

## Pursuing a Remedy in Microsoft: The Declining Need for Centralized Coordination in a Networked World

by

RANDAL C. PICKER\*

In the pre-networked world, Windows played the central role in coordinating the sharing of software. The rise of the network changes how software should be distributed and changes the role of Windows in software coordination. There is less of a need for mandatory incorporation of software into Windows, as decentralized distribution and coordination is now possible. In impermissibly maintaining its monopoly, Microsoft distorted the channels for software distribution and added software to Windows for the purpose of raising the cost of distribution of rival software. A proportionate Microsoft remedy should address that distributional distortion and seek to prevent future distortions. The article suggests such a remedy. (JEL: K 21, L 4)

### 1 Introduction

The *en banc* decision of the D.C. Circuit in *United States v. Microsoft*, 253 F. 3d 34 (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001]), partially overturning Judge Thomas Penfield Jackson's findings of liability and fully rejecting his proposed break-up remedy sets the stage for a full reset. As discussed below, the appellate court upheld in large measure the lower court's finding of monopoly maintenance in violation of Section 2 of the Sherman Act; overturned the finding of attempted monopolization of the browser market; and re-

---

\* At the conference, this paper was given under the name of "Understanding *Microsoft*: The Decline of Centralized Coordination in a Peer-to-Peer World." A subsequent version dated as of July 28, 2001 was generally circulated. This article reflects the state of the world as of that date. Much has happened since then – including a partial settlement – but there is no good way to evolve the paper with each turn in the case. I thank conference commentators Christoph Engel and Ulrich Kamecke, as well as Dennis Carlton, Richard Epstein, Jack Goldsmith, Mark Lemley, Larry Lessig, Douglas G. Lichtman, Michael O'Donnell, Michael Waldman, and participants at a University of Chicago Law School workshop. I also thank the Sarah Scaife Foundation and the Lynde & Harry Bradley Foundation for their generous research support.

manded for a rule of reason analysis of Microsoft's alleged tying of Internet Explorer (henceforth IE) to Windows. The liability ruling alone would have sufficed to require remand for reconsideration of the appropriate remedy, but the appeals court also found the remedy process deficient and found that Judge Jackson's out-of-court statements gave the appearance of bias. These provided additional sufficient reasons for overturning the remedy, and the extra-judicial statements sufficed to give Judge Jackson the boot on the remand.

The ruling therefore means that a new federal district court judge will need to conduct a hearing on remedy, and the federal government and the plaintiff States will need to consider whether to pursue the tying claim on remand. Microsoft has already indicated that it would like to settle, on the right terms of course, and there is some reason to think that the Bush Antitrust Division would be willing as well. (The States are another matter, though New Mexico has already jumped in and settled.)

Given the reset, what should happen? How should we think about an appropriate remedy for the monopoly maintenance violation? How should the tying claim play out under a rule of reason analysis? What provisions should be central to any possible settlement? This paper offers thoughts on these questions. It rests on three basic premises:

(A) *Software Sharing is the Norm*. By this I mean that one piece of software will look to another piece of software for some of its functionality. Without intending too much by these words, one "application" might look to another "application" or an "application" might look to the "operating system" for this shared code.

(B) *Software Sharing Must be Coordinated*. Programmers make the initial sharing decision through their design decisions. A program could be self-contained and rely on nothing else. That could be accomplished either through writing all of the relevant code or by bundling the new code with the code to be shared. Bundling by the programmer is one form of coordination. If full bundling turns out not to be sensible – and it often will not – coordination will need to be achieved through other means. For if a program anticipates that another program will be present, and it is not, the first program will fail. A person buying the first program must therefore have access to the second program. Put differently, the first program must be coordinated with the second program.

(C) *The Internet has Altered the Cost of Coordination*. The rise of the Internet, and more generally, networked computers, has drastically changed the cost of software coordination. More precisely, decentralized coordination was prohibitively costly prior to the Internet. This forced us to rely instead on centralized coordination – in many ways, the Windows operating system (henceforth OS). Now, networked coordination or peer-to-peer coordination – of both distribution and payment – is possible.

These premises push me towards the following conclusions:

(a) *Windows as Hub*. Centralized coordination was achieved most straightforwardly through direct incorporation of the shared code into the operating system and naturally accounts for much of the growth of the size and scope of Windows.

Pre-Internet, this ensured that required shared software would be available for programs added later. It also ensured that only one payment was made for this shared software.

(b) *Incorporating IE*. A decision to incorporate a browser would have been perfectly consistent with this prior practice.

(c) *Understanding Tying*. In this framework, the question of tying turns not on some elusive conception of “separate” or “integrated” software, but rather on the assignment of the role of coordinator for shared software. Again, in the pre-network era, such an assignment would flow naturally to the hub.

(d) *The Network Changes Everything*. This changes dramatically in the network era. Decentralized coordination of shared software is increasingly possible. This would make it possible to radically resize the OS. We can easily envision a new, smaller centralized OS, supplemented by additional software acquired through networked coordination.

We have therefore reached a breakpoint. The prior need for centralized coordination should make us skeptical about the government’s pending tying claim against Microsoft. If that is right, the plaintiffs are left “only” with the core ruling on monopoly maintenance. We have a blank slate on the remedy for that violation. The suggested shift in coordination possibilities should guide us in crafting that remedy. Given the finding of monopoly maintenance, the government may be able to jump in now and facilitate the shift from centralized coordination to networked coordination. To date, I do not think that the government has understood that its remedies should be directed at the suggested coordination shift, though some of the conduct remedies at least might have that consequence; the suggested and still possible break-up almost certainly would not.

On a going forward basis, we should have substantial doubts about Microsoft’s need to incorporate new code into the OS *on a mandatory basis*. Given Microsoft’s now demonstrated willingness to use its control over Windows to preserve its Windows monopoly, we need to construct a framework to prevent further abuses of the sort seen in this case. That framework also needs to reflect the substantial difficulties of identifying in advance precisely what new features might emerge as potential competitors to the OS.

In Section 6 the remedy is set forth in detail out the suggested remedy in detail, but a remedy should consist of five central features:

(1) *DI Visibility Flexibility*. There should be no mandatory icons on the Windows desktop or spots reserved in the Start Menu or its equivalent. Distribution intermediaries (DIs) – original equipment manufacturers (OEMs) and others – would have complete freedom to add or subtract icons from the interface.

(2) *Mandatory Versioning*. During the remedial period – see (5) below – Microsoft should be required to issue Windows versions with and without any new middleware that it adds to Windows. For this to be meaningful, this means that the baseline Windows XP could not include instant messaging (henceforth IM) and the Windows media player, but that those features could be included in an

upgraded version of Windows XP. Microsoft could charge the same price for basic and deluxe versions of Windows.

(3) *Direct Distribution Only Period.* For a moratorium period, perhaps of six months to 2 years, Microsoft should be able to distribute through distributional intermediaries only the baseline Windows without the new middleware. Microsoft would be able to distribute such middleware only through downloading from its website or through direct distribution of CDs to end-users.

(4) *DI Neutrality.* After the moratorium period, we should rely on competition among software producers and others for DI “shelf space” – hard disk space for OEMs, web presence for Internet service providers (henceforth ISPs) and others – to control software distribution abuses, and should only seek to control possible abuses of market power by limiting conditions that Microsoft can impose on DIs. Mandatory versioning would be continued during this second period.

(5) *Sunset.* All of these provisions should sunset, perhaps after a period of three years.

In two sentences, Microsoft distorted the distribution of software through its monopoly maintenance; the right remedy for that it is to distort back against Microsoft for a period, and then make sure that Microsoft is not able to distort distribution again. The presence of the always-on network means that we can do this while minimizing possible harms to consumers from introducing a corrective distortion. That said, the direct distribution remedy may impose interim losses on consumers for uncertain later consumer gains. We need to be sure we are willing to make that investment, and that turns in part on whether we think that we can restore competition without the direct distribution remedy.

## 2 A Quick Introduction to the Case

In *United States v. Microsoft* (see U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999] for “Findings of Fact,” [2000a] for “Conclusions of Law,” and [2000b] for “Final Judgment”), Judge Jackson reached four legal conclusions:

- (i) Microsoft maintained its Windows monopoly by anticompetitive means in violation of Section 2 of the Sherman Act;
- (ii) Microsoft attempted to monopolize the Web browser market, also in violation of Section 2;
- (iii) Microsoft impermissibly tied Internet Explorer (IE) to Windows in violation of Section 1 of the Sherman Act; and
- (iv) Microsoft did not engage in illegal exclusive dealing.

These conclusions obviously depended on subsidiary legal findings, including those covering market definition.

In many ways, the three violations found by Judge Jackson turn on the legitimacy of Microsoft’s decision to distribute IE with Windows. Regardless of what one

thinks of the legal conclusions, Judge Jackson was certainly correct in seeing Microsoft's decision as having important consequences for competition between Microsoft and Netscape. In his now infamous May 26, 1995 memo (GATES [1995, p. 4]) on "The Internet Tidal Wave," Bill Gates feared that Netscape was playing a platform strategy that would "commoditize" the OS:

"A new competitor 'born' on the Internet is Netscape. Their browser is dominant, with 70% usage share, allowing them to determine which network extensions will catch on. They are pursuing a multi-platform strategy where they move the key API [application programming interface] into the client to commoditize the underlying operating system."

Developers would write software directly to the Netscape layer and would cease to write to the underlying OS. Control would shift from Microsoft to Netscape. Microsoft could drive a stake in the heart of that strategy by fragmenting the browser market, and did so when IE captured a substantial market share. The attempted monopolization claim turns largely on the details of a June, 1995 meeting among assorted Netscape and Microsoft employees, plus Microsoft's success in displacing Navigator with IE. And the decision to distribute IE with Windows makes it possible to allege that they have been tied together.

The D.C. Circuit's through-going review of Judge Jackson's opinion left only the monopoly maintenance claim intact. The claim of attempted monopolization of the browser market was killed off completely: the lower court's finding was overturned and the claim was not sent back to the lower court for reconsideration. The tying claim was overturned and remanded for consideration under rule of reason analysis. Each of these will be discussed in more detail below.

### *3 Shared Software and Software Coordination*

Before doing so, focus on two standard software sharing situations. One possibility is that two pieces of software will have a common component, best captured as a file with the same name. (This is no guarantee, of course, that files having the same name have the same content, but ignore that here.) Another way in which software can share code is by having one application rely on the expected presence of other software. This may be as basic as relying on the printer drivers provided with Windows in your word processing application. We could think of these as software externalities.

For ease of discussion, label the first version of sharing the common component case, the second the expected component case, and start with the first. Installation – and deinstallation, as we need to consider both of these – creates the risk of corruption if the common component is actually shared by the two packages. Package 1 is added to the system; package 2 is added thereafter. If both packages use precisely the same version of the shared component and look to the same file path for the component, we have avoided some basic problems. If not, we may overwrite an old component with a new component (or vice versa) and break one of the

packages. Deinstallation creates similar issues: If package 2 uninstalls the common component, package 1 will not work. In the software business, this is referred to as “DLL hell,” as many of the relevant files are stored in files known as dynamic link libraries (DLLs). In the expected component case, again, the software may not work at all and certainly will not work as planned if the expected component is missing. You can write words on the screen but you cannot print, if the printer drivers that the word processor looks for are missing.

In both of these cases, we have a natural alternative: have each application bring all of its components with it. Indeed, this has been a common practice on the Unix platform (ANDERSON [2000]). This of course would increase the size of each application, quite dramatically in some cases. It also poses interesting licensing questions. For example, suppose that Microsoft licensed Windows to every application developer who wanted to rely on the Windows infrastructure for its applications. Windows and the application would be distributed together, thereby ensuring the necessary OS for running the application. This poses pricing problems. Do I have to rebuy the OS with each application? Should the application test for the presence of the OS already, and not charge you for the OS if you already have it? How do we make that system work when software is distributed on CDs and floppies and not over the Internet?

### 3.1 *Minimizing the Operating System (OS) and the Windows Installer (WI)*

Try a thought experiment: Suppose we wanted to minimize the OS. How far could we shrink and what would be the consequences of doing this?<sup>1</sup> We can get a handle on this by delving into an obscure piece of Windows 2000, a tool called the WI. The WI (MICROSOFT [1999]) implements key related concepts – just-in-time software, installation on demand (IOD), and “advertisement” – which in turn help to make clear what the minimalist OS might look like.

As you might guess, absent perverse naming, the WI would be a tool for installing software, and, of course, it is. It establishes a unified framework for installation – and deinstallation – of new software on a computer running the Windows 2000 OS. One of the key issues in adding and removing software is the management of shared software. Software sharing, of course, in some sense defines the software business. When we think of an OS as serving as a platform for a large number of application programs, we mean that the applications share that OS. Each application need not come with its own OS but instead can rely on the existence of certain basic functionalities from the OS.

WI creates new possibilities in installation. The standard binary choice – the component is installed or it is not installed – is replaced with a richer structure that allows installation of software as and when needed. Features are “advertised,”

---

<sup>1</sup> OS theorists might object that we should focus on the microkernel when looking for the smallest element of an OS, though I gather this is a contested issue (see, e.g., STALLINGS [2001, pp. 172–178], TANENBAUM [2001, p. 62]). I intend the OS notion in the text to be less precise and more conceptual, I hope without doing serious damage to language.

meaning that the software displays the feature in question as available for use, but the actual software to implement the feature is not already installed. When the user tries to use the feature, the installer obtains the software, usually by asking the user to insert an original Windows 2000 CD – and loads it at that point. This is IOD or just-in-time software.

IOD works at the feature level. For example, I recently received an email from an Israeli student, which, in turn, used a Hebrew font. It was not already on my machine, but the computer realized that, and asked me to load it. Advertising could work at the application level itself, so that when I first try to use an application, say PowerPoint, the entire application is loaded.

In this framework, we can naturally distinguish advertising (or visibility), which relates to whether a particular feature is visible to the user, and presence, which relates to whether or not software has been installed on the computer or is directly available for use across a network. Software can be visible without being present, indeed, that is the core idea of installation on demand. Software could also be present without being visible: an end-user would not see how to invoke the software directly but a software developer could rely on the presence of the software for sharing when designing complementary software. We should also distinguish between presence and ownership: software could be installed but unowned; first use of the software might trigger a request for payment to be made across the network.

We can now define the minimalist OS. It has no functionality other than advertising features and applications. It also will maintain a database of the locations of advertised software. You want to browse the Internet? You click on the browser icon on the desktop, and install a browser (which browser?). You want help understanding the desktop – go to the advertised help feature, which in turn will require the loading of content and a display engine. Email? Word processing? It all operates the same way. The minimalist OS does little more than provide access points to features and applications and uses the WI to manage the installation on a demand basis.

### *3.2 OEMs and OS Fragmentation*

We should examine how software was distributed in the pre-network era. Software was installed by a computer seller – an original equipment manufacturer or OEM in the trade – at the time of the purchase of a new computer, and by end-users off of CDs and floppies when new software was purchased. OEMs dealt directly with software vendors, such as Microsoft and others. Microsoft defends its decisions regarding IE from the broad position that software developers, not courts, need to be in charge of innovation. Microsoft also argues more narrowly that by bundling together software in the OS and insisting that each user have the same platform, it creates standards that redound to the benefit of all consumers. Software developers, at least as a group, rely on the presence of all of the components of the OS in developing their applications. Losing a component means breaking an application.

Allowing OEMs to install some pieces of the OS and excluded others will result in a “fragmented” installed base for the OS. With a fragmented OS – with multiple flavors – application developers must create multiple versions of their application or ignore groups of potential users.

Examine this concern in the framework of our two basic software sharing scenarios: common components and expected components. Consumers do not want to pay twice for the same piece of software. In the common component case – if managed correctly – the consumer needs only one copy of the component and does not want to pay two software houses for it. We now need to coordinate distribution of the single copy of the component. With an extremely fine-grained setup for distributing software, we could do this in a number of ways. All software is installed over the network. When the consumer purchases software package *a*, the installer searches the consumer’s computer to see if any components required for the package are already installed. Finding none, a full installation is done for package *a* and the full price is charged for the software. Time passes and the consumer seeks to buy software package *b*. The installer searches for common components and finds the components that overlap between *a* and *b*. Package *b* is installed, but only the new components, and the price charged for the software reflects the fact that the common components were not sold in the second installation.

