Paragraph 52: A Window into Judge Jackson's Findings of Fact

Jonathan Band
Morrison & Foerster LLP

Every once in a while a passage appears in a judicial opinion which succinctly captures the essence of an exceedingly complex issue. Paragraph 52 of Judge Jackson's findings of fact in the *Microsoft* case is such a passage. In this one paragraph, Judge Jackson lays to rest many of the vexing factual questions relating to software interoperability in general and reverse engineering in particular. Because interoperability plays such a crucial role in permitting competition in the networked environment, these insights are of growing importance as electronic commerce expands.

I. The Context of Paragraph 52.

Judge Jackson's 207 page Findings of Fact, issued on November 5, 1999, center on Microsoft's efforts to protect the Application Program Interface (API) barrier to entry. According to Judge Jackson, the large number of application programs written to the APIs exposed by Windows hindered competition from rival operating systems, such as Apple's Mac OS, which could not run these applications. The Findings of Fact recite in great detail Microsoft's efforts to preserve the API barrier to entry by preventing the broad adoption of "middleware," such as Netscape's Navigator, which would run on top of Windows and expose a different and attractive set of APIs to independent software developers.

A critical link in Judge Jackson's analysis was the inability of a competitor to develop an operating system that exposed the same APIs as Windows. In paragraph 46, Judge Jackson recounts IBM's unsuccessful effort to do so: "In late 1994, IBM introduced its Intel-compatible OS/2 Warp operating system and spent millions of dollars in an effort to attract ISVs [independent software vendors] to develop applications for OS/2 and *in an attempt to reverse engineer, or 'clone,' part of the Windows API set.* Despite these efforts, IBM could obtain neither significant market share nor ISV support for OS/2 Warp." In Paragraph 52, he draws conclusions from IBM's experience on the feasibility of reverse engineering Windows to uncover the APIs, and then replicating them in a competing operating system -- a process he refers to as "cloning."

II. The Text of Paragraph 52

Paragraph 52 reads as follows:

"Theoretically, the developer of a non-Microsoft, Intel-compatible PC operating system could circumvent the applications barrier to entry by cloning the APIs exposed by the 32-bit versions of Windows (Windows 9x and Windows NT). Applications written for Windows would then also run on the rival operating system, and consumers could use the rival system confident in that knowledge. Translating this theory into practice is virtually impossible, however. First of all, cloning the thousands of APIs already exposed by Windows would be an enormously expensive undertaking. More daunting is the fact that Microsoft continually adds APIs to Windows through updates and new versions. By the time a rival finished cloning the APIs currently in existence, Windows would have exposed a multitude of new ones. Since the rival would never catch up, it would never be able to assure consumers that its operating system would run all of the applications

written for Windows. IBM discovered this to its dismay in the mid-1990s when it failed, despite a massive investment, to clone a sufficiently large part of the 32-bit Windows APIs. In short, attempting to clone the 32-bit Windows APIs is such an expensive, uncertain undertaking that it fails to present a practical option for a would-be competitor to Windows."

III. The Lessons of Paragraph 52.

As noted above, Judge Jackson understands the "cloning" of the Windows APIs to include the reverse engineering of Windows to uncover its APIs. Accordingly, in paragraph 52 Judge Jackson is saying that the process of reverse engineering Windows to uncover its APIs, and their subsequent incorporation in a competing operating system, is expensive and time consuming -- so time consuming, in fact, that the competitor could never keep up with Microsoft. This seemingly simple observation contains many implicit and explicit lessons for the continuing debate about interoperability and competition in the information technology industry.

1. The information necessary for interoperability is not readily made available by the vendor. In numerous fora -- most recently Australia -- Microsoft and other large software companies have argued that reverse engineering was not necessary for achieving interoperability because all the needed interface information could easily be licensed.[1] As IBM's experience recounted in Paragraph 52 clearly demonstrates, this information is not made available to would-be competitors. Microsoft may have been willing to license some of this information to an ISV developing an application program designed to run on Windows, but it was not willing to license the full API set to the developer of a competing product -- in this case, IBM.

2. Reverse engineering is an expensive and time consuming process. In the global debates over reverse engineering, opponents of the practice have argued that it could be accomplished by the push of a button. Indeed, these opponents contended that reverse engineering was so easy that it facilitated disguised software piracy. A pirate could "decompile" the object code of the program into a higher level language, rename the variables and make other slight alterations, and then recompile the program.[2] This recompiled program would look sufficiently different from the original program so as to defeat claims of copyright infringement. Paragraph 52 debunks this "disguised piracy" myth. Reverse engineering is so time consuming and expensive that it would never be employed for disguised piracy. In fact, in the IBM case, it was so costly that it prevented IBM from developing a fully compatible operating system.

