court, however, did not get lost in this technical thicket, but ruled in accord with a fundamental precept of the copyright system: innovators should be rewarded.

This ruling illustrates two points. It shows again that courts are capable of deciding issues having a technical component. Also, it points out the wisdom of having the courts decide both the technical and legal questions embodied in a dispute. The *Apple* court was able to consider all aspects of the dispute before it—legal and technical. It could address the economic underpinnings of the case in the face of uncertainty as to the accuracy of its technical determination.[30] As it turned out, the judges were both technically correct and fair.

IV. *Conclusion*

Apple demonstrates that the judiciary has the capacity to resolve disputes arising from new technologies. The *Apple* court exhibited two characteristics needed to resolve such disputes: a willingness to learn about the technologies involved, and an awareness of the ways in which

---

27. *Id.*

28. *Franklin and Apple Settlement*, N.Y. Times, Jan. 7, 1984, at D1, col. 2 (reporting that besides paying Apple $2.5 million in damages, Franklin agreed to "substitute another [operating] system [which it has developed] that it said would still make it possible for Franklin machines to use the programs and information designed for Apple computers.")

29. *Compare Davidson*, *supra* note 4, at 371, *with* Reply Brief for Appellant at 19, Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983). Davidson asserts that at least some of Apple's operating system programs cannot be emulated; Apple's experts claimed that all of Apple's operating system programs could be emulated.

30. It is interesting to note that Apple would probably have lost if a separate tribunal had decided that, as a strictly technical matter, it was impossible for Franklin to develop operating system programs to take the place of Apple's. Such a binding ruling could conceivably be made under the kind of bifurcated institutional arrangement envisioned by Yellin, *supra* note 3.

the legal and technical aspects of the disputes are intertwined. Accordingly, the decision stands as an exemplar of technical competence and legal acuity.