This requires an enormous amount of coordination. In the pre-network era – pre-Internet really – there was no way to engage in this sort of real-time coordination of distribution and payment. Software was sold on CDs and the price for that software was paid in exchange for and at the time of the physical transfers of the CD. Networked coordination of the sort possible in the network software distribution model was difficult. It would have been possible to use a lock and key system, requiring a telephone call and payment to purchase a code to unlock the software, but this is somewhat clunky. Instead, we needed to rely on centralized coordination of the hub-and-spoke variety with Windows at the hub. We avoided multiple purchases of the same software by assigning the sale of common components to the central hub. These components were distributed as part of the OS, so as to facilitate coordination, rather than as somehow being an inherently necessary part of the OS.

In the pre-Internet era, we could think of OEMs as playing the key role in software coordination. In the abstract, OEMs could treat Windows as an input, and they could subtract at will, and then add other software. In the pre-network era, this would have worked poorly. Software was distributed on floppies or CDs, and not over the network. If a software developer anticipated that software would be present on the consumer’s machine, the developer would not bundle that software with its software. When that supposition turned out to be wrong, the new software would crash. Absent a network connection, there would be no way to add that component on an as-needed basis. In that world, fragmentation of the OS is a serious concern.

We have been discussing the common component case with sharing between peer applications. The expected component case is more hierarchical, where the

second piece of software is more fundamentally dependent on the first. In the expected component case, if that component is missing, WI just reaches out to get the necessary component. If an OEM has not installed the component that the application needs, the WI solves that problem by adding the component as needed. Again, with always-on networks, this is easy to solve. The rise of the ubiquitous network changes the costs of coordination dramatically. Networked component sharing becomes possible in a technical sense, and the real question relates to the cost of coordinating the markets in these transactions.

It is even easy to address with standalone computers, so long as the Windows CD distributed with the computer contains the missing component, even if the OEM chose not to install it originally. Of course, if we think of the OEM as buying some subset of the Windows OS and the end-user receiving only that subset on the CD, we then have a problem. The end-user will need to enter into a separate transaction with Microsoft to buy the needed component, and, again, pre-network, that transaction is quite expensive. Microsoft would need to prepare a missing components CD for each OEM, and arrange to distribute those through retailers.

The alternative is that the Windows CD contains all of the components, even if OEMs can choose to make visible only a subset of that Windows functionality. IOD would then add missing functionality on as needed basis as an application looked for that functionality. Of course in that framework, OEMs would seek to pay for as little of Windows as possible, given that consumers would receive the entire package anyway.

#### *4 Monopoly Maintenance and the Limits of Centralized Coordination of Software Sharing*

In the pre-network era, the OS emerges naturally as the key device for implementing software sharing coordination. Microsoft policed that coordination by insisting on full availability of Windows. That of course gave Microsoft enormous power, as Windows itself emerged as the dominant vehicle for software distribution. In the pre-network period, the need for centralized coordination gave Microsoft a bottleneck power, the ability to include and at least partially exclude simply through inclusion or exclusion from Windows.

Obviously, as the central coordinator, Microsoft could just include IE with Windows, which in turn gives rise to the tying claim in the case. But Microsoft went beyond this by giving others access to Windows in an effort to influence their use of Navigator. Microsoft also used its control over software design to disadvantage Netscape. This is what drives the monopoly maintenance claim in the case. Microsoft used its control over access to Windows not merely to continue to coordinate software sharing but to affirmatively prevent Netscape from succeeding. The irony here is that inclusion of IE into Windows might have sufficed to protect the applications barrier to entry as conceived of in the case, and the contractual efforts aimed directly at Netscape were probably overkill.

The standard approach to monopolization under Section 2 of the Sherman Act was set forth by the Supreme Court in *United States v. Grinnell Corp.*, 384 U.S. 563, 570–571 (1966) (see also <http://www.ripon.edu/faculty/bowenj/antitrust/grinnell.htm>):

“The offense of monopoly under Section 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.”

The appeals court upheld “in its entirety” the lower court’s conclusion that Microsoft exercised monopoly power in the market for Intel-compatible computers (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 15]).

Monopoly in hand, the monopoly maintenance claim is straightforward and turns on Microsoft’s role as centralized coordinator. The maintenance claim itself rested on four separate allegations: (1) contractual dealings with a blizzard of abbreviations – OEMs (original equipment manufacturers), IAPs (Internet access providers), ICPs (Internet content providers) and ISVs (independent software vendors) – and those with Apple and Intel; (2) integration of IE and Windows; (3) efforts to subvert Java; and (4) Microsoft’s course of conduct as a whole (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 25f.]). The appeals court rejected the latter claim as insufficiently developed (pp. 58f.), but upheld, in whole or in part, the lower court’s ruling on the other three allegations.

#### 4.1 Contractual Dealings

The appeals court in the main upheld the lower court’s analysis of the contracts:

*Agreements with Original Equipment Manufacturers (OEMs)*. Microsoft barred OEMs from changing desktop icons or folders or the menu entries under the “Start” Menu. Microsoft also did not allow the OEMs to alter the sequence of screens viewed when the computer booted initially, nor did it allow the OEMs to substitute another interface shell for Windows itself (here thinking of Windows, as it was originally, as an overlay for DOS). The appeals court found each of these restrictions, save for the last, anticompetitive without any countervailing justification. For example, Microsoft’s insistence that the icon for IE be visible on the desktop made it less likely that OEMs would install Navigator as well, since, as the lower court found and Microsoft did not successfully challenge on appeal, two icons might create consumer confusion, and confused consumers call their OEMs first. Alterations in the boot sequence might make it possible for OEMs to promote rival browsers. In separating icon removal or alteration of the boot sequence from wholesale replacement of the Windows shell, the appeals court emphasized that the latter made substantial alterations to Microsoft’s copyrighted work, while the former made only minor changes.

*Agreements with Internet Access Providers (IAPs).* The court of appeals blessed Microsoft's efforts to attract IAPs to IE by giving it and the IE Access Kit to them gratis and by paying IAPs for getting their users to adopt IE. These agreements became anticompetitive when they included direct limits on loading Navigator, as this, in the view of the court, partially foreclosed one of the two key modes of distributing browser software. Microsoft's failure to offer a procompetitive justification for these limits sealed its fate.

*Deals with Internet Content Providers (ICPs) and Independent Software Vendors (ISVs).* The court of appeals rejected liability for Microsoft's deals with ICPs, as these agreements were not shown to have had a substantial effect upon competition. The court reached a contrary conclusion for Microsoft's dealings with ISVs. Microsoft had agreed to give preferential access to early versions of Windows 98 and the next version of Windows NT to the ISVs in exchange for an agreement to use IE as the default software for HTML-based materials. The lower court concluded that these agreements foreclosed Netscape from an important channel for distributing Navigator, and the appeals court found this to show a *prima facie* case of monopoly maintenance, and again Microsoft offered a minimal procompetitive justification.

*Deal with Apple.* The appeals court affirmed the lower court's finding the Microsoft had behaved anti-competitively in threatening to cancel the Mac version of Office if Apple did not bundle IE with Mac OS and make it the default browser.

*Deals Relating to Java.* The court of appeals reversed the lower court's ruling that Microsoft acted illegally in creating an incompatible version of the Java virtual machine (JVM) in competition with Sun's version of the JVM. Incompatibility alone did not violate the Sherman Act, and the court understood the competitive benefits of having multiple versions of the JVM. This is true even if Microsoft's strategic reason for incompatibility was to fragment the Java market. But the court of appeals upheld lower court rulings on deception in the marketing of Microsoft's Java authoring software; on *de facto* exclusivity in certain agreements for the distribution of Navigator, which in turn blocked the distribution of Sun's JVM; and on Microsoft's anticompetitive threats to Intel on a possible Intel JVM.

As the court of appeals concluded, Microsoft cut deals with a variety of industry players to give them special access to Windows in exchange for their support of IE and limitations on their use of Navigator. In doing so, Microsoft stepped beyond its assigned role of hub, both by including software in Windows for reasons that had little to do with consumer satisfaction and by using its control over Windows to tilt the competitive playing field against Netscape.

Microsoft repeatedly incorporated new features into Windows that allowed it offer special placement to other companies. Doing so takes the idea of "advertising" discussed above in connection with the Windows Installer and applies it quite literally to the Windows desktop. Two features, the Online Service Providers

folder and the Channel Bar, are of particular note. The Windows desktop advertised, actually quite literally, ISPs, including America Online (AOL). The Channel Bar advertised content providers. It was precisely the ability to advertise on the Windows desktop that created leverage for Microsoft in its effort to attract users to IE away from Netscape Navigator.

In these cases, Microsoft was able to use its control over Windows to create scarcity and then to use that scarcity as a lever against Netscape. Start with scarcity. To understand Microsoft's incentives, it is useful to keep in mind an example known as the card game (BRANDENBURGER AND NALEBUFF [1996, pp. 41ff.]). Contrast two versions. In the first, one person holds five black cards, while five other individuals each hold one red card. Each successful pairing of a black and red card is worth \$100, and that value can be split in whatever fashion agreed to by the cardholders. We cannot be sure what will happen here, but we might guess that we would get five matches, and roughly a 50/50 split of the value. The black card monopolist would get \$250 and each red card holder would get \$50, for a total of \$250 going to the red card holders.

Now suppose that the black card holder destroyed one black card. How would this change the outcome? It is clear that society is worse off. We can now only get four matches for a total of \$400 in value. Nonetheless, our black card monopolist may do better. How? One holder of a red card is going to get left out. Imagine that tentative deals have been struck with four red card holders with a 50/50 division. The red card holder on the sidelines has every incentive to jump in and offer a different split, say 60/40, to at least get something from his red card. Of course that would just exclude a different red card holder, who in turn would offer a better deal to the black card monopolist. When black cards are scarce, the black card monopolist has enormous bargaining power through its ability to play one red card holder off against another. This bargaining power was created by creating artificial scarcity. The very act of making the overall pie smaller gives a bigger piece to the monopolist.

This is artificial scarcity at work. It would have been easy enough to have just put the ten largest ISPs into the online folder with appropriate code to sign up with the ISPs. Hard disk space is not at a premium. Getting the code right would have been some work, though the ISPs would certainly have been delighted to bear the costs. Customers would have had many choices highlighted for them in a straightforward way. But this would not have worked in Microsoft's interests. Microsoft would not have been able to extract a commitment from AOL regarding support for IE. Microsoft took exactly the same line with AT&T when it was seeking inclusion in the online folders.

Look at the case of AOL in more detail. David Colburn, Senior Vice President for Business Affairs of AOL, testified that AOL feared that its business would be substantially undercut by the bundling of Microsoft's new online service, MSN, with Windows 95. Control over Windows gives Microsoft control over the best possible distribution channel for software, plus Microsoft can give that software prominent placement by sticking an icon for it on the Windows desktop.

AOL desperately wanted something to match this distribution and placement. It had built its business through “carpet bombing,” that is, mass distribution of AOL floppies and CDs, where only 1–2% of those receiving the CDs would ever become AOL customers (how many of these have you thrown out?) (BARKSDALE [1998, para. 228]). Microsoft offered an unmatched inducement to adopt Microsoft’s Internet browser and to reject that of Netscape: placement of an AOL icon in an “online services” folder on the Windows desktop and distribution with Windows of the code required to sign up with AOL. As a result, on March 12, 1996, Microsoft and AOL signed an agreement in which AOL agreed to, in COLBURN’s [1998, para. 29] words, “virtual exclusivity in favor of Internet Explorer on AOL.” AOL could ship a browser other than IE only when required to do so by a third party, and after AOL had taken all “reasonable efforts” to induce the third party to use IE, plus there was an absolute cap that non-IE shipments were limited to less than 15% of AOL’s total browser shipments.

Microsoft can and does manipulate this scarcity. The Online Services folder was made less important with the addition of the Internet Connection Wizard and the Active Desktop Channel Bar (COLBURN [1998, para. 40]). This gave Microsoft a chance to play the card game again, and it once again extracted promises from AOL to benefit IE. AOL and Microsoft executed the Active Desktop Marketing, Promotion and Distribution Agreement in September, 1997, which barred AOL from promoting Netscape within the AOL websites and from compensating Netscape for promoting AOL content (para. 42).

This should make clear why the government succeeded in persuading the District Court that Microsoft had violated Section 2 of the Sherman Act by maintaining its monopoly. Microsoft was able to distort its competition with Netscape by its role as hub by adding content to Windows solely for the purpose of buying allegiance. This is not competition on the merits, as it privileges the position of the central coordinator.

Some of these restrictions might have been justified on a pro-competitive basis given the tricky incentive issues associated with distribution agreements among a producer and its various agents – and we should think of the alphabet entities in those terms. These are principal–agent relationships. Microsoft, the principal, engages the OEM or IAP as agent, to distribute its software. Antitrust law recognizes, *Continental T.V., Inc. v. GTE Sylvania, Inc.*, 433 U.S. 36 (1977) (see also <http://www.ripon.edu/faculty/bowenj/antitrust/cotvvsyl.htm>), that there is no simple way to characterize vertical relationships and so a rule of reason analysis applies. In this particular situation, we have two principals – Netscape and Microsoft – negotiating with the same group of agents. Vigorous competition between the two principals could result in a rough split of the agents, with each principal entering into exclusive agency agreements with ten agents. The real issue is the value of exclusive agency relative to the value of shared agency. Microsoft did not really develop this issue, hence the willingness of the D.C. Circuit to accept the lower court’s findings. See also B. KLEIN [2001].

Other restrictions are more difficult to characterize in a pro-competitive fashion. Preferential access to early versions of Windows betas, for example, is probably about maximizing Microsoft's leverage of that information by creating a sense of advantage and exclusivity from the special access. This is the Studio 54 approach to information disclosure: without have-nots standing on the outside clamoring to get in, those on the inside feel they have obtained nothing special and of course would be willing to pay nothing for it. Again, this could be a way of disciplining an agent – if all get the carrot there is no reason to work for it – but Microsoft did not make this case out and nothing, other than its sense of self-interest, prevented Microsoft from distributing these betas to all ISVs.

Microsoft emphasizes that it has dropped many, if not all, of these contractual limitations long before the ruling (GATES [2001]). That said, it would be a mistake to treat these restrictions as having been unimportant or to infer that from Microsoft's willingness to drop them. We must remember that at the point where Microsoft most faced a competitive threat to its OS monopoly – when Netscape Navigator held a dominant market share and regardless of whether this threat was real or just Microsoft-imagined – Microsoft used its monopoly power over Windows to tilt the competitive process against Netscape. Having succeeded in preserving that monopoly by building up Explorer, the contract limits became much less important.

#### *4.2 Design Monopoly Maintenance*

The claim of monopoly maintenance through the integration of IE and Windows is quite different. This goes directly to the question of how the software is designed. The claim itself is conceptually straightforward: If Microsoft designs software to harm competition and preserve its Windows monopoly and not to make better software for consumers, it has engaged in monopoly maintenance. The difficulty, of course, is in separating beneficial and anticompetitive designs, and the difficulty of doing so has made courts, in the word of the D.C. Circuit, “properly very skeptical” about sorting through product design changes. The lower court focused on three particular acts of alleged design monopoly maintenance: (1) the exclusion of IE from the Add/Remove programs utility; (2) the commingling of code for the browser and other parts of the OS; and (3) the use of IE as the browser in certain situations, even when another browser had been specified as the default. The court of appeals upheld (1) and (2) and reversed on (3).