3. Most legitimate developers engage in reverse engineering. In the late 1980s and early 1990s, IBM was one of the major opponents of the legalization reverse

---

[1] *See* Jonathan Band, *Gunboat Diplomacy on the Pearl River: The Tortuous History of the Software Reverse Engineering Provisions of Hong Kong's New Copyright Bill*, The Computer Lawyer at 8, 10 (February 1998) for discussion of the Business Software Alliance presentation to Hong Kong's Legislative Council, Bills Committee on the Copyright Bill (April 18, 1997).

[2] *Id*.

engineering.[3] And yet as Paragraph 52 shows, in 1994 it reverse engineered Windows. Further, there is ample evidence that Microsoft has reverse engineered competitors' products -- most recently, America Online's Instant Messaging protocols.[4] Reverse engineering is a basic software development tool.

4. Copyright law does not prevent reverse engineering or interoperability. As important as what Judge Jackson does say in Paragraph 52 is what he does not say. Specifically, he does not say that copyright law prevented the "cloning" of the Windows APIs. This, of course, is consistent with the Ninth Circuit ruling in *Sega v. Accolade*, 977 F.2d 1510 (9th Cir. 1992), that the copying incidental to decompilation was a fair use so long as the decompilation was the only way to uncover the information and the decompilation was performed for a legitimate purpose. It is also consistent with the line of cases following the Second Circuit's *Computer Associates v. Altai*, 982 F.2d 693 (2d Cir. 1992), refusing to extend copyright protection to interface specifications. (Judge Jackson relied on these cases in an opinion rejecting Microsoft's motion for summary judgement, where Microsoft argued that its copyright in Windows entitled it impose whatever terms it wished in its Windows license agreements.[5]**)** However, the law in this area might not be as settled as it seemed: last year a district court in California held that Connectix's reverse engineering of the Sony PlayStation violated Sony's copyrights. *Sony Computer Entertainment Inc. v. Connectix Corp.*, 48 F. Supp. 2d 1212 (N.D. Cal. 1999). The case is now on appeal before the Ninth Circuit.

5. Interface information is essential to competition in the information technology industry. Although Judge Jackson devotes only one paragraph to the "cloning" issue, without question it forms a critical factual predicate for the entire case. As he notes, "[t] heoretically, the developer of a non-Microsoft, Intel-compatible PC operating system could circumvent the applications barrier to entry," and thereby undermine Microsoft's monopoly. However, "[t]ranslating this theory into practice is virtually impossible" because of the difficulty of reverse engineering a program as complex as Windows. In other words, Microsoft's monopoly in Windows is sustained in large measure by the difficulty in uncovering information concerning the Windows' APIs. Interface information is the key to competition in the information technology industry, and there is no competition in the PC operating system market because Microsoft maintains tight control over the needed information.

It is intriguing to consider how different the computer industry would be today had IBM succeeded in its Windows reverse engineering effort. For a historical precendent, one can consider Phoenix Technologies' reverse engineering of the IBM PC's basic input/out system (BIOS) in the 1980s. With this information, Phoenix developed a compatible BIOS which it made available to other hardware manufacturers. This

---

[3] *See* Jonathan Band and Masanobu Katoh, *Interoperability Down Under: The Australian Copyright Law Review Committee*, The Computer Lawyer at 20 (July 1995) for a discussion of IBM's reverse engineering position in Australia; *see also* Jonathan Band and Masanobu Katoh, *Interfaces on Trial*, at 18-28 (Westview Press 1995).

[4] Saul Hansell, *In Cyberspace, Rivals Skirmish Over Messaging*, New York Times, July 24, 1999 at A1.

[5] *See* Jonathan Band and Taro Isshiki, *Peace at Last? Executive and Legislative Branch Endorsement of Recent Software Copyright Case Law*, The Computer Lawyer at 1 (February 1999).

contributed significantly to the emergence of the IBM compatible PC industry and the ultimate rise of companies such as Dell and Gateway.

IV. The Remedy Which Follows From Paragraph 52.

The difficulty of uncovering the critical API information suggests an obvious remedy in this case, in the likely event Judge Jackson finds Microsoft liable for violating the antitrust laws -- ordering Microsoft to disclose the Windows source code. While it would still require significant effort by a competitor to discern a complete set of the Windows APIs from the Windows source code, it would be faster and less expensive than examining the object code for the same information. The disclosure of the Windows source code is one of the remedies that has been suggested in the press coverage of the case, and it probably is the remedy most likely to produce meaningful competition in the relevant market. It has the added advantages of not requiring significant judicial involvement or the disruptive restructuring of Microsoft. Microsoft would still hold the copyright in Windows, thereby preventing a competitor from copying lines of code or other protected expression. In essence, Microsoft would be placed in the same position as Honda, whose cars can be bought and carefully examined by General Motors. In that event, Microsoft, like Honda, would achieve continued success only through innovation, quality, and service.