

1 DAY CASEBEER
MADRID & BATCHELDER LLP
2 Lloyd R. Day, Jr. (90875)
James R. Batchelder (136347)
3 Robert M. Galvin (171508)
20300 Stevens Creek Boulevard, Suite 400
4 Cupertino, CA 95014
Telephone: (408) 873-0110

5 BROBECK, PHLEGER & HARRISON LLP
6 Jeffrey S. Kingston
James L. Miller
7 Spear Street Tower
One Market Street
8 San Francisco, CA 94105
Telephone: (415) 442-0900

9 CLIFFORD CHANCE ROGERS & WELLS LLP
10 Kevin J. Arquit
200 Park Avenue
11 New York, NY 10166
Telephone: (212) 878-8000

12 Attorneys for Plaintiff,
13 SUN MICROSYSTEMS, INC.

14 UNITED STATES DISTRICT COURT
15 NORTHERN DISTRICT OF CALIFORNIA
16 SAN JOSE DIVISION

17
18 SUN MICROSYSTEMS, INC.,
a Delaware corporation,

19 Plaintiff,

20 v.

21 MICROSOFT CORPORATION,
22 a Washington corporation,

23 Defendant.

No. C 02-01150 PVT

**SUPPLEMENTAL DECLARATION OF
RICHARD GREEN IN SUPPORT OF SUN
MICROSYSTEMS, INC.'S MOTION FOR
PRELIMINARY INJUNCTION**

1 I, Richard Green, declare:

2 **1.** I submit this declaration to supplement and replace my March 8, 2002 declaration,
3 in which a number of graphical and typographical errors were made. In addition, in paragraph 90
4 of my March 8, 2002 declaration, I mistakenly stated: “Sun is a member of each Executive
5 Committee [of the Java Community Process] but does not have veto power over their decisions.”
6 At the time I wrote and signed my first declaration, I understood and believed this statement to be
7 true and correct. After signing my declaration, I learned that in two narrowly defined
8 circumstances, neither of which have ever occurred, Sun does in fact have a right to disapprove
9 Executive Committee decisions. This declaration corrects that mistake and in addition provides a
10 textual description of the graphics attached to this declaration.

11 **2.** I am Vice President and General Manager of Java and XML Platforms for Sun
12 Microsystems, Inc.

13 **3.** As Vice President of Java Software Development for Sun Microsystems, Inc., I
14 manage the strategy, product development, support, and compatibility technologies for all of Sun’s
15 Java/Jini Network Technology/XML platforms. I have been with Sun for 12 years.

16 **4.** Prior to my current position, I was Vice President of the Solaris Products Group,
17 where I managed the strategy, product development, and marketing of the Solaris operating system
18 product line.

19 **5.** Also at Sun Microsystems, I managed the design and development of distributed
20 object systems, network communication products and development tools, PC integration
21 technologies, software development tools, and database systems.

22 **6.** Before joining Sun in 1989, I managed CAD/CAM design and graphics
23 engineering, and I designed simulation software for various transportation research projects.

24 **7.** I graduated from the State University of New York at Albany with a BA and an
25 MA.

26 **8.** In my capacity as Vice President of Java Software Development at Sun, and in my
27 prior experience as Vice President of Sun’s Solaris Products Group, I have personal knowledge of
28 the facts relating to the history, features, and benefits of the Java platform; the current status of the

1 Java platform; the market conditions (including Microsoft's monopoly power) that necessitate a
2 solution to break down the applications barrier to entry; the effects of Microsoft's attempts to
3 destroy the Java desktop platform; and the urgent need for and the benefits to competition of an
4 order requiring Microsoft to distribute a current, compatible implementation of the Java Runtime
5 Environment with its Windows XP operating system and Internet Explorer web browser.

6 **9.** I have personal knowledge of the facts contained in this declaration and if called as
7 a witness, I could and would testify competently thereto.

8 **I. OVERVIEW**

9
10 **10.** The Java platform is a type of middleware designed to allow software applications
11 to run on a number of different operating systems. The Java platform and its ability to undermine
12 the "applications barrier to entry" that sustains Microsoft's monopoly were addressed in the
13 District Court for the District of Columbia's Findings of Fact ("FOF"). In *United States v.*
14 *Microsoft Corp.*, 84 F. Supp. 2d 9 (D.D.C. 1999) and 87 F.Supp.2d 30 (D.D.C. 2000), *aff'd in part*
15 *and rev'd in part*, 253 F.3d 34 (D.C. Cir. 2001), the District Court for the District of Columbia
16 made relevant findings of fact and conclusions of law regarding Microsoft's monopoly in the Intel-
17 based personal computer operating system market, the threat posed to that monopoly by Sun's
18 Java platform and the anticompetitive acts of Microsoft that illegally maintained its monopoly.