Consider each of these in turn. The inclusion of IE in the Add/Remove utility facilitates consumer choice. Large-scale corporate users who want to deny their employees the ability to surf the Internet can do so easily by eliminating all browsers. Other large users who prefer Navigator and want to eliminate the costs of supporting dual browsers can swap in Navigator for IE. The inability to remove IE created the same cost-entry barrier for Navigator as was created by the contractual icon restrictions, as part of a raising rivals' cost strategy. In both cases, the presence of IE meant that OEMs would have to support IE, and this made it less likely that

they would also install and support Navigator. We can with some ease articulate the consumer benefits that would have obtained from inclusion of IE in Add/Remove as well as the competitive harms that resulted from the failure to do so.

But the nature of the after-the-fact looks at product design means we have, with the benefit of hindsight, zoomed in on the single design decision we think mattered. Before the fact, what should have limited the scope of the Add/Remove obligation? This is a question of how much the software should be able to be customized by end-users. In buzzwords, this is about the extent of mass customization. There is no simple answer to this question, but we can say that we want software houses to make design decisions on the merits, not on the competitive consequences. The evidence suggests that it was precisely the fact that Microsoft knew which features mattered for competition – the browser and not printer drivers – that led it to eliminate Add/Remove for IE.

As to the consumer confusion issue, the actual presence of IE – meaning its pre-installation on the computer even if in a form invisible to the end-user – is probably unimportant – although the district court did suggest that hard disk space was wasted. Visibility is what matters as this is what triggers the support cost concern, which in turn made it more difficult to get Navigator installed. This suggests that the fix here is less one about design itself and much more about which features are advertised or made visible. Eliminate advertising of IE – no icon on the desktop or in the Quick Launch Toolbar (the little bar at the bottom with the IE icon) – and we take a big step towards solving the support problem (as noted in MICROSOFT [2001b]). This is precisely what Microsoft accomplished in its July 11, 2001 announcement (MICROSOFT [2001a]) that OEMs would now have the option to remove icons for IE from the desktop and that IE would be restored to the Add/Remove function.

What problems would that leave? Consider the second design issue, the commingling of code for the browser and the OS. The appeals court found that the lower court evidence was mixed on whether Microsoft had commingled to impair competition, a result Microsoft has challenged in its petition for rehearing. Nonetheless, this misses the bigger picture. Consumers have no direct stake in how these files are organized. Yes, lumping them together may mean we cannot delete parts of the file, but this is barely even small potatoes given the growth of disk space. So the direct benefits of deletion are small, and controlling visibility is a better instrument for solving these issues than trying to control directly the assignment of code to particular files.

Do note in this regard that the claim in the case is not that the presence of the IE code directly impaired the operation of Navigator. There was some suggestion of this. For example, Brad Chase of Microsoft stated that “[w]e will bind the shell to the Internet Explorer, so that running any other browser is a jolting experience” (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999, para. 160]). The claim is one of strategic impairment, not technical impairment. The visibility remedy minimizes the strategic impact of the design decision but would do nothing if there were a direct technical problem.

There is a second issue here though, one addressed somewhat obliquely by the court of appeals but more squarely in the lower court. The visibility solution focuses on the appeals court's view of the strategic harm of integrating the browser, namely that it directly raised the cost of installing a second browser. Making it possible for IE to be invisible, even if present, would solve the support cost problem. It would leave in place though a second strategic problem. We need to distinguish the strategic consequences of visibility and presence. So far we have focused on visibility. The mere presence of IE, for "free" as it were – more precisely, included with the OS at no separate marginal price – means that Microsoft can rely on it for various features, such as the Windows Help system and Windows Update, and outside developers can rely on its existence as they create their products. That is useful – this is precisely Microsoft's point about providing a common ubiquitous infrastructure – but this also may mean that Netscape would be deprived of a chunk of users it would otherwise get. Absent a large enough user base, the economies of scale of software may foreclose Netscape from successfully developing its products (see WHINSTON [1990], [2001]; and RASMUSEN, RAMSEYER, AND WILEY JR. [1991]), plus developers may continue to focus on developing for Windows directly if the browser market is fragmented.<sup>2</sup>

In the extreme case, the OS monopolist might try to foreclose a related product completely. Try a simple situation to see this (see CARLTON AND WALDMAN [1998]; see also CARLTON [2001]). Users have no interest in the OS *per se*. Users just want to surf the web. One-third of the users would prefer to do so using IE, one-third using Netscape, and one-third are indifferent between the two. Each browser runs on top of the OS, so the OS is required to run the browser. There is a fixed cost of \$100 to build the OS and of \$25 to build each browser. We have 150 consumers, 50 who value browsing with Netscape at \$2, and with IE at \$0, 50 with contrary preferences; and 50 who value browsing at \$2 with either browser. (No value is attached to just having the OS.)

What would an all-powerful social planner do? It obviously makes no sense to build just the OS, as that would cost \$100 and create no value. She could build the OS and one browser at a cost of \$125, and this would result in value of \$200, for a net gain of \$75. She also could build the OS and both browsers at a cost of \$150 for a value of \$300, and a net gain of \$150. She should obviously do the latter.

How should we price this? What price should we charge for the OS and for each of the browsers? In this example, with three types of identical consumers, it really does not matter. Suppose that the social planner just wanted to recover fixed costs and leave the consumer surplus to the consumers. She could price the OS at \$1 and give away both browsers. She could give away the OS and charge \$1 for each browser. She could do a hundred other variations.

---

<sup>2</sup> See U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 39]. (“[W]e conclude that such commingling has an anticompetitive effect; as noted above, the commingling deters OEMs from pre-installing rival browsers, thereby reducing the rivals’ usage share and, hence, developers’ interest in rivals’ APIs as an alternative to the API set exposed by Microsoft’s operating system.”)

It is clear what an OS monopolist would do. He would set the price of the OS at \$1.66. This would permit the monopolist to fully extract all of the consumer surplus while just leaving enough value for the browser producers to recover their fixed costs. They, in turn, would charge \$0.33 for the browser. The OS monopolist would get revenues of \$250, each browser company of \$25. Obviously, if the OS company is the first browser company, its total revenues would be \$275 against costs of \$125 for profits of \$150.<sup>3</sup>

Now change this slightly. Imagine this scenario as being played out period after period, where we have new consumers each period, but the same products, and, to make matters easy, make it certain that in the second period the browser would emerge as a competitor to the OS. Assume if that happens that competition erupts such that all of the consumer surplus goes to consumers, and there are no profits going forward for the software producers.

How would this change the outcome for our social planner? Not at all: still build the OS and both browsers and set charges to cover costs. But our monopolist would face a different choice. The monopolist could proceed as before and would get profits of \$150 in the first round and no profits thereafter. Alternatively, the monopolist could seek to prevent entry by the second browser. The monopolist could sell the OS for \$2 and give away the browser or could bundle the two together and sell at \$2. These are equivalent here.

At that price the monopolist sells 100 copies of the OS and the browser: 50 of each to the devotees of its browser, 50 to the different. Total revenues are \$200 against costs of \$125, for profits, in the first round, of \$75. In the second round, the monopolist either gets profits of \$75 again, if we think of the fixed costs as being incurred in each period, or profits of \$200, if not. The monopolist will seek to deter entry if \$150 now is less than the present value of the stream of profits without the entry of the second browser company. In the first version of this hypothetical, the OS monopolist wants both browsers produced, as this increases sales of the OS. In the second version, assuming modest rates of interest for present value calculations, the monopolist will deter entry of the second browser to prevent losing the original OS monopoly.

That leaves the third design issue. The appeals court overturned the finding below that Microsoft engaged in monopoly maintenance in overriding, in some circumstances, the consumer's default choice of browser and instead invoking IE. Microsoft offered a defense of this practice, and the appeals court saw the plaintiffs as offering no response, hence the Microsoft victory. In some ways, the court was too deferential to Microsoft's analysis. Microsoft argued that IE had to be invoked when consumers used the Windows 98 Help system or Windows Update, as both relied on ActiveX controls not supported by Navigator and on Microsoft's Channel Definition Format, also not supported by Navigator. Even if all true, we

---

<sup>3</sup> This could be seen as a little casual. Once the fixed costs are spent, software in hand, we could easily expect Bertrand price competition to push browser software prices to zero. The analysis in the text assumes that the entry and pricing decisions are just one decision.

have design decisions layered on design decisions. If we knew that Microsoft uses ActiveX controls in the help system precisely because they were not supported by Navigator, this would make the need for IE pretextual. This is difficult to evaluate without more information.

We need a bottom line on design monopoly maintenance. It is easy to state the D.C. Circuit's conclusion (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 40]): "[a]ccordingly, we hold that Microsoft's exclusion of IE from the Add/Remove Programs utility and its commingling of browser and operating system code constitute exclusionary conduct in violation of § 2." It is less clear what this means. Does this mean that if Microsoft distributes a feature in or with Windows for which there is a competitor – instant messaging (IM), for example – Microsoft engages in illegal monopoly maintenance by making it less likely that the competitor's software will be installed?

Most narrowly, on the court's analysis, we should limit this rule to features that are possible (likely?) competitors to the OS and to visibility and not mere presence. This tracks the court's analysis in that the browser was seen as a potential competitor to the OS by Microsoft, and hence its anti-competitive design decisions were clearly about maintaining the OS monopoly. The focus on visibility and not just presence matches the view that the strategic harm here was the way in which support costs were raised if consumers face the choice of using two different browsers. Even with the narrow rule, we have the difficult problem of identifying potential competitors to the OS. Instant messaging? The RealNetworks media player?

#### 4.3 Causation and Monopoly Maintenance

That said, there is little reason to think that this monopoly maintenance actually mattered, at least on the government's theory of the case. The court of appeals acknowledged that the causation issues here were tricky, but saw the issue as going to remedy, not liability (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 59–62]). The government saw Microsoft as seeking to protect the "applications barrier to entry," meaning the software ecosystem organized around Windows. Had Netscape succeeded in establishing a parallel presence, a new ecosystem might have emerged around it. This misses the mark in two ways. First, Microsoft almost certainly would have succeeded in preserving its ecosystem simply by adding IE to Windows. Fragmentation of the browser market, not monopolization of it, was all that was required.<sup>4</sup> Second, in some more fundamental sense, Microsoft was losing by winning. A new ecosystem did emerge, the World Wide Web, and that has supplanted the primacy of Windows, both as a focal

---

<sup>4</sup> That said, Microsoft itself may not have believed this to be true. As the lower court found (see U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999, para. 160]), Microsoft moved to integrating IE and Windows precisely because it feared that the contractual restrictions might not be enough. We have less direct evidence on whether Microsoft would have thought that integrating IE with Windows would have sufficed.

point of attention and as a means of distributing software. Placement on the Windows desktop may still be the premier location in computer real estate, but it is now just one of many valuable locations in a consumer's virtual space.

And the paramount role of the Windows CD and desktop as the device for distributing software has substantially declined in the networked world. As even Judge Jackson realized in rejecting the claim that Microsoft had engaged in illegal exclusive dealing, Netscape was able to distribute tens of millions of copies of Navigator, many over the Internet. Indeed, ironically, the distribution of IE itself made it easier for consumers to get Navigator over the net.

## 5 Tying

The D.C. Circuit reversed the lower court on the claim that Microsoft tied IE to Windows in violation of Section 1 of the Sherman Act. The lower court had found a *per se* violation, but the D.C. Circuit remanded for a rule of reason inquiry into the alleged tie. In doing so, the court recognized the "poor fit" of applying traditional tying doctrine to software markets. Standard antitrust tying analysis requires that we have two goods (*Jefferson Parish Hosp. Dist. No. 2 v. Hyde*, 466 U.S. 2 (1984); *Times-Picayune Publishing Co. v. United States*, 345 U.S. 594 (1953)). Without two goods, by definition, there can be no tying of one good to a second good. Hence, the focus on integration and separation of software in the District Court and in the D.C. Circuit's *Microsoft II* (*United States v. Microsoft Corp.*, 147 F. 3d 935 (D.C. Cir. 1998) – see also [http://eon.law.harvard.edu/msdoj/msft\\_ruling.html](http://eon.law.harvard.edu/msdoj/msft_ruling.html)).

This discussion has little meaning in this context, given the malleability of software design (see also LESSIG [2000]). Good software design strives for modularity. Breaking up code into discrete chunks facilitates team creation of software and maximizes re-use of software. By setting out a piece of code, other software developers need not recreate functions in application after application but instead can call and use distinct modules. In that framework, an "integrated" piece of software is poorly-designed software: it looks to no other software for functions and sets forth no functions that other software can access. It does not share. In contrast, fully-unintegrated software shares well: it takes full advantage of preexisting software modules and creates modules available to other software. It would be perverse to tilt our antitrust tying analysis in a fashion that pushed towards poorly-designed software.

### 5.1 Many Free Goods and Traditional Tying Doctrine

How should we evaluate tying in this context? Consider the traditional formulation of the harms of tying as set forth by the D.C. Circuit (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 73]):

"Direct competition on the merits of the tied product is foreclosed when the tying product either is sold only in a bundle with the tied product or, though offered separately, is sold at a

bundled price, so that the buyer pays the same price whether he takes the tied product or not. In both cases, a consumer buying the tying product becomes entitled to the tied product; he will therefore likely be unwilling to buy a competitor's version of the tied product even if, making his own price/quality assessment, that is what he would prefer."

We immediately encounter a problem when we apply this to software markets: we see lots of free stuff and tying theory does not handle that very well. So, for example, is Adobe's free Acrobat Reader tied to Windows? You could answer this in a number of ways. Yes, it is a free, third-party complement to Windows, but it is not specific to Windows as Adobe makes free versions for a variety of platforms, including Macintosh, Palm, and Linux. Suppose Adobe made only a Windows version, would that change the analysis? We usually think of the tie as being made by one company, and not arising through the choice of a third party, so this does not square with the traditional doctrine at all.

Now the question: if free software – or more precisely, a two-piece pricing strategy, with a single company selling a mix of free and for-a-fee software – is a common software strategy, even in markets we believe to be competitive, does this preclude applying tying law to software markets? Or should we say that a company with monopoly power in one market is barred from giving away free software in a second market, even though other companies in that market will give away the software as well?

As we know from the case itself, this is not a fanciful scenario. Netscape gave away versions of Navigator to some individuals, though it tried and succeeded in making sales to corporations. Netscape was looking to making the browsing market drive sales in the server market, just as Adobe gives away its reader to sell the authoring software. In contrast, Microsoft was focused on the relationship between the browser market and the OS market. As Judge Jackson found, Microsoft's entry into the browser market created competition and thereby pushed down prices to the benefit of consumers. Moreover, the central concern of tying – that the consumer does not use the preferred second good – vanishes if the second good is free as well. This is free versus free competition, and tying doctrine does not get that.

A rule that says that a company with monopoly power in one software market cannot give away free software in a second market will help to push up prices making consumers worse off. This could be a bizarre double-whammy. The browser market competition linked competition in two markets that might otherwise have been separated, OSs and server software. If Netscape had obtained monopoly power in the server software market, the anti-free software tying rule would have barred both Microsoft and Netscape from distributing free browsing software.

## 5.2 *Tying as Mandatory Software Coordination*

We should focus instead on software coordination. Impermissible tying occurs when an entity with market power attempts to take over coordination of software

sharing on a mandatory basis in a context where it should not do so and need not do so. That was a little vague, so consider again coordination in the pre-network and post-network worlds. In the pre-network world, most if not all shared software needed to be provided by the centralized hub. The inability to distribute across a network as needed and to pay for that software at the same time created a real pressure to centralize in the OS and distribute that software upfront. In contrast, in the networked world, applications can rely on whatever components they choose to, and the WI reaches out to get them as needed. The transaction can take place in real time. Hence, almost no bundling is required. Software can be purchased on an as-needed basis. Impermissible tying then emerges as an attempt to impose mandatory centralized coordination where it has ceased to be required.

What does this say about Microsoft's decision to include IE in Windows? We have reached the end of the line, the transition point between the *pre* and the *post*. Incorporating IE was fully consistent with Microsoft's past role as central coordinator of shared software, but at the same time, marks the end of the need for Microsoft to play a unique role in coordinating software. The D.C. Circuit noted that bundling has virtues in reducing distribution and consumer transaction costs and in facilitating shared libraries (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 73]). Precisely so, but the rise of the network changes the mechanism for doing this. Pre-network, Microsoft did this through Windows; now the always-on network makes it possible to decentralize this coordination.

The court of appeals paid too little attention to this idea in rightly abandoning the *per se* condemnation of tying by the lower court. The D.C. Circuit (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 76]) pushes for a rule for reason analysis that respects "potentially innovative technological integration." The court focused on precisely the right issues regarding software sharing and the issues of coordination (p. 83):

"For example, the bundling of a browser with OSs enables an independent software developer to count on the presence of the browser's APIs, if any, on its own package and thus to omit them from its own package ... It is true that software developers can bundle the browser APIs they need with their own products ... but that may force consumers to pay twice for the same API if it is bundled with two different software programs. It is also true that OEMs can include APIs with the computers they sell ... but diffusion of uniform APIs by that route may be inferior. First, many OEMs serve special subsets of Windows consumers, such as home or corporate or academic users. If just one of these OEMs decides not to bundle an API because it does not benefit enough of its clients, ISVs that use that API might have to bundle it with every copy of their program. Second, there may be a substantial lag before all OEMs bundle the same set of APIs."

Well and good, but the more important issue is whether we could achieve the same technical benefits of integration or bundling seen by the court – and for the purpose of coordination these are fungible – with across-the-network distribution while doing a better job of preserving competition.

### 5.3 *Visibility, Presence, and Price*

How might we do that? Tying and/or bundling are notions too crude to get at the issues in these markets. Instead, we should focus on three more textured characteristics: (a) *visibility (or advertising)*: What software does the OS make visible to the end user? (b) *presence*: What software is available for use, either through pre-installation or by invocation across the network? and (c) *price*: Are separate charges set out for software or is one package offered at a lump-sum price?

In each case, we need to focus on the costs and benefits of centralized provision and those of decentralized provision. Taken together, these three characteristics create a rich framework for evaluating situations we would otherwise lump under the tying/bundling rubric.

We can now be more precise about our comparison of centralized and decentralized software. What should limit the scope of centralized provision of visibility, presence, and price? The evidence in the case suggests that in the pre-network world, pre-installation was seen as the best method of distributing software. Pre-installation meant both visibility within Windows – either an icon on the Desktop, a spot in the Channel Bar or a line in the Online Services folder – and presence, as the relevant software was automatically installed on the computer prior to delivery.

As to presence, in the pre-network regime, we cannot install on demand over the network, meaning that we cannot distribute and pay at the time that the end-user is ready to use the software. Instead, the software must be made available ahead of time, either through pre-installation on the hard disk or through availability on a CD. When I needed to install the requested Hebrew font, the system asked me to insert my Windows Update CD, which was where exactly? There are clear costs to IOD depending on how cumbersome/invisible the installation process is. Installation off of a CD is cumbersome, especially if this must occur repeatedly. The decision to pre-load is essentially a forecast about the expected future use of a product, the timing of that use and the cost of IOD. We need to trade off the costs of pre-installation – even if in a hidden form only activated on request (invisible but present) – against the costs of true on-demand installation. The physical cost of pre-installation is largely just hard disk space, which has, with the remarkable progress made in storage technology, become less than dirt cheap.

So, in the pre-network era, maybe we pre-install a bunch of software, maybe we do not, but focus on the question of payment. We need to separate pre-installation from the question of whether the pre-installed software is paid for as a group. Pre-installation is essentially about the cost of distribution versus the cost of storage. Pricing is a separate question. Again, in the pre-network era, it is much more difficult to pay in real time when the software is used (either initiating a purchase transaction or a rental transaction). So we either bundle large amounts of software based on some forecast about consumer demand, or we rely on the consequences of large amounts of bundling for collapsing consumer demand valuations (BAKOS AND BRYNJOLFSSON [1999]).

Networked coordination eliminates many of these problems. In this framework, the OS emerges as a key place in which software is advertised and purchase transactions are initiated. The real question is how hard it is to organize a market in online software. Given the network, distribution is easier, if not perfect. What may be harder is the organization of payment, though if you have purchased any intangible good on the Internet – audio files, for example – you understand that this is reasonably straightforward. And some of the difficulties may be solved by the market participants themselves. If software package *A* anticipates a file from software package *B* and that file is missing, networked coordination will allow instant downloading, and payment may be made not by the consumer but rather by company *A* to company *B*. Doing that would insulate the consumer from many of these online micro-transactions.

As to price, consider three cases: “integrated” software sold at a price  $p$ ; bundled software – bolted software – sold for  $p$ ; and two “separate” pieces of software, the first sold for  $p$ , with the second given away. In each case, the consumer has access to all of the software and has parted with  $p$ . Absent a meaningful price for the “second” piece of software – ignoring transaction costs – consumers will treat these three situations identically. Put differently, consumers would think of the OS as being bundled with free complements and would not distinguish integration as a single product from bundling with “separate” free complements.

The antitrust tying analysis turns on distinguishing case 1 from cases 2 and 3, though as suggested above, good software design principles suggest that we should not try to distinguish these cases. How do transaction costs matter for these situations? Cases 1 and 2 are identical: the software is pre-installed and is made visible on the desktop. Pre-network, case 3 is quite different, as it was quite costly to distribute free software, as those of us who use AOL sign-up software CDs for coasters are well aware. The rise of the network brings case 3 substantially closer to cases 1 and 2, though gaps remain.

#### *5.4 Three Examples of Free Complement Distribution*

Focus on three real situations and consider the differences in actual distribution. In each case, we are looking at a free Windows complement, and the only questions are visibility and presence. In Windows XP, Microsoft’s next OS, Microsoft will have IM (instant messaging) come pre-installed and visible, with no marginal price assigned to the IM component. Obviously, this is software that is visible, present, and free. In contrast, Microsoft has announced that it will not distribute a JVM (Java virtual machine) with Windows XP, but instead will make one available for downloading from its website (WILKE AND CLARK [2001]). Microsoft describes the Java code as “a lot of code that many users don’t need” but that of course is true of most possible features in the OS and does not provide a rationale for distinguishing the decision on Windows IM software and Java. The IM software could just be visible through a link for downloading the software for free from Microsoft’s website. The only meaningful difference between these two situ-

ations is that in the one case, the software is pre-installed – meaning installed by the OEM for a consumer purchaser – and in the second the software is installed over the network.

Compare the ease of use of Windows IM with that of another free Windows complement – one partially decentrally coordinated – the Adobe Acrobat Reader. Acrobat is a two-piece authoring and reading system. A person using the authoring program “prints” a document to a file, creating a portable document that can be moved from computer to computer. This document will preserve the look and feel of the original document: it will appear on the second computer as it did on the first computer, so long as the recipient has the Acrobat Reader program to view and print the file. The reading program is free and can be downloaded from Adobe’s website; you have to buy the authoring program.

Acrobat files end in the three letter extension .pdf (thus the reason they are frequently referred to as “pdf” files). If you obtain an Acrobat file on your Windows 2000 machine and attempt to read it, if you have not already downloaded the Reader, you will not be able to open the file. Instead, a box will open with the heading “Open With ...” and you will be asked to select the program that you would like to use to open the file. This box will come up, because Windows matches three letter suffixes to programs, and will have looked on your computer in the table that matches suffixes to programs. Since you do not have any program matched with that suffix, it will ask you to make the match by hand.

Unlike pre-installation in XP of Windows IM or pre-linking and therefore visibility of Java, Windows will tell you nothing about Acrobat. Windows certainly could come with a table matching all three letter suffixes to programs and websites. Suffix tables are readily available on the Internet, so this is not a hard problem (see, for example, <http://extsearch.com/>, <http://filext.com/>; or [http://webopedia.internet.com/quick\\_ref/fileextensions.html](http://webopedia.internet.com/quick_ref/fileextensions.html)). It turns out that there are multiple programs associated with the .pdf extension. For Adobe, this is its portable document format and matches up with Acrobat. For Microsoft, this is package definition file, and is used by its systems management server. Windows could offer links to both websites, and you would quickly download the free Adobe Reader.

Does this mean that our user will throw up her hands, curse, and stop? Probably not. For the Acrobat Reader, we see decentralized, viral coordination: sites posting .pdf files frequently will also link to Adobe’s site making clear how a user gets the free Reader. A person distributing an Acrobat file can easily solve the linking problem that Windows XP will solve for Java but will not solve for Acrobat.

### *5.5 Pre-Installation Again*

Should we think much turns on whether the free complement is distributed centrally? Should an antitrust result turn on the mechanics of installing the instant messenger software? The case for doing so would run something like this. We can see the advantage of bundling in these examples. Pre-installation lowers the transaction costs of creating presence, though the network has shrunk this advantage.

Pre-installation usually comes with visibility on the Windows desktop, but this is hardly the only method of advertising. Pre-installation also lowers the transaction costs of installation. We should expect that if we have two different pieces of software, with equal visibility on the desktop, we will see much greater use of the software with a presence advantage. So even if we had icons on the desktop for both IE and Navigator – or, in the new fight, of Windows IM and, say, AOL's version of IM – if IE was pre-installed and Navigator could be installed by downloading over the network, IE would have a substantial advantage. Downloading can take forever, and a substantial percentage of download attempts fail.

On the liability claims, this leaves outstanding the claim of attempted monopolization of the browser market, the third of the violations found by the District Court. One piece of this, the alleged June, 1995 offer of market division to Netscape, may have great legal significance, but it has zero scholarly significance. Of more interest is the District Court's conclusion that Microsoft's success in the browser market adds credence to the attempted monopolization claim. The lines here between success in legitimate competition and attempted monopolization seem especially thin, with severe consequences for getting the answer wrong. We will find a violation when we should not, or discourage beneficial competition, much to consumer's loss. All of this is especially troublesome in markets that seem to tend to monopoly, so that the new competitor can tip into monopolization quite quickly. The court of appeals ultimately rejected the attempted monopolization claim based on the plaintiffs' failure to define meaningfully the browser market or to make out significant entry barriers in that market (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 64]).

## 6 Remedies

The court of appeals well-recognized the difficulty in identifying an appropriate remedy in this case (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 10f.]):

“[S]ix years seems like an eternity in the computer industry. By the time a court can assess liability, firms, products and the marketplace are likely to have changed dramatically. This, in turn, threatens enormous practical difficulties for courts considering the appropriate measure of relief in equitable enforcement actions, both in crafting injunctive remedies in the first instance and reviewing those remedies in the second. Conduct remedies may be unavailing in such cases, because innovation to a large degree has already rendered the anti-competitive conduct obsolete (although by no means harmless). And broader structural remedies present their own set of problems, including how a court goes about *restoring* competition to a dramatically changed, and constantly changing, marketplace.”

As the court itself noted (p. 11), we could obviously just accept the state of the market and move on, instead emphasizing deterring antitrust violations and could do that through a very large penalty or through private damage actions.

We should start with the remedy implemented by the lower court. The Final Judgment in the District Court contemplated a two-stage remedy. The first stage would consist of so-called conduct remedies that originally were to go in effect in three months. (The District Court ultimately stayed this remedy.) The second stage called for the break up of Microsoft into two entities, an operating system company and an everything-else applications company. That remedy was to go in effect only after all appeals had been resolved.

### 6.1 *The Break Up*

Of course, the appeals court ruling requires a fresh look at the remedies. The break-up ruling was thrice damned: first, by the partial reversal of the liability ruling; second, by the finding that the lower court erred in failing to conduct an evidentiary hearing on the remedy; and third, by the conclusion that Judge Jackson's extra-judicial statements created an appearance of bias that infected the remedies phase.

Although the appeals court leaves the possibility of a break up on the table, it notes that historically divestiture has been used for companies that were merged together, so a break up simply takes the pieces apart and restores the old situation. Microsoft, of course, was built from the ground up as a single company. While it frequently buys software and makes it its own – as it did with DOS itself – mergers have played no role in Microsoft's amazing growth or in the dominance of Windows. The D.C. Circuit also hinted that there was only a weak causal connection between Microsoft's exclusionary conduct and its market dominance, suggesting that the court thought a break up unwarranted.

Indeed, the government's vision behind the break up has never been very clear. It seems fair to say that the government felt that it got burned in its prior settlement with Microsoft and was loathe to repeat that experience. The conduct remedies might give Microsoft wiggle room that would render those remedies empty, as had occurred before. The break up would also limit the possibility of ongoing monitoring and supervision by the government of Microsoft. That is to be applauded, but taking Microsoft into the back yard and shooting it would do that as well, yet that hardly suggests it is a sensible remedy. On a more theoretical level, it is possible to articulate only a weak basis for the break up. One concern is that Microsoft has given its in-house applications developers special information about the Windows API (applications programming interface), and that cleaving Microsoft will eliminate that problem. That is true, but the government made nothing of this in the case, so it is unlikely it should play a role in the design of the remedy.

A second possibility is that separation will lead to OS competition. The story would go something like this. The single most important application is Microsoft Office. Microsoft must consider how decisions for Office alter the position of Windows. Macintosh conspiracy theorists have long believed that Microsoft has intentionally delayed Office for the Mac as a way to weaken the Mac as a competing OS. Indeed, as noted above, the government alleged that Microsoft threatened

to kill off Mac Office if Apple did not support IE. Separate Office from Windows and the applications company will make clean decisions on whether to extend Office to other platforms and will not favor Windows.

Or so the story goes. The reality, of course, is that with Windows as the dominant OS, AppCo would almost certainly focus its resources on Windows. If we assume that AppCo cannot just scale up immediately so that it does face internal constraints, we would expect Windows to continue to get priority. Any new OS would face the standard chicken-and-egg problem: developers do not want to develop for small OSs, and consumers will not adopt the OS if there are no applications. If this is the point of the separation, rather than administrative convenience, it is hard to see how this will work.

## 6.2 *The Conduct Remedies and Coordination*

What this will also not do is address the key issue of facilitating the transition from centralized coordination of software sharing to networked coordination. Simply popping Windows into OSCo will not change the ability of Microsoft to use Windows to coordinate software sharing. That is a good thing if we believe that Microsoft needs to continue to play that role, but unfortunate if we can now move to networked coordination. Absent controls on Microsoft's decisions to incorporate features into the OS, it can continue to claim that it should be the only seller of shared software and can enforce that claim by bundling the software together and selling it for a single price.

That is what must be addressed, and the original conduct remedies actually do a much better job of this. Section 3 of the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b]) set out nine conduct remedies, some with subparts. Microsoft's proposed final judgment set forth five conduct remedies. There was some overlap in concept between the two, even if there was enormous difference in the nitty-gritty. As is common, these remedies focus on nondiscrimination, both as among various groups of outsiders (Dell v. Compaq) as well as as between outsiders and Microsoft. The remedies also attempt to address overall software competition, including dealing with Microsoft's ability to bundle together software, plus possible disclosure obligations for Microsoft.

### 6.2.1 Structuring Software Market Competition

Microsoft distorted the distribution of software through its monopoly maintenance; the right remedy for that is to distort back against Microsoft for a period, and then make sure that Microsoft is not able to distort distribution again. The presence of the always-on network means that we can do this while minimizing possible harms to consumers from introducing a corrective distortion. A remedy should consist of five central features:

(1) *DI Visibility Flexibility*. There should be no mandatory icons on the Windows desktop or spots reserved in the Start Menu or its equivalent. DIs (distribution

intermediaries) would have complete freedom to add or subtract icons from the interface.

(2) *Mandatory Versioning*. During the remedial period – see (5) below – Microsoft should be required to issue Windows versions with and without any new middleware that it adds to Windows. For this to be meaningful, this means that the baseline Windows XP could not include IM and the Windows media player, but that those features could be included in an upgraded version of Windows XP. Microsoft could charge the same price for basic and deluxe versions of Windows.

(3) *Direct Distribution Only Period*. For a moratorium period, perhaps of six months to 2 years, Microsoft should be able to distribute through distributional intermediaries only the baseline Windows without the new middleware. Microsoft would be able to distribute such middleware only through downloading from its website or through direct distribution of CDs to end-users.

(4) *DI Neutrality*. After the moratorium period, we should rely on competition among software producers and others for DI shelf space – hard disk space for OEMs, web presence for ISPs and others – to control software distribution abuses, and should only seek to control possible abuses of market power by limiting conditions that Microsoft can impose on DIs. Mandatory versioning would be continued during this second period.

(5) *Sunset*. All of these provisions should sunset, perhaps after a period of three years.

The analysis of bundling above suggests three characteristics of interest: visibility, presence, and price. These remedies address each of these directly. As discussed below, in the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b]), Sections 3(a)(iii) and 3(g) come closest to addressing these issues. Section 3(a)(iii) addresses OEM flexibility in product configuration, while Section 3(g) controls attempts by Microsoft to “bind” software to the OS. Before looking at these in detail, consider what we should want to accomplish.

(A) *Visibility Possibilities*. The D.C. Circuit concluded that Microsoft used its control over visibility within Windows to disadvantage Netscape Navigator so as to maintain its monopoly over OSs for Intel-based computers. An appropriate remedy should control visibility. We could imagine a range of a possible approaches here:

(a) *The Neutrality Trustee*. We could appoint a trustee with the power to supervise – regulate – the process by which Microsoft puts icons on its desktop or application names in folders visible from the Start Menu. Trustees of this sort are increasingly common to implement antitrust remedies or settlements; for example, there will be a monitor trustee in the AOL–Time Warner (TW) merger to manage the process of ISP access to AOL/TW’s cable network (U.S. FEDERAL TRADE COMMISSION [2000]).

(b) *Lotteries*. We could try to remove as much discretion as possible from this process of creating visibility within Windows. Again, focus on the inclusion of ISPs in the Online Services folder. We could arbitrarily choose to have ten listed

and simply hold a lottery for the slots, while allowing secondary sales of the slots.

(c) *Auctions*. Microsoft would simply auction off the ten spots. This is very much like the lottery proposal, except that Microsoft gets the money that might otherwise flow to lottery winners.

(d) *DI Visibility Flexibility*. We could simply free the distributional intermediaries of restrictions by Microsoft, allow them to make choices, and assume/hope that competition among the DIs will lead to smart choices. A rule of no mandatory icons or folders – no mandatory visibility for Microsoft software or services on the Windows desktop – would accomplish this result. Could an OEM provide a blank desktop? Sure, but it would probably not stay in business very long.

I think one of these – freeing the DIs – clearly dominates the others, and Section 3(a)(iii) of the Final Judgment basically embraces this idea (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b, p. 6f.]):

“OEM Flexibility in Product Configuration. Microsoft shall not restrict (by contract or otherwise, including but not limited to granting or withholding consideration) an OEM from modifying the boot sequence, startup folder, internet connection wizard, desktop, preferences, favorites, start page, first screen, or other aspect of a Windows OS Product to

- (1) include a registration sequence to obtain subscription or other information from the user;
- (2) display icons of or otherwise feature other products or services, regardless of the size or shape of such icons or features, or to remove the icons, folders, start menu entries, or favorites of Microsoft products or services;
- (3) display any user interfaces, provided that an icon is also displayed that allows the user to access the Windows user interface; or
- (4) launch automatically any non-Microsoft Middleware, OS or application, offer its own IAP or other start-up sequence, or offer an option to make non-Microsoft Middleware the Default Middleware and to remove the means of End-User Access for Microsoft’s Middleware Product.”

Interjecting a trustee would make the interface design decisions intensely regulatory and would almost certainly impose law speed as a powerful brake on making tech progress. Lotteries or mandatory auctions<sup>5</sup> would be less cumbersome, but each would depend on the key step of defining the good in question, and that would probably involve a regulatory step. In contrast, DI freedom avoids all of this, and so long as that market is competitive, should achieve a good result.

I should note that the more difficult we make it for Microsoft to get out its version of the Windows interface, the more likely it will look for other means of distributing Windows. Obviously, it does this to some extent through direct sales to consumers, but Microsoft could decide to vertically integrate and enter the PC

---

<sup>5</sup> Although Microsoft is frequently criticized for the deals that it cuts with third parties to distribute their software through Windows, deals for cash, rather than for anti-competitive benefits, should not be problematic, and would be akin to Microsoft voluntarily embracing the auction idea.

business directly. It also could redefine the OS and offer all DIs the same very limited OS, requiring almost automatic upgrades directly from Microsoft before an end-user could use a new PC.

*(B) Presence Possibilities.* The DI flexibility provision addresses visibility. Consider presence and focus on three possible approaches to achieving neutral presence:

*(a) Opening Windows to Others and the Neutrality Trustee.* Our neutrality trustee also could have the power to require Microsoft to distribute third party software with Windows. This would be a form of open access regime of the sort debated in the context of ISP access to cable lines.

*(b) Minimal Bundling and Installation over the Network.* We could look for a common denominator for distribution, and the always-on network is a natural choice. If Microsoft wants to add IM to Windows, make it post the code on its website, and let users go to that site to download it. This would create greater parity between the Windows IM and all of the other versions of IM, including those of AOL and Yahoo.

*(c) DIs.* Again, by freeing DIs of Microsoft restrictions, we could hope that we would free competition. Microsoft could offer Windows to DIs, the DIs could reject it. Microsoft could offer free Windows add-ons to DIs, and they could reject those as well. Other software producers would compete with Microsoft by offering their software to OEMs for pre-installation.

Consider each of these and start by returning to the bundling discussion. I suggested there that, even in the networked world, pre-installation of software might confer a substantial advantage due to reduced transaction costs. That means we should look to a level playing field for pre-installation and the three ideas above are different approaches.

The idea of opening Windows to others and putting a government agent squarely in the middle of choosing what goes into Windows should be self-refuting. This would be a feeding frenzy and coupling that with processes that would make a lawyer happy would mean that the next Windows OS would not be Windows XP but Windows 3000. The Final Judgment sensibly avoided even the hint of this idea, and it should not be considered now merely because the break up has been killed off, at least for now, by the D.C. Circuit.

Less intrusively, we would come closer to achieving competitive neutrality and thereby also guard against additional attempts at monopoly maintenance if we required Microsoft to live with the same across-network installation used by its competitors. In many ways, the question of whether we want to try to police tying to the OS with the network in place turns on these questions about the mechanics of distributing software. The analysis cuts both ways. If we think that we might cause serious consumer harm by requiring Windows IM to be installed across the network, then we also should think that Microsoft has a competitive advantage by pre-installing it. If we think that it would be a small matter to force Microsoft to be

an outsider to Windows – to force it to have new features downloaded – then we probably lose little by doing so, and this has the benefit that third parties now have access to the same tools of decentralized coordination.

If your reaction to this idea is, “You’ve got to be kidding, you expect me to install my software over the network, using a 56K modem that disconnects every 5 minutes,” two responses. First, if that is a serious point, you get a good sense of the distributional advantage Microsoft may have by simply being able to add software to Windows. Second, you should get used to this idea, because the coming world of online software services, such as Microsoft’s *\_.Net* initiative, is precisely about this kind of software distribution.

On the question of whether pre-installation or downloading matters, we should look to the behavior of participants in the industry. Certainly in the pre-network era, there is little doubt that distribution with Windows was seen by most industry players as quite valuable. This fact drove the card game described earlier. Even now, we should look to Microsoft’s decisions in including and excluding items from Windows XP as providing valuable information. Microsoft explains the choice to exclude Java as in part about irrelevance, in part about contract dispute fallout, but it seems clear that the decision is also partially strategic. Microsoft will make it harder for Java distribution if Java is unbundled from Windows, so that Java is visible from within Windows but must be installed over the network. In similar fashion, Microsoft intends pre-installation of Windows IM, not just visibility coupled with downloading. These decisions suggest that Microsoft itself believes that something turns on the mechanics of distribution, even with an always-on network.

Finally, consider the third idea, competition at the OEM level. Although Microsoft sells CDs directly to consumers, its main way of distributing Windows to consumers is through deals with OEMs, who pre-install Windows before shipping their computers. The OEMs are then a key means of distributing software, and we could just look to strengthen competition there. We clearly can achieve some of that through remedies directed at Microsoft’s contracting practices (see the next section below), but the real question is whether we need to control the scope of Windows, whether achieved through integration or bundling, and pricing. Mandatory versioning, discussed in the next subsection, would address the former issue, and, as set forth in two subsections, I am skeptical that the case makes out a basis for controlling pricing of Microsoft’s Windows add-on components.

Unleashing the forces of competition at the DI level will be quite powerful, as the early evidence, courtesy of AOL, suggests. Press reports describe an aggressive campaign by AOL to alter the Windows desktop to advertise AOL and to include a series of pop-up notices at various points in the user experience as an additional nudge. Besides whatever lump-sum payments AOL might make to OEMs, it would pay \$35 a head for sign-ups. Consumer advocates are nervous about this, and Microsoft describes the behavior as “anti-consumer,” but this is precisely what we should hope would happen, multiplied many-fold (A. KLEIN [2001]; see also BUCKMAN AND MCWILLIAMS [2001]).

We also need to address direct sales to consumers off of CDs. Microsoft has no unique distributional advantage there – AOL, among others, has demonstrated that it can create CDs with the best of them. Could Microsoft have an advantage in obtaining shelf space? That takes us to another complex antitrust area (see, for example, FARRELL [2001]), but in any event, nothing in the case as litigated really addresses this issue.

As to presence, I have suggested above a two-period remedy, a downloading only period followed by a period of DI neutrality. The core notion here is that the central harm of Microsoft's monopoly maintenance was the distributional disadvantage it imposed on competitors. We should seek to restore competition and undo the harm created by Microsoft's activities by relatively disadvantaging Microsoft's distribution. One approach that might do that and do the best job of restoring competition would be to do both the mandatory downloading remedy and the OEM neutrality remedy, in sequence, as described above. Note that the Final Judgment makes no effort to remedy directly the consequences of the distribution advantage created by Microsoft's behavior. The break-up remedy did not get at this, and the conduct remedies ignore the issue as well. The suggested two-step distributional remedy gets at this directly.

It is important to acknowledge that the first stage of this remedy, direct distribution only, would likely impose extra transaction costs on at least some consumers. Indeed, the point of the remedy is to make it more difficult for Microsoft to distribute its software for a period to correct for the distributional advantages it obtained from its illegal monopoly maintenance. Whether we should impose the direct distribution remedy or instead just skip ahead to the period of DI neutrality will clearly turn on the size of the losses imposed on consumers during the first period, a matter as to which evidence might be taken during the new remedies phase in the district court.

The merits of this also turn on one's sense of how equilibria are restored in distorted markets. Compare two situations. In the first, think of a ball in a bowl, where the ball artificially sits on a ledge in the bowl and thus cannot come to rest at the bottom of the bowl. Remove the ledge and we know that the ball will get to its resting spot, just as it would have had the barrier never existed. In the second, think of one ball and many bowls. Here the artificial act pushed us from one bowl to another, and in the second bowl, the ball sits on the ledge. Remove the barrier and the ball comes to rest in the second bowl, but without a nudge, we never get back to the first bowl.

If you believe that just preventing Microsoft from engaging in anti-competitive behavior going forward will suffice to restore competition, we do not need the direct-distribution-only remedy. Yes, it would also act as a penalty, which would help to deter future violations, but we can do that directly through monetary fines, and those will not risk disrupting consumer transactions. If instead you believe that Microsoft's behavior has pushed us into the wrong bowl, then we need something like the direct-distribution-only remedy to help to restore competition, and we need to make sure that the gains to consumers of doing that exceed the interim losses that may be imposed on them.

(C) *Mandatory Versioning*. Both the period of direct distribution and the period of meaningful DI neutrality depend on the notion of mandatory versioning. Section 3(g) of the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b, p. 8]) relies on that as well, so consider it for specificity:

“g. Restriction on Binding Middleware Products to Operating System Products. Microsoft shall not, in any Operating System Product distributed six or more months after the effective date of this Final Judgment, Bind any Middleware Product to a Windows Operating System unless:

i. Microsoft also offers an otherwise identical version of that Operating System Product in which all means of End-User Access to that Middleware Product can readily be removed (a) by OEMs as part of standard OEM pre-installation kits and (b) by end users using add–remove utilities readily accessible in the initial boot process and from the Windows desktop; and

ii. when an OEM removes End-User Access to a Middleware Product from any Personal Computer on which Windows is pre-installed, the royalty paid by that OEM for that copy of Windows is reduced in an amount not less than the product of the otherwise applicable royalty and the ratio of the number of amount in bytes of binary code of (a) the Middleware Product as distributed separately from a Windows Operating System Product to (b) the applicable version of Windows.”

This obviously depends on two key definitions (p. 15):

“r. ‘Middleware Product’ means

i. Internet browsers, e-mail client software, multimedia viewing software, instant messaging software, and voice recognition software, or

ii. software distributed by Microsoft that (1) is, or has in the applicable preceding year been, distributed separately from an Operating System Product in the retail channel or through Internet access providers, Internet content providers, ISVs or OEMs, and (2) provides functionality similar to that provided by Middleware offered by a competitor to Microsoft.

q. ‘Middleware’ means software that operates, directly or through other software, between an Operating System and another type of software (such as an application, a server Operating System, or a database management system) by offering services via APIs or Communications Interfaces to such other software, and could, if ported to or interoperable with multiple Operating Systems, enable software products written for that Middleware to be run on multiple Operating System Products. Examples of Middleware within the meaning of this Final Judgment include Internet browsers, e-mail client software, multimedia viewing software, Office, and the Java Virtual Machine. Examples of software that are not Middleware within the meaning of this Final Judgment are disk compression and memory management.”

These are a matter of substantial importance and dispute, but ignore that for now and focus on the concepts at work.

It is important that Microsoft can continue to innovate by making features available from the OS. Like the neutrality trustee straw-man set forth above, any struc-

ture that required pre-clearance of new features of Windows would create an intolerable burden on software innovation, one that in many ways would track the line of commerce restrictions that emerged from the break up of AT&T (SHELANSKI AND SIDAK [2001, pp. 95f.]). In U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b, Section 3(g)], the government understood this and avoided that kind of barrier to innovation.

At the same time, if Microsoft can just continue to add features to Windows to be accepted by market participants on a mandatory basis, Microsoft can repeat the behavior of this case and continue to maintain the dominance of Windows. The trick is to allow innovation, while controlling what Microsoft can do.

Mandatory versioning creates the infrastructure to accomplish this result. My version of this is different from that in Section 3(g) in an important way, as I will describe below, but the core idea is similar. We need to make it possible for DIs to reject Windows features. Obviously, there is an important question of scope – which features are we talking about – and the elided definitions issue above gets to that. But on the rejection notion, only if Microsoft is forced to present multiple version of Windows to DIs do we make it possible for features to be rejected. The requirement that Microsoft separate out middleware components means that we will not end up with Microsoft middleware, visible and present, simply because Microsoft controls the content of Windows. Section 3(g)(i) takes away that control. This will permit level competition in free middleware software components, whether done through the downloading remedy or simply through competition at the DI level.

Mandatory versioning is also important if we want to implement the first stage of the two-stage remedy I have suggested. Achieving DI neutrality going forward will do nothing to remedy the harms to competition inflicted by Microsoft through its distortion of the channels of distribution. The natural remedy for that is a corresponding correction, a period of distortion against Microsoft. The suggested moratorium period is an approach to that. It is inconceivable that we would want to stop all distribution of Windows through DIs for any period. Absent mandatory versioning, if we just achieve DI neutrality, this would mean that Microsoft would distribute Windows XP through the DIs as presented by Microsoft to the DIs.

With mandatory versioning, we can implement the corrective moratorium period. Microsoft would distribute the baseline Windows systems – think of this casually as Windows 2000 or Windows ME – through DIs. Microsoft could distribute the full Windows XP directly to end-users, through its website or via CD. Microsoft's Windows Update feature would facilitate downloading of the add-ons for consumers who want them.