19 **11.** Sun introduced the Java platform in 1995. The Java platform is a middleware
20 platform upon which applications run. The Java platform was designed to run on a number of
21 different operating systems. Thus, application developers can write Java-based applications that
22 will operate effectively with any number of different operating systems. The Java platform lowers
23 the cost of porting applications to different operating systems, and the cost of substituting
24 ("switching") one operating system for another.

25 **12.** The Java platform includes: a programming language; a set of Java APIs provided
26 in class libraries that include standard modules of preconfigured software code to ease software
27 development; a compiler, which translates Java-based source code (instructions that are written by,
28 and can be understood by, software developers) into Java-based byte code (instructions that can be

1 understood by a computer); and a Java Virtual Machine (“JVM”), which translates Java-based
2 byte-code into instructions comprehensible to the underlying operating system. The JVM and the
3 set of Java class libraries used by a Java-based application running on the JVM comprise the Java
4 Runtime Environment (“JRE”). The term JVM is sometimes used to refer to the entire Java
5 Runtime Environment. For a Java-based application to operate effectively on a particular
6 computing device, such as a desktop computer, a JRE must be installed on that computing device.

7 **13.** In order to encourage the rapid and widespread adoption of compatible JREs on as
8 many operating systems as possible, Sun licensed the source code for the Java platform to
9 hundreds of industry participants, including IBM, Hewlett Packard, and even Microsoft. It has
10 also created an industry standards body, called the Java Community Process, to shepherd the
11 continued development and enhancement of the Java platform.

12 **14.** In the U.S. government’s litigation against Microsoft, I understand that the D.C.
13 District Court found that Microsoft viewed the Java platform as a significant threat to Microsoft’s
14 monopoly in Intel-compatible personal computer operating systems. Microsoft believed that broad
15 distribution of the Java platform would prompt ever more developers to create Java-based
16 applications for the Java platform. The more Java-based applications developers produced for
17 desktop computers, the more consumers could replace or ignore Microsoft’s Windows operating
18 system, thus eroding Microsoft’s monopoly in the PC operating systems market. *U.S. v. Microsoft*,
19 *supra*, 84 F.Supp.2d at 29-30; FOF ¶¶ 75-77, 386.

20 **15.** In response to the threat posed to Microsoft’s core monopoly by the Java platform,
21 Microsoft acted to undermine both the distribution channels for compatible Java platforms, and the
22 demand for development of compatible Java-based applications. First, Microsoft undermined the
23 distribution of Netscape’s Navigator browser, the primary non-Microsoft distribution channel for
24 desktop Java runtimes. By destroying the Navigator distribution channel for Java platforms,
25 Microsoft became the dominant distribution channel for Java platforms to desktop computers, and
26 its incompatible Java implementation became the most ubiquitous JRE on desktop computers.

27 **16.** Second, Microsoft breached its license agreement with Sun by distributing an
28 incompatible implementation of the JRE with its browser and operating system products. It

1 deceived software developers about its platform and tools, and it entered into exclusive deals with
2 developers and OEMs to supplant the Java platform with an incompatible version that was
3 dependant on and tied to Microsoft’s Windows platform.

4 **17.** Consequently, insofar as developers wished to create Java-based applications for
5 desktop computers, the ubiquity of Microsoft’s JRE prompted them to create applications for its
6 incompatible, Microsoft-dependent JRE rather than the compatible Java platform. This had the
7 effect of fragmenting the installed base of Java runtimes into compatible and incompatible
8 runtimes, thus greatly reducing the size of the compatible runtime market. It also had the effect of
9 forcing distributors to incur unnecessary costs to port Java-based applications from Microsoft’s
10 JRE to any other JRE.¹ This, in turn, had the effect of reducing, if not eliminating, the incentive of
11 developers to create desktop applications for the Java platform. Not only could they no longer
12 create a single application that would execute properly on every desktop JRE, but the size of the
13 potential market for such applications was dramatically reduced by Microsoft’s breach. It also had
14 the effect of reducing the supply of Java-based applications, by requiring developers to create
15 incompatible, non-standard Java-based applications in return for early “first-wave” access to
16 Windows technical information. *See* FOF ¶¶ 388-91, 397-98, 407.