*(D) Pricing Add-On Components.* Section 3(g)(ii) of the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b]) will impose a mandatory price gap between the basic version of Windows and more advanced versions in sales to DIs. The impetus behind it is clear. On this story, if Microsoft can continue to incorporate features into the OS and not offer a separate charge for

those features, it will make it very difficult for competing features to arise. On my story, that may have made sense in the pre-network world where it would have been difficult, if not impossible, to coordinate distribution of those features from someone other than Microsoft, but it is hard to justify in the networked world.

The problem is that the case as litigated to date does not really support a price remedy. Consider the following two scenarios. Assume we require installation over the network for “new” Windows functions, such as IM (instant messaging). Microsoft announces its price for Windows Basic. Users can upgrade to Windows Deluxe by downloading the free Microsoft IM program from its website. Competitors can post their IM software on their websites, and competition ensues. Consider version 2 of this scenario where we look to foster competition at the DI level. Microsoft produces Windows Basic and Windows Deluxe. It sells them at the same price to OEMs who choose one or the other. Competitors can also go to OEMs and offer free software for installation, and again OEMs are free to accept or reject the software.

In both cases, Microsoft receives the same payment for Windows Basic and makes the additional component available for no separate charge. As discussed above, this two-piece pricing strategy – one free component, a complement for sale – is common in the software business. Section 3(g)(ii) would deny Microsoft – and only Microsoft – access to this strategy. The government wants the price reduction clause because otherwise “OEMs, in effect, ... pay for Microsoft Middleware Products, even if they want to remove them, and thus would provide a substantial disincentive for the OEMs to license and install competing Middleware Products, thereby foreclosing opportunities for those products to create competition for the Windows platform” (UNITED STATES [2000, Comment 3.g.[2]]).

There is something of a sleight of hand here. It is always the case in the two-piece pricing strategy that someone “in effect” has to pay for the free component elsewhere. But that hardly represents a reason for not allowing the strategy, as we would clearly permit it in a competitive market. Moreover, in some sense, anyone has the power to set the price of the complement at zero. Microsoft can do so, but so can AOL and Yahoo, as they indeed have done.

The D.C. Circuit addressed price issues at a number of points in its opinion. The lower court did not find that Microsoft engaged in predatory pricing, and given that, the court of appeals stated (U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 42]):

“The rare case of price predation aside, the antitrust laws do not condemn even a monopolist for offering its product at an attractive price, and we therefore have no warrant to condemn Microsoft for offering either IE or the IEAK (Internet Explorer Access Kit) free of charge or even at a negative price.”

With good reason, as we know that the fear of antitrust liability for low prices will interfere with a firm’s willingness to reduce prices, and that is a central driver of consumer benefits in competitive markets. Predatory pricing cases are hard to make out generally and that should be especially so in software markets where we

see even new firms give away their products. Without much more extensive development on the tying remand, I see no basis in the case for the pricing remedy that Section 3(g)(ii) represents.

Note that there are also mechanical questions about the workings of Section 3(g)(ii). When we distribute the middleware product separately can we do so without shared components and the expected components? The royalty reduction is then only calculated based upon the unique components that come with the product; this effectively would assign the revenues associated with the shared component to the OS. If, in contrast, and this is probably what is intended, the middleware product calculation must include all shared components, this leaves the royalty as only those components that are unique to the OS.

*(E) Administration and Alternatives.* There have been a number of explicit or implicit lines drawn in the core five remedies, and I should discuss those. There are at least three key questions: (i) Should the remedies apply to so-called middleware add-ons or to all add-ons? (ii) Should the remedies look backward, so that we could define certain functions as add-ons and subject those to the same set of remedies? and (iii) Are there different costs in unbundling or de-integrating, so that removing function visibility and access should suffice without the additional step of limiting presence?

First, the DI visibility flexibility remedy would apply to all icons, so that there would be no mandatory icons or other functionality entrance points. That would apply to middleware – things we think might have platform potential and therefore compete with the OS – as well as other new functions, such as Windows XP’s planned “Scanner & Camera Wizard,” which presumably will not emerge as a competitor of the OS. Although a plausible case could be made out for distinguishing these, the administrative costs of doing so probably would be high, and in my judgment, the cleaner remedy is no mandatory icons.

Second, the mandatory versioning remedy, which in turn feeds into the two distribution remedies, would be limited to middleware, however defined. Differences between the government’s definition and that of Microsoft would have to be resolved (see UNITED STATES [2000, Sections 7.q. and 7.r.] and MICROSOFT [2000a, Comments on Sections 7.q. and 7.r.]), but the focus is platform potential. This presumably would not cover the Scanner & Camera Wizard – an item of some controversy (WILKE AND BANDLER [2001]). Here the right default rule is exclusion, so that we have identified the precise scope of the burden imposed on Microsoft. That does have the disadvantage of requiring us to tie down a vision of middleware now, but the alternative seemingly is to require Microsoft to separate out all new features from Windows proper. Certainly, without a resolution of the tying claim – and recall that I think Microsoft should win that claim as to the browser – there would seem to be little basis for forcing Microsoft to separate out all new features. Put differently, we are entitled to remedy what Microsoft did, but that does not empower us to impose a broad open access obligation on Windows. That said, on remand, the district court should conduct hearings to determine the costs

of implementing mandatory versioning. The lower the costs of doing so, the more we should fold into the middleware category, given the difficulties of forecasting what will emerge as a platform possibility.

Finally, what we learn about the costs of mandatory versioning should also help us determine whether we should just live with the flexibility remedy and not pursue my suggested distributional remedies. As discussed above, this turns on part on the question of how competition is restored to a market infected with prior anti-competitive acts. AOL's recent efforts with OEMs to increase its visibility on the desktop suggests the icon remedy itself may be quite powerful, and mandatory versioning will clearly impose extra costs. That said, versioning is an extraordinarily common strategy in information markets (SHAPIRO AND VARIAN [1999, p. 53]), so we should be skeptical of the claim that we would be forcing Microsoft to do something that can be done only with great difficulty. It is done all of the time.

### 6.2.2 Section 3(g) Details

While Section 3(g) clearly has price consequences, standing alone, it is not crystal clear whether this is also a visibility remedy and/or presence remedy. The definition of "End-User Access," set forth in Section 7(j) of the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b, p. 14]), helps to sort this out:

"'End-User Access' means the invocation of Middleware directly or indirectly by an end user of a Personal Computer or the ability of such an end user to invoke Middleware. 'End-User Access' includes invocation of Middleware by end users which is compelled by the design of the Operating System Product."

Start with the second sentence of the definition. As discussed above, the appeals court found that at least some invocations of IE by the OS were not shown to be anti-competitive steps of monopoly maintenance. I argued that the court paid too little attention to the possibility that these design choices, such as the use of ActiveX controls, were made simply to exclude the use of Navigator. That said, absent smoking emails of the sort seen in this case, it will be difficult to sort out whether, for example, an ActiveX control was used in presenting help info on the OS because it made sense or because it justified the presence of IE on the computer. The second sentence flatly bars Microsoft from designing to anything other than the lowest common denominator – the features supported by all browsers – and that seems like a serious mistake.<sup>6</sup>

Turn to the first sentence. The limit on "direct" invocation of middleware sounds like a visibility remedy. So remove the icons from the desktop and take it out of the Start Menu, but leave the underlying software on the computer it-

---

<sup>6</sup> Microsoft sought to rewrite this sentence to make clear that it could indeed use IE to display the Windows help system, but the government rejected the changes. Compare MICROSOFT [2000, Comment 2 on Section 7(j)] with UNITED STATES [2000, Comment 7.j.[2]].

self. The limit on “indirect” invocation sounds much more like a presence remedy, so that, to use Microsoft’s example, Intuit’s Quicken could not invoke IE to get supplemental information for a consumer from Intuit’s website. To limit indirect invocation means that the software either is not present or is present but dead to all uses.<sup>7</sup>

That makes this both a visibility and a presence remedy. The basis for the visibility remedy is the potential consumer confusion issue discussed by the D.C. Circuit. The basis for the presence remedy is the fragmentation rationale, namely, that the presence of Microsoft’s middleware will make it much more difficult for competing middleware to gain a sufficiently large user base to cover the fixed costs of software development, and that in turn will deter middleware entry. Both of these are aspects of the D.C. Circuit’s conclusions regarding design monopoly maintenance, and do not depend necessarily on how the tying remand is resolved. That said, as a remedy for the design monopoly maintenance violation, this should be limited to software that might reasonably act as a platform alternative to Windows.<sup>8</sup>

### 6.2.3 Quarantining Monopoly Power Abuse

Given the core finding of monopoly maintenance, I would expect an imposed remedy or a consensual settlement to include the following provisions – all of which are taken from the Final Judgment (U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b]) – to attempt to create a structure to constrain Microsoft’s ability to misuse its monopoly position:

(a) *DIs*. Microsoft will be barred from taking adverse actions against a distributional intermediary for the DI’s actions as to any product that competes with a Microsoft product (3(a)(i), 3(d)). Microsoft will be required to offer uniform terms for licensing Windows (3(a)(ii)). Both of these will reduce Microsoft’s ability to pressure DIs to comply with Microsoft dictates.

(b) *Technical Impairment*. Microsoft will be barred from some form of taking actions that degrade the performance of competitor middleware (3(c)).

(c) *Agreements Limiting Competition, Exclusive Dealing, and Contractual Tying*. To combat the Apple situation, where Microsoft was found to have threatened to kill off or delay Office if Apple did not see the light, Microsoft will be barred in some form from inducing a third party from limiting its development of software (3(h)). In the proposed final judgments from the government and Microsoft, there was conceptual overlap with important difference in details. Microsoft would also face limits on its ability to engage in exclusive dealing (3(e))

<sup>7</sup> Again, Microsoft sought a change suggesting that “indirectly” be deleted, see MICROSOFT [2000a, Comment 1 on Section 7(j)], and again the government rejected the change. See UNITED STATES [2000, 7.j.[1]].

<sup>8</sup> One more remedy for software market structure should be noted. The remedies suggested by both Microsoft and the plaintiffs would require Microsoft to continue to sell its old OS for three years at a capped price when it released a new OS.

and contractual tying (3(f)). Of course, it was precisely a similar limit on contractual tying that led to *Microsoft II*, so there is good reason to think this may work poorly.

As to all of these, the precise details will be difficult, but for better or worse, we can expect the remedy here to address all of these issues.

#### 6.2.4 Disclosure Obligations

Section 3(b) of the Final Judgment would impose a different nondiscrimination obligation on Microsoft relating to the disclosure of APIs, communications interfaces, and technical information. The shadow case against Microsoft claims that Microsoft uses its superior knowledge of the insides of Windows to help it write applications for Windows, such as Microsoft Office. Many of these claims crystallize around the notion that Windows has nondisclosed APIs (the applications programming interface), hooks in the software that developers use to share code from Windows itself. Section 3(b) would require Microsoft to disclose the APIs and other related information to outsiders, just as it does to Microsoft insiders.

The underlying claim is somewhat interesting. Microsoft is in a tricky position in both wanting developers to create applications for Windows and in also competing directly in that market. Developers would like assurances that Microsoft will not disadvantage outsiders in that competition, etc. As a practical matter, it would seem some inside advantage is almost inevitable even if Microsoft did its utmost to maintain equality between insiders and outsiders.

In any event, the APIs allegation has played no direct role in the case itself, and therefore as a “remedy” it appears to come out of the blue. Microsoft (MICROSOFT [2000c]) quite understandably sees this as a forced transfer of its valuable intellectual property. It might be true that disclosure would foster competition in the applications market, but that is not what the plaintiffs’ case was about. Instead it was about competition in the OSs market, and more particularly it was about Microsoft’s ability to use Windows as a device for making visible and present additions to the OS that would compete with products such as Navigator and Java. The questioned conduct related to integration and tying and therefore as I have talked about things, visibility and presence, and not to a use of insider knowledge. I thus see little basis, within the confines of the case itself, for a broad disclosure obligation such as that set forth in Section 3(b).

Will this mean that Microsoft will have a sizable advantage over outsiders? Probably, but unless we think that the finding of monopoly maintenance means that Microsoft forfeits everything associated with the monopoly that it lawfully acquired, we need some limiting principle, and a tie to what actually animated the case makes sense. In contrast, to prevent Microsoft from using API disclosure to third parties as a club, we must have the same nondiscrimination duty among outsiders that I have already discussed. Microsoft proposed a version of this in its proposed remedy (MICROSOFT [2000b, para. 6]).

## 7 Conclusion

In this paper, I make a number of points about the Microsoft case itself and the next steps that should take place. In particular, I argue that:

(A) *No Liability for Tying*. Microsoft should not be found liable under the Sherman Act for tying Internet Explorer to Windows. In the pre-networked world, Windows played the central role in coordinating the sharing of software. Incorporating a browser would have been perfectly consistent with that role.

(B) *The Drop in the Cost of Software Coordination*. The rise of the network changes how software should be distributed and changes the role of Windows in software coordination. There is less of a need for mandatory incorporation of software into Windows, as decentralized distribution and coordination is now possible.

(C) *Distorted Distribution Channels*. As found by the D.C. Circuit, Microsoft engaged in impermissible monopoly maintenance. In so doing, Microsoft distorted the channels for software distribution and added software to Windows for the purpose of raising the cost of distribution of rival software.

(D) *Distribution Remedies*. A proportionate Microsoft remedy should address that distributional distortion and seek to prevent future distortions. These remedies should: (1) foster desktop flexibility for distributional intermediaries, so that there are no mandatory icons on the Windows desktop or spots reserved in the Start Menu or its equivalent; (2) require Microsoft to engage in mandatory versioning, so that it issues Windows versions with and without any new middleware that it adds to Windows; (3) impose a moratorium period of six months to 2 years during which Microsoft would be able to distribute through distributional intermediaries only the baseline Windows without new middleware, while permitting distribution of the full version of Windows via CD or Microsoft's website; (4) after the moratorium period, rely on competition among software producers and others for distributional intermediary shelf space to control software distribution abuses; and (5) sunset, perhaps after a period of three years.

(E) *Equilibria and Restoring Competition in Distorted Markets*. The direct-distribution-only remedy will likely impose interim costs on consumers. We need to assess those costs and understand whether they need to be paid. That turns in part on whether preventing further anti-competitive acts will suffice to create the competitive level that would have existed absent Microsoft's acts, or whether such competition can be restored only through a more direct measure such as the suggested direct-distribution-only remedy.

## References

- ANDERSON, R. [2000], "The End of DLL Hell," <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnw2k/html/dlldanger1.asp>.
- BAKOS, Y., AND E. BRYNJOLFSSON [1999], "Bundling Information Goods: Pricing, Profits, and Efficiency," *Management Science*, 45, 1613–1630.

- BARKSDALE, J. [1998], "Direct Testimony in U.S. v. Microsoft," <http://www.usdoj.gov/atr/cases/f1900/1999.htm>.
- BRANDENBURGER, A. M., AND B. J. NALEBUFF [1996], *Co-opetition*, Currency Doubleday: New York–London–Toronto–Sydney–Auckland.
- BUCKMAN, R., AND G. MCWILLIAMS [2001], "Compaq to Give AOL's Web Service Exclusive Position on Most XP PCs," *The Wall Street Journal*, July 27, p. B2.
- CARLTON, D. W. [2001], "A General Analysis of Exclusionary Conduct and Refusal to Deal: Why Aspen and Kodak are Misguided," *Antitrust Law Journal*, 68, 659–683.
- AND M. W. WALDMAN [1998], "The Strategic Use of Tying to Preserve and Create Market Power in Evolving Industries," Working Paper #6831, National Bureau of Economic Research: Cambridge, MA.
- COLBURN, D. M. [1998], "Direct Testimony in U.S. v. Microsoft," <http://www.usdoj.gov/atr/cases/f2000/2045.htm>.
- FARRELL, J. [2001], "Some Thoughts on Slotting Allowances and Exclusive Dealing," Working Paper, U.S. Department of Justice, Washington, DC.
- GATES, W. [1995], "The Internet Tidal Wave," <http://www.usdoj.gov/atr/cases/exhibits/20.pdf>.
- [2001], "Transcript of Press Conference of June 28, 2001," <http://www.microsoft.com/presspass/trial/jun01/06-28newsconftrans.asp>.
- KLEIN, A. [2001], "AOL to Offer Bounty for Space on New PCs," *The Washington Post*, July 25, p. A01.
- KLEIN, B. [2001], "The Microsoft Case: What Can a Dominant Firm do to Defend its Market Position?" *Journal of Economic Perspectives*, 15, 45–62.
- LESSIG, L. [2000], "Brief of Professor Lawrence Lessig as Amicus Curiae," <http://cyberlaw.stanford.edu/lessig/content/testimony/ab/ab.pdf>.
- MICROSOFT [1999], "Windows Installer Service Overview," <http://www.microsoft.com/windows2000/docs/WIS-Pro.doc>.
- [2000a], "Comments on Plaintiffs' Revised Proposed Final Judgment," <http://www.microsoft.com/presspass/trial/may00/05-31comments.asp>.
- [2000b], "Proposed Final Judgment," <http://www.microsoft.com/presspass/trial/remedies/05-10FinalJudgment.asp>.
- [2000c], "Reply to Plaintiffs' Response to Microsoft's Comments on their Revised Proposed Final Judgment," <http://www.microsoft.com/presspass/trial/jun00/06-06reply.asp>.
- [2001a], "Microsoft Announces Greater OEM Flexibility for Windows," Press Release, <http://www.microsoft.com/presspass/press/2001/Jul01/07-11OEMFlexibilityPR.asp>.
- [2001b], "Petition for Rehearing," <http://www.microsoft.com/presspass/trial/appeals/07-18motion.asp>.
- RASMUSEN, E. B., J. M. RAMSEYER, AND J. S. WILEY JR. [1991], "Naked Exclusion," *American Economics Review*, 81, 1137–1145.
- SHAPIRO, C., AND H. R. VARIAN [1999], *Information Rules*, Harvard Business School Press: Boston, MA.
- SHELANSKI, H. A., AND J. G. SIDAK [2001], "Antitrust Divestiture in Network Industries," *The University of Chicago Law Review*, 68, 1–99.
- STALLINGS, W. [2001], *Operating Systems*, 4th ed., Prentice Hall: Upper Saddle River, NJ.
- TANENBAUM, A. S. [2001], *Modern Operating Systems*, 2nd ed., Prentice Hall: Upper Saddle River, NJ.
- U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001], "United States v. Microsoft," <http://ecfp.cadc.uscourts.gov/MS-Docs/1720/0.pdf>.
- U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999], "United States of America v. Microsoft Corporation: Findings of Fact," 84 F. Supp. 2nd 9 (D.D.C. 1999), <http://www.dcd.uscourts.gov/ms-findings2.pdf>.

- [2000a], "United States of America v. Microsoft Corporation: Conclusions of Law," 87 F. Supp. 2d 30 (D.D.C. 2000), <http://www.dcd.uscourts.gov/ms-conclusions.pdf>.
- [2000b], "United States of America v. Microsoft Corporation: Final Judgment," 97 F. Supp. 2d 59 (D.D.C. 2000), <http://www.dcd.uscourts.gov/ms-final2.pdf>.
- U.S. FEDERAL TRADE COMMISSION [2000], "Decision and Order In the Matter of America Online, Inc. and Time Warner, Inc.," <http://www.ftc.gov/os/2000/12/aoldando.pdf>.
- UNITED STATES [2000], "Plaintiffs' Summary Response to Microsoft's Comments on Revised Proposed Final Judgment," <http://www.usdoj.gov/atr/cases/f4800/4893.htm>.
- WHINSTON, M. D. [1990], "Tying, Foreclosure, and Exclusion," *American Economics Review*, 80, 837-859.
- [2001], "Exclusivity and Tying in U.S. v. Microsoft: What We Know, and Don't Know," *Journal of Economic Perspectives*, 15, 63-80.
- WILKE, J. R., AND J. BANDLER [2001], "New Digital Camera Deals Kodak a Lesson in Microsoft's Methods," *The Wall Street Journal*, July 2, p. A1.
- AND D. CLARK [2001], "Microsoft Pulls Back Support for Java, Dealing New Blow to Rival Technology," *The Wall Street Journal*, July 18, p. A3.

Randal C. Picker  
The University of Chicago Law School  
1111 East 60th Street  
Chicago, IL 60637  
USA  
E-mail: [r-picker@uchicago.edu](mailto:r-picker@uchicago.edu)

## Windows as an Institution Organizing the Markets for Applications Software

*Comment*

by

CHRISTOPH ENGEL

### *1 Introduction*

A common enemy is a valuable resource for social organization. For many techies Microsoft is the archetypical common enemy. It is indeed hard not to be impressed by all the evidence of bad faith uncovered during the Microsoft antitrust case. Yet antitrust is not about assuaging a wounded sentiment of justice. It is about efficiency or, more generally, about welfare. In terms of welfare economics, it is not so obvious that structural or conduct remedies directed at Microsoft are beneficial. In light of this, it is thus understandable that the Court of Appeals has not much left over from Judge Penfield Jackson's verdict in the Microsoft case.<sup>1</sup>

This comment takes into consideration only one – albeit central – aspect of the case. The Microsoft operating system, Windows, implicitly serves as an institution that organizes the markets for applications software (Section 2). It is not the only institution that can serve this purpose (Section 3). But in comparison with the alternatives, Windows has a considerable number of advantages (Section 4). Moreover, a transition from Windows to alternative institutions would be both costly and cumbersome (Section 5). Even if institutional change were desirable, antitrust proceedings are a poor mechanism for bringing it about (Section 6). Antitrust could thus at best contribute to keeping evolutionary paths open to alternative ways of organizing the markets for applications software (Section 7).

### *2 Windows Organizing the Markets for Applications Software*

Computer programs are not standalone products. They share many components with other programs. In the Microsoft world, most of these shared components

---

<sup>1</sup> U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001]; appeal on U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000b], [1999], and [2000a].

come with the operating system. Applications software is “written to” this operating system. Its technical features enable it to exploit the functionality of the operating system (PICKER [2002]). Put differently, Windows is not only a product, competing for demand on the market for operating systems. It is simultaneously an institution organizing the markets for application software (PICKER [2002, Section 3]; confer also SIDAK [2001, pp. 15 and 59]).

The organizational effect stems from partly standardizing the quality of applications software: it must be Windows-compatible. The organization technology combines incentive with code. The large installed base of Windows generates an incentive to write software that can run in a Windows environment. Once software opts for that environment, the governance effect of Windows as an institution becomes strict. Applications software will only run on Windows if it fits to the application programming interfaces (APIs) of Windows.<sup>2</sup> With respect to the markets for application software, Windows is thus an instance of governance by code, not by law or by social norms (confer LESSIG [1999]). Finally, it is worth noting that Windows, by offering a large set of APIs, simultaneously organizes a huge number of markets for applications software.

Is this governance effect of Windows a good thing? Obviously, the general pros and cons of market organization apply (FURUBOTN AND RICHTER [1997, ch. VII]). Substitution gaps are narrowed down. Market transparency is enhanced. But innovation must circumvent the quality standard. And market transparency can be exploited to coordinate behaviour or enforce a cartel.

The specifics of software markets are more important. The markets for applications software have to be organized in order for the users to exploit the economies of scope, i.e., in order to derive the benefits from complementarity. Accordingly, complementarity does not change the organization technology. A complementarity standard is still a quality standard. But it adds Microsoft as a possible organizational entrepreneur. The argument also works the other way around. From a welfare perspective, complementarity is only of interest if users have a preference for the specific utility derived from simultaneously using both complementary elements. Vertical integration is an almost natural response to this preference. If the producer of one component is able to organize the markets for the other components, vertical disintegration becomes possible. This effect is particularly beneficial if economies of scale for one of the components are more pronounced than for the other. Organizing the latter market then frees it up for competition. This is exactly what has happened with operating systems and applications software. An operating system is the link between hardware and software. It therefore exhibits pronounced economies of scale for the users, i.e., large positive network externalities (KATZ AND SHAPIRO [1985]).

Although network externalities for applications software tend to be smaller, they do exist. A typical case is file sharing among users. It presupposes that the

---

<sup>2</sup> For the technical details, see U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 17f.].

shared files are compatible with the software used by all participants. Windows generates a beneficial side effect here, too. If applications software is written for Windows, this fact often makes it also easier to share files with other Windows-based computers.

### 3 *Alternative Institutions*

Windows is not the only institution for organizing markets for applications software. Picker demonstrates the feasibility of what he calls a networked solution (PICKER [2002, Section 4]); in the original version of the paper discussed at the conference, he called it a peer-to-peer solution. By this he meant that other producers of applications software offered shared components for download over the Internet. Whether these peers will want to exploit this new option seems less certain, however. Today, the producers of applications software get the organization of their own markets for free. It is given to them by Microsoft. Although the transaction cost for self-organization has fallen dramatically since the development of integrated networks, it is still considerably above zero. But these producers and their customers might value their independence from Microsoft highly enough to accept this transaction cost. The growing Linux community may be proof of this.

Technically and economically, peer-to-peer coordination is only a halfway solution. It still combines production and market organization. Technically, pure intermediation should be possible. One might call this an *Internet-type* solution. In this scenario, there would just be APIs out there for download. A producer of applications software would simply specify which APIs are necessary for using his software successfully. Personal computers would have a device that automatically goes through this list. If a necessary API were already stored on the hard disk, the device would only establish an automatic link. For the missing APIs, the user would be guided to a service from which they could be downloaded free of charge or for a small price. But of course, the incentive problem for bringing such an Internet-type solution about is even stronger than for the peer-to-peer solution originally proposed by Picker. The following argument focuses on the alternative between Windows and the peer-to-peer solution. It thus looks at the less radical of the two possible networked solutions. It does so because this less radical solution seems more likely to establish itself in practice.

### 4 *Comparing Institutions*

A peer-to-peer solution is feasible, but is it better? A common prejudice among economists seems to be that greater decentralization is always a good thing. But in this particular case, the prejudice may well be misleading. The list of pros and

cons is long and equivocal (confer SALOP AND ROMAINE [1999]; SHELANSKI AND SIDAK [2001]).

The main drawback of the Windows solution is its effect on the competition among operating systems. In the present state of the world, Microsoft organizes the markets for applications software free of charge. In other words, the organizational effect is a positive externality. Even if Microsoft offers the application software itself, it normally does not try to make it harder for competing software to run on Windows. Now Microsoft is not a charity. However, setting the positive externality becomes rational for it once one looks at the markets for operating systems.<sup>3</sup> This is exactly what has been dubbed the “applications barrier to entry”<sup>4</sup> in the Microsoft case.

A related effect is of even greater practical importance. As the antitrust case unfolded, Microsoft was afraid that, in the future, producers of applications software might write their programs to browsers and other “middleware,” and no longer to Windows.<sup>5</sup> Microsoft does perceive itself as engaged in a war over standards with producers of middleware (confer SHELANSKI AND SIDAK [2001, pp. 66–69]). It regards middleware as a potential substitute, competing for the hub function. Giving producers of applications software access to a full array of APIs creates a barrier to entry, making this type of substitutive competition more demanding and costly.

But it has many advantages, too. The main positive effect of Windows is precisely the reliable and cheap organization of many markets for applications software. The peer-to-peer alternative is able to offer this advantage, too. But it has the drawbacks of any standardization cartel, whereas Microsoft is an outsider to most markets for applications software. In more abstract terms, the conflict between Microsoft and peer-to-peer for organizing applications software is an instance of institutional competition, not product competition. Normatively it is far from certain under which conditions institutional competition enhances welfare (out of the rich literature see, e.g., GERKEN [1995], MONOPOLKOMMISSION [1998], MÜLLER [2000], TJONG [2000]).

Antitrust scholars distinguish two different types of justifications. Firstly, the pro-competitive effects of a restriction on competition in one market can outweigh the otherwise illegal restriction of competition in another one.<sup>6</sup> This is why the pro-competitive effects on the markets for applications software come into play. But re-

<sup>3</sup> In the antitrust case, markets were defined more narrowly. The District Court and the Court of Appeals unanimously defined the relevant market as one for *Intel-compatible* operating systems: U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [2000a, p. 1]; U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 15–19]. This is convincing for estimating monopoly power. But our argument is about barriers to entry for suppliers of a qualitatively different operating system.

<sup>4</sup> U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 20].

<sup>5</sup> U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999, §§ 68–77]; U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, p. 18].

<sup>6</sup> The German *Gesetz gegen Wettbewerbsbeschränkungen* (GWB) has a specific clause on this in § 36 I.

restrictions on competition may also be inevitable or at least beneficial for allocative efficiency.<sup>7</sup> In the case of Windows, there is a long list of such advantages.

The peer-to-peer solution tries to match the first of these advantages. By organizing decentralized access to APIs, it purports to strongly reduce network externalities. In a peer-to-peer world, a large installed base of an operating system is no longer a precondition for writing and selling applications software. But Windows is not only a complementary product for many different software applications, it is also a common interface between software and hardware. Technically it is true that the peer-to-peer approach might also be extended to the complementarity between hardware and the operating system. But this leads to an all-or-nothing choice. To give users the full utility of complementarity, peer-to-peer would have to be introduced across the board. If not, Windows would still be needed for the hardware–software interface. In that scenario, competition on the markets for operating systems would be as restricted as before. Only the positive externality for the markets of applications software would be cut out. Another way of putting the same point: there are considerable economies of scope between the production and distribution of an operating system and the organization of markets for applications software.

There is only a minimal cost for producing another unit of software. This creates a serious incentive problem. The producer must find ways to charge prices above marginal cost, or he will have to leave the market. One rational strategy is bundling: The operating system is preinstalled on a new computer. Since the difference between the average and marginal costs is not nearly as pronounced for hardware, the hardware manufacturer can then also charge a price for the operating system. If the APIs are integrated into the operating system, the same mechanism can also be used to generate revenue for the development of new APIs.

A further problem created by the peer-to-peer alternative arises in the internal relationship between the producers of complementary products. Complementarity means that single components are useless for the end user. The desired utility depends on the well-adjusted combination of all the components. In the structurally related field of telecommunications deregulation, this is called the problem of network integrity. One can obviously overcome the problem with vertical integration. But Windows gets the same result cheaper. It uses reputation as a credible guarantee. Reputation is very important for Microsoft, since its success in the market hinges upon its market share. To understand why, it is useful to distinguish between the technology effect and the network effect of network goods. The technology effect characterizes the utility a single user can derive from the program. The network effect characterizes the utility a user can derive from the fact that other users rely on the same program. For the technology effects, market share is (almost) irrelevant. For the network effect, it is crucial.<sup>8</sup> Reputation is hard to build, but easy to destroy. If the APIs from Windows are no longer reliable for some type

---

<sup>7</sup> There is no specific clause on this in the German statute. Efficiency defences and interventions into competition for mere political reasons both come under § 42 I 1 GWB.

<sup>8</sup> I owe this distinction to Günter Knieps.

of applications software, there is a risk that the demand for Windows will decrease. Given the network effect, such an erosion at the margin can quickly trigger a process of unravelling. Windows runs the risk of going out of business.

The same reputational logic helps overcome a related problem. In general, vertical disintegration is fraught with the holdup problem (WILLIAMSON [1985]). But since Windows risks its reputation if it tries to extract a rent from the producers of applications software, it is not likely to do so.<sup>9</sup>

Windows is also an elegant solution to another problem. Legally, software is of course protected. But enforcing these legal property rights is no easy task. Digitized information can be copied at no cost and without loss of quality. In the future, copyright management systems might be a solution. The technological threat would then be parried by technological means (BURK AND COHEN [2000]). But software producers must also solve the already mentioned problem of low marginal cost. Privately enforced property rights do not address this problem. However, bundling the APIs with the core elements of the operating system and selling both to the hardware manufactures solves both problems at the same time.

Finally, users might prefer Windows to the peer-to-peer alternative. It makes life much easier for them if they can trade with just one company. In telecommunications this is called a one-stop-shopping or a one-stop-billing arrangement. Moreover, adding a large variety of APIs to the operating system is of value because it opens up new options. Buyers know that the operating system will also serve them, should they later want to use new types of software (confer SIDAK [2001, pp. 16f.]).

### 5 *Transitional Issues*

Even if the peer-to-peer approach is preferable, due to the large installed base of Windows, users and producers of applications software will incur a substantial adaptation cost. Moreover, a war over standards between Windows and peer-to-peer is likely. It might easily result in the general deterioration of the penetration rate, or in a loss of positive network externalities. Put differently, switching from Windows to peer-to-peer might well be an all-or-nothing decision if the network externalities are still exploited.

### 6 *Antitrust as a Poor Technology for Institutional Change*

Even if one assumes that the transitional problems have been solved, it still seems questionable whether antitrust is an appropriate means for bringing the switch to a peer-to-peer world about. Antitrust proceedings are geared towards handling concrete conflicts, not towards designing a new regulatory policy. There is no place in antitrust dogmatics for comparing institutional options. The need to protect prop-

---

<sup>9</sup> Of course, this mechanism does not work the other way around. But given the lower barriers to entry into the markets for applications software, Windows might be less vulnerable to threats from the producers of such software.

erty and freedom from government intervention severely limits the room for action available to antitrust authorities. Antitrust procedures are not prepared to take genuinely political decisions, to weigh competing pros and cons, or to take decisions under conditions of high conceptual and factual uncertainty. Finally, once an antitrust case is decided, it becomes very difficult to adapt the solution to future changes. Hence, there is a considerable risk of ossification.<sup>10</sup>

### 7 Keeping Evolutionary Paths Open

All in all, at this point it is not convincing that peer-to-peer is preferable to Windows as a technology for organizing the markets for applications software. It is even more problematic to justify centralized intervention that aims to impose a change in the market. In any case, antitrust proceedings are not an appropriate means for bringing the change about. But it would be equally exaggerated to exclude a shift from Windows to the peer-to-peer world forever. Technical or institutional innovation may find ways to solve the many efficiency problems inherent in the production and distribution of software with equal or even greater elegance in a peer-to-peer framework. Users' preferences for the traditional solution might wane. In principle, a sound policy might therefore focus on keeping an evolutionary path for peer-to-peer open.

But this is not as easy a choice as one might think. The reason has already been mentioned repeatedly: operating systems exhibit strong positive network externalities. Central intervention striving to keep evolutionary paths open thus comes at a considerable opportunity cost. At the very least it prevents the existing network from growing. *Nota bene*, this is as much of a cost to users as it is to the producer. Often, the effect of such intervention goes even further. Since it becomes more difficult for the existing network to adapt to changed preferences or circumstances, some users will be tempted to go away. The ensuing dynamics are hard to predict. They may well result in a situation where neither the old nor the new system exploits network externalities.

From a legal perspective, another feature of the centralized intervention to keep evolutionary paths open is almost more critical. Since it is impossible to predict the dynamics, there can be no general rule. Centralized intervention aimed at securing healthy evolution is inevitably discretionary. The law in general and the courts in particular can at best see to it that antitrust authorities do not obviously surpass the outer bounds of discretion. Should those bounds be surpassed, antitrust policy will be faced with an almost tragic choice. If it insists on sound reasons in economic theory and on the rule of law, it will come close to the hands-off approach. But historical lock-ins are a true social problem. The public will be even less inclined to bear them if the evidence of bad faith is as strong as in the case of

---

<sup>10</sup> The U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001, pp. 10f.] has been aware of these limitations in the Microsoft case.

Microsoft. Occasional discretionary central intervention will therefore be hard to avoid. But a convincing theory for a policy of economic evolution is as much a *desideratum* as a constitutional framework for such intervention.

### References

- BURK, D. L., AND J. E. COHEN [2000], "Fair Use Infrastructure for Copyright Management Systems," Georgetown Public Law Research Paper #239731, Georgetown University Law Center, Washington, DC.
- FURUBOTN, E. G., AND R. RICHTER [1997], *Institutions and Economic Theory: The Contribution of the New Institutional Economics*, University of Michigan Press: Ann Arbor.
- GERKEN, L. [1995], *Competition among Institutions*, MacMillan: Houndmills.
- KATZ, M. L., AND C. SHAPIRO [1985], "Network Externalities, Competition, and Compatibility," *American Economic Review*, 75, 424–440.
- LESSIG, L. [1999], *Code and Other Laws of Cyberspace*, Basic Books: New York.
- MONOPOLKOMMISSION [1998], *Systemwettbewerb*, Nomos: Baden-Baden.
- MÜLLER, M. [2000], *Systemwettbewerb, Harmonisierung und Wettbewerbsverzerrung*, Nomos: Baden-Baden.
- PICKER, R. C. [2002], "Pursuing a Remedy in Microsoft: The Declining Need for Centralized Coordination in a Networked World," *Journal of Institutional and Theoretical Economics*, 158, 113–154.
- SALOP, S. C., AND R. C. ROMAINE [1999], "Preserving Monopoly: Economic Analysis, Legal Standards, and Microsoft," *George Mason Law Review*, 7, 617–671.
- SHELANSKI, H. A., AND J. G. SIDAK [2001], "Antitrust Divestiture in Network Industries," *University of Chicago Law Review*, 68, 1–99.
- SIDAK, J. G. [2001], "An Antitrust Rule for Software Integration," *Yale Journal on Regulation*, 18, 1–83.
- TIJONG, H. I. T. [2000], "Breaking the Spell of Regulatory Competition: Reframing the Problem of Regulatory Exit," Preprints aus der Max-Planck-Projektgruppe: Recht der Gemeinschaftsgüter Bonn 2000/13, Bonn.
- U.S. COURT OF APPEALS FOR THE DISTRICT OF COLUMBIA CIRCUIT [2001], *United States of America v. Microsoft Corporation*, June 28, <http://ecfp.cadc.uscourts.gov/MS-Docs/1720/0.pdf>.
- U.S. DISTRICT COURT FOR THE DISTRICT OF COLUMBIA CIRCUIT [1999], "Findings of Fact," 84 F. Supp. 2nd 9 (D. D. C. 1999), <http://www.dcd.uscourts.gov/microsoft-all.html>.
- [2000a], "Conclusions of Law," 87 F. Supp. 2nd 30 (D. D. C. 2000), <http://www.dcd.uscourts.gov/microsoft-all.html>.
- [2000b], "Final Judgement," 97 F. Supp. 2nd 59 (D. D. C. 2000), <http://www.dcd.uscourts.gov/microsoft-all.html>.
- WILLIAMSON, O. E. [1985], *The Economic Institutions of Capitalism: Firms, Markets, Relational Contracting*, Free Press: New York.

Christoph Engel  
 Max-Planck-Projektgruppe  
 Recht der Gemeinschaftsgüter  
 Poppelsdorfer Allee 45  
 53115 Bonn  
 Germany  
 E-mail: engel@mpp-rdg.mpg.de

## Pursuing a Remedy in Microsoft: The Declining Need for Centralized Coordination in a Networked World

*Comment*

by

ULRICH KAMECKE

### *1 Introduction*

The Internet is about to revolutionize the distribution of software. Whereas software producers used to distribute their products on inconvenient compact disks, they can now use the Internet for up-to-date interactive installation of the packages necessary to get the desired task to work. In his paper PICKER [2002] chooses this development as the central unifying principle for his analysis of the Microsoft case. He observes that the efficient organization of software sharing used to fall naturally to the *hub*, that is, to the firm that supplied the operating system. With the help of the Internet, however, it has become possible to implement availability and compatibility checks online, so that peer-to-peer coordination of software sharing is now feasible. This observation has important implications for the dominant supplier in the market, and so Picker concentrates his analysis on the implications of this development as a benchmark for the may and must-nots in the Microsoft case.

For an economist it is very attractive to discuss an antitrust case from such a unifying perspective, since it allows him to narrow the class of models that are applicable in a rule-of-reason analysis. However, this narrow view has the obvious problem that seemingly clear conclusions may be false whenever the view does not capture the essentials of the economic problem. In this comment I want to demonstrate this at two points. First, I will take a closer look at the nature of the products sold by Microsoft and its competitors to argue that tying is even less problematic than indicated in Picker's analysis. Then, I will take a closer look at the question of the relevant markets involved in this case, to demonstrate that monopoly maintenance in violation of Section 2 of the Sherman Act is indeed a justified finding in this case.

## 2 *Tying and the Public-Goods Problem*

According to the standard definition, a good is *public* if it can be used by many consumers without diminishing the amount available to each of them.<sup>1</sup> Mathematically, this requirement means that the demand of an additional customer for an amount (or quality) of a public good that is already provided can be satisfied at zero marginal cost. Economic theory has established several reasons why the efficient provision of such a public good is essentially impossible. The problem is that an efficient solution calls for perfect price discrimination, and this outcome cannot be expected from any private or public institution. Competitive outcomes are particularly inefficient in this context, as they lead to revenues that do not in general cover the costs of production, and hence to a complete market failure.

Nevertheless, it is also clear that private firms can provide public goods very profitably. Take for example the pharmaceutical industry, where the largest part of the cost of a medication is the research and development needed to introduce the product. This cost is sunk when the medication is marketed, so that the competitive price is close to zero. The public good in this case is the knowledge about the biochemical action of the medication. The corresponding public-good problem is solved with the help of a monopolistic property right on the use of the good, and society accepts the welfare losses to generate a sufficient incentive for research and development.

In the software industry the problem is even more difficult. Again we have several examples of effective protection of intellectual-property rights that allows the sale of public goods where knowledge is the major input into a product, but now there remain at least as many products for which such legal protection is ineffective. The Internet browser is one example. Of course, Netscape has the intellectual-property right to its browser, but nobody would want to deny Microsoft or any other software house the right to develop a similar product. Thus, it is not surprising that the profitable marketing of competing (almost) public goods becomes very difficult. Moreover, if firms manage to sell such products at a profit, an active antitrust authority will usually find some good reason to complain about anticompetitive conduct in the industry.

Take Adobe's Acrobat Reader as an example, discussed by PICKER ([2002, p. 136]). At the moment, this package is profitable because the Acrobat Writer becomes more valuable with every Reader installed. But what if some company enters the market with a competing Adobe-compatible "Writer"? The first natural defense against this would be an illegal attempt by Adobe to tie its two products in

---

<sup>1</sup> In public-finance textbooks it is also required that it is impossible to exclude anyone from consumption. This implies that the private provision of the good is even more difficult, so that the call for government action seems justified. However, to prove impossibility of efficient provision the no-rivalry assumption is enough, and so we take this to characterize the public-good problem in our context.

order to force full line entry. If technical or legal problems lead to a failure of this attempt, the next consequence would be a very asymmetric development effort. While fast development can be expected for the Writer, the Reader technology will not be improved correspondingly by the two competitors. Alternatively, the firms could try to sell also the advanced versions of the Reader – at the cost of lower welfare, because the price would then exceed the marginal social cost of provision of the good.

OLSON [1971] discussed tying as one possible solution to the public-good problem. In his examples tying looks much less problematic, since he talks about unions or interest groups who solve a participation problem by tying membership to the provision of a consumption good. But from an antitrust perspective this is not fundamentally different from the tying efforts implemented by Microsoft. Thus, economic theory tells us that some monopolistic exploitation with the help of tying may indeed be necessary to provide a public good privately, so that Microsoft's activities may be justified by standard static efficiency arguments alone. However, a rule-of-reason analysis would always take monopolistic exploitation as one of the reasons against the arrangement, so that this case may even call for *per se* permission for all vertical restraints involved.

### 3 Monopoly Maintenance

So, if every contractual arrangement may be necessary to organize a software market, does this not imply that the Microsoft case should be dismissed as without merit? I think that the answer is no, not because of what Microsoft did (I think that none of the strategies discussed is outrageously wrong), but because of why they did it. Picker sees the difficulty in “separating beneficial and anticompetitive designs,” and indeed, this task is as difficult as sorting out after a street fight which punches were meant to hurt the opponent and which were meant to impress the ladies. I have to admit that I am not enough of a lawyer to see why it is necessary to solve this puzzle. As far as I understand, there is clear evidence that Bill Gates himself was afraid that the Netscape Navigator could provide a next-generation platform for software development and that he implemented several measures just to fight this possibility.

The only open question at this point seems to be the market for which the monopoly was maintained. It is clear that Microsoft has a monopoly in the market for operating systems, but it is just as clear that Netscape never attempted to challenge this monopoly. So, how can there be monopoly maintenance without a corresponding potential competitor, or, as PICKER [2002, p. 130] puts it, “... we have the difficult problem of identifying potential competitors to the OS.”

In order to answer this question one has to go back to the basics of market definition. Before it is possible to determine the substitutes for a product, one has to find the needs of the opposite market side that are satisfied by the product in question. At this point, Picker's analysis is not careful enough. He accepts the obvious,

namely that an operating system serves to organize all operations on a computer, that this need is satisfied in 90% of the computers by a Microsoft product, and that none of the products involved in the case has the potential to take over this task. However, this narrow view neglects a second need satisfied by Microsoft's operating system: at the moment Windows is the platform for most developments in the market for computer applications. An operating-system monopolist can cash in a fair share of the revenues for new computers, but the platform provider can cash in a small share of the enormous money spent for the introduction of new products in this industry. Moreover, the huge amount of application software generates the network effect that protects Microsoft's monopoly for operating systems so that without the platform monopoly there would probably appear competition in the form of other operating systems.

In the sixties and seventies the development platform was the core of the hardware, and IBM was the dominant provider of this platform. The IBM antitrust case received so much attention because IBM was considered to be the central player in the computer business, a player that had controlled every new development for more than 20 years. A strategic mistake gave Microsoft the chance to take over the platform monopoly when IBM bought DOS instead of developing this "unimportant" product itself. After this experience it is not surprising that Bill Gates is afraid of doing something similarly stupid. Consequently, he tries to pay as much attention as possible to every potential new platform, and the small awareness of this profitable monopoly makes his maintenance efforts relatively easy.

#### 4 Remedies

The argument above shows that more or less every contractual arrangement or business practice may be justified in the complex software markets, so that the remedies proposed by Picker are far less convincing than they appear in the narrow framework of efficient software distribution. In the light of the difficulties of this market it is easier to follow the Chicago School and advocate *per se* legality of all vertical restraints, including tying (and resale price maintenance).

The argument on monopoly maintenance, on the other hand, implies that Microsoft indeed violated the Sherman Act and should therefore be punished. So, how should the court choose adequate remedies? In my opinion the answer to this question is much easier than one would expect after the complicated and subtle argument in Picker's paper. All of Microsoft's actions may be justified if they are implemented with the right intention, but there is sufficient evidence that Microsoft's intention was not right. It is, however, not possible to design remedies that make it impossible for a powerful firm to implement actions with such a wrong intention in the future, so the court must make sure that there is no incentive to do so. Since it is difficult to prove monopoly maintenance, the punishment in the rare case in which one is caught must be very tough. In this case the responsibility for the action as well as the proof of the intention goes back to the owner

of Microsoft, and so it seems natural to discuss imprisonment of Bill Gates instead of complicated restrictions on complicated contractual arrangements.

#### *References*

- OLSON, M. [1971], "The Logic of Collective Action: Public Goods and the Theory of Groups," Harvard University Press: Cambridge, MA.
- PICKER, R. C. [2002], "Pursuing a Remedy in Microsoft: The Declining Need for Centralized Coordination in a Networked World," *Journal of Institutional and Theoretical Economics*, 158, 113–154.

Ulrich Kamecke  
Institut für öffentliche Finanzen, Wettbewerb und Institutionen  
Humboldt-Universität zu Berlin  
Spandauer Str. 1  
10178 Berlin  
Germany  
E-mail: kamecke@wiwi.hu-berlin.de