17 **18.** Although Microsoft acknowledges that millions of consumers seek to access and
18 use the Java platform daily, Microsoft still does not distribute a compatible JRE with its Windows
19 operating system products. Sun has repeatedly offered Microsoft the opportunity to distribute a
20 compatible, current JRE as part of its browser and operating system products, but Microsoft has
21
22
23
24
25
26

27
28 ¹ Microsoft’s behavior also forced end-users who might wish to switch from Microsoft’s JRE to
any other JRE to incur unnecessary costs as well.

1 refused to do so.²

2 **19.** In July 2001, Microsoft announced that it would not include any support for the
3 Java platform in its upcoming desktop operating system, Windows XP. The *Wall Street Journal*
4 reported at the time that “this favors Microsoft’s new technologies, and will inconvenience
5 consumers. . . . [i]f you want your Web page accessible to the largest number of people, you may
6 want to drop Java and switch to Microsoft’s competing set of products, which is under
7 development and is known as .NET.” With this announcement, subsequently confirmed when
8 Microsoft released Windows XP in October 2001, application developers learned that Microsoft
9 would abandon the Java platform and that there would be a reduction in the frequency with which
10 leading personal computer suppliers would pre-load *any* JRE, even an outdated one, on their
11 products.

12 **20.** The information technology industry is now watching as the final phase of
13 Microsoft’s campaign to undermine the Java platform unfolds. In June 2000 Microsoft announced
14 its plan to build and distribute an imitation of the Java platform, called .NET. Although Microsoft
15 promises that .NET will provide many of the innovative benefits of the Java platform, one very
16 significant difference remains: Microsoft refuses to port the complete .NET platform to any
17 operating system or platform other than Microsoft’s platforms, such as Windows XP or Internet
18 Explorer. In contrast, anyone can port the Java platform to any operating system. Thus, while Sun
19 licenses hundreds of industry competitors the right to use its complete source code for the Java
20 platform to make and sell competing Java products, Microsoft refuses to license or disclose the
21 complete source code for its .NET platform to anyone. The purpose and effect of the Java

22 ² In 1997, Sun sued Microsoft for breach of its Java licensing agreement based on Microsoft’s
23 distribution of incompatible Java platforms and tools. In March 1998, November 1998 and
24 again in January 2000 this Court issued orders preliminarily enjoining Microsoft from engaging
25 in certain deceptive practices and unfair competition resulting from its distribution of browser,
26 operating system and tool products that were designed by Microsoft to create and run
27 incompatible, Microsoft-dependent Java-based applications by default. *See Sun Microsystems,*
28 *Inc. v. Microsoft Corp.*, 999 F. Supp. 1301 (N.D. Cal. 1998), 21 F. Supp. 2d 1109 (N.D. Cal.
1998) and 87 F. Supp. 2d 992 (N.D. Cal. 2000). Two months prior to the expiration of the
licensing agreement between Sun and Microsoft, the parties agreed to settle certain claims. In
order to minimize disruption to Java developers and customers, Microsoft requested and Sun
agreed to grant Microsoft a limited license to continue to distribute an out-dated, four year old
JRE in certain products until 2008. The settlement agreement does not preclude Microsoft from
distributing a compatible JRE in its products.

1 platform is to facilitate and promote competition on a level playing field. The purpose of .NET
2 appears to be to stifle that competition.

3 **21.** I understand that Sun seeks an order requiring Microsoft to distribute the most
4 current, compatible implementation of the Java Runtime Environment with Microsoft's Windows
5 operating system and Internet Explorer browser. Contrary to Microsoft's contractual commitment
6 to Sun to distribute only the most current, compatible implementations of the Java platform,
7 Microsoft not only failed to do so, but it deliberately contrived to create and distribute
8 *incompatible* Java platforms while simultaneously undermining the most important alternative
9 distribution channel of compatible Java platforms to desktop computers. While the preliminary
10 injunction that Sun seeks cannot undo the years of competitive harm caused by Microsoft's illegal
11 conduct, it can help to re-create the competitive dynamics that existed prior to Microsoft's
12 unlawful acts, ensure that Microsoft will not further benefit from its established wrong-doing,
13 ensure that Sun and its licensees, developers and customers will not be irreparably harmed, and
14 that the possibility of effective, appropriate relief in this matter will not be lost pending trial of this
15 dispute.

16 **22.** By lowering and possibly eliminating the applications barrier to entry that currently
17 sustains Microsoft's monopoly, the preliminary injunction sought by Sun will foster and promote
18 competition in the market monopolized by Microsoft. By ensuring that the distribution of a
19 compatible, current JRE to desktops is at least as ubiquitous as the distribution of Microsoft's IE
20 browser and Windows operating system, the order sought by Sun will provide both application
21 developers and consumers an equal choice, and a comparable incentive, to create and use Java-
22 based desktop applications. Without the preliminary relief sought by Sun, developers and
23 consumers will effectively have no choice but to create and use desktop applications that are tied
24 to and require the use of a Microsoft browser or PC operating system. The preliminary relief
25 sought by Sun will not only provide competitive choice in applications development and use, but it
26 will reduce the switching cost and burden that consumers must otherwise incur in order to
27 substitute a non-Microsoft operating system or middleware platform for Windows. It will also
28 enhance the ability of non-Microsoft operating systems to surmount the applications barrier to

1 entry by expanding the number and variety of applications capable of running on all such
2 operating systems.

3 **II. THE NATURE OF MICROSOFT'S MONOPOLY POWER AND THE**
4 **BUSINESS DYNAMICS THAT SUSTAIN IT**

5 **23.** To understand why an order requiring Microsoft to distribute the current,
6 compatible Java Runtime Environment with Windows XP and Internet Explorer is needed to
7 preserve potential competition in the market monopolized by Microsoft pending trial, it is first
8 necessary to understand how the Java platform threatened to end Microsoft's monopoly in PC
9 operating systems, and how Microsoft's illegal acts worked to eliminate that potential competition
10 and maintain its monopoly.

11 **24.** In the U.S. government's antitrust suit against Microsoft, the D.C. District Court
12 found that Microsoft's PC operating systems monopoly is sustained by the "applications barrier to
13 entry." The applications barrier results from a combination of three factors: (1) the high "porting"
14 and "switching" costs associated with applications developed for traditional PC operating systems,
15 (2) the incentive of desktop developers to create applications for the platform with the largest
16 market share, and (3) the incentive of desktop consumers to acquire and use the operating system
17 with the largest market share. *See* FOF ¶¶ 30-32, 36.

18 **25.** On a PC or similar computer, there are traditionally two layers of software: the
19 operating system and the software programs (or "applications") that run on the operating system.
20 *See Figure 1.* An operating system is designed to function on a particular type of hardware, such
21 as an Intel-based PC or an Apple Macintosh with a PowerPC processor. An application is
22 designed to run on a particular operating system (and processor), such as Windows on Intel or
23 Macintosh OS X on PowerPC. Such operating systems and applications are thus said to be
24 "platform-specific."

25 **26.** An application interacts with the operating system through application
26 programming interfaces ("APIs") that are exposed by the operating system. FOF ¶ 2. The APIs of
27 an operating system comprise the means by which an application developer can access and use the
28 underlying functionality and services of the underlying operating system and hardware. By

1 writing the application to incorporate “calls” to the APIs exposed by a particular operating system,
2 the developer can enable the application to access and use the full set of features, functions and
3 services provided by the operating system and the hardware on which it runs.

4 **27.** Traditional operating systems, such as Windows or Macintosh, typically expose
5 different (and incompatible) sets of APIs. That is why an application developed to run on one
6 operating system typically cannot run on a different operating system without extensive, costly
7 modification to the application to incorporate calls to the different APIs of the second operating
8 system. See *Figure 2*.

9 **28.** The process of rewriting an application to use a different platform’s APIs is
10 commonly known as “porting.” Porting an application from one platform to a different platform is
11 a very costly, time-consuming process. Porting involves substantial reprogramming and testing of
12 the application code to make it compatible with a different platform’s different set of programming
13 interfaces and services.³ As shown in *Figure 3*, porting an application can typically cost anywhere
14 from 40% to 80% of the total cost of developing the original application. The greater the
15 differences between the set of APIs of the platform for which the application was originally
16 developed, and those to which the application is to be ported, the greater the cost. Where the cost
17 of porting an application is high relative to the potential market for the ported application, there is
18 generally little or no incentive to port the application. As *Figure 4* depicts, that is particularly true
19 for desktop operating systems, where the market potential of Windows applications far exceeds
20 that of any other platform. Due to the high cost of porting PC applications, and the relatively small
21 base of customers for non-Microsoft desktop operating systems, application developers have little
22 or no economic incentive to port applications developed for the Windows platform to other
23 desktop operating systems.

24 **29.** High customer “switching” costs also pose a substantial barrier to competition with
25 Microsoft’s PC operating system monopoly. For those customers who wish to switch from
26 Microsoft’s Windows operating system to an alternative operating system or platform, the cost to

27 ³ To the extent an application needs to call functions and/or services in a platform that are
28 undocumented or not fully disclosed by the platform vendor, it can be virtually impossible to port
the application to any other platform.

1 do so entails much more than simply the cost to purchase or produce new versions of its existing
2 applications for its new platform. As depicted in *Figure 5*, the cost to switch from Microsoft's
3 operating system to any other platform also entails a variety of additional costs, including the cost
4 of acquiring the new platform; the cost of re-training highly skilled staff to work with and operate
5 the new platform; the cost of transferring data from the Windows applications and system to a
6 replacement system; and the cost of incorporating the new platform into the customer's
7 infrastructure and other systems. See FOF ¶¶ 30-32. Frequently, the sum of such ancillary costs
8 far outweighs any performance, reliability or cost advantages that can be achieved by replacing
9 Microsoft's Windows operating system with an alternative platform.

10 **A. THE BUSINESS INCENTIVES CREATED BY HIGH PORTING AND SWITCHING**
11 **COSTS ALSO SUSTAIN THE APPLICATIONS BARRIER**

12 **30.** In general, the business of developing and distributing software applications for
13 desktop computers is characterized by increasing rates of return with increasing sales. In other
14 words, once the cost to create, test, de-bug and package an application for a particular platform has
15 been incurred by a developer, the cost to produce and distribute an additional copy of that
16 application for that platform is virtually zero. To the extent that the fixed cost of application
17 development is spread over an increasing number of sales, the average profit earned on all sales
18 increases with each additional copy sold. Thus, as illustrated in *Figure 6*, the greater the total
19 number of application copies sold, the lower the average cost incurred to produce each additional
20 copy, and the greater the profit per copy earned on each additional application copy sold.

21 **31.** The increasing rate of return earned on each additional application copy sold creates
22 a powerful business incentive to develop applications for the largest possible markets, that is, for
23 the platform with the largest installed base. By creating an application for the platform with the
24 largest installed base, the application developer can maximize its opportunity to sell the largest
25 number of application copies, and thus earn the highest possible return on its investment in
26 applications development. The more copies it can sell, the higher the average rate of profit earned
27 on all copies sold. In other words, because the platform with the largest installed base provides the
28 largest number of potential customers for an application, a developer can maximize the

1 opportunity to earn the largest possible return by designing its application to run on the platform
2 with the largest installed base.

3 **32.** The increased rate of return possible with each additional sale also causes
4 developers to shy away from platforms with relatively smaller installed bases. In such cases, a
5 developer will commonly compare the potential rate and amount of return it can earn on
6 applications developed for the platform with the largest installed base, with the rate and amount of
7 return it could earn on applications developed for or ported to platforms with smaller installed
8 bases.

9 **33.** Unless a computing platform has a sufficiently large installed base, that is, a
10 sufficiently widespread distribution to provide competitively attractive opportunities for economic
11 returns on application development (or unless the porting costs to a given platform are negligible),
12 third-party applications developers have little or no incentive to develop applications for any
13 platform with a relatively smaller installed base. As illustrated in *Figure 7*, the fact that Windows
14 has by far the largest installed base of all desktop operating systems means that developers
15 generally have an overwhelming business incentive to create desktop applications first, and often
16 only, for the Windows platform.

17 **B. THE FEEDBACK CYCLE AND THE APPLICATIONS BARRIER TO ENTRY**

18 **34.** The business incentive of third party application developers to create applications
19 first, and frequently only, for the computing platform with the largest installed base, coupled with
20 the high cost of porting applications from one operating system to another, have had a profound
21 effect on the commercial viability of desktop operating systems. In short, the larger the relative
22 installed base of a given platform, the greater the incentive of third parties to develop and
23 distribute applications for that platform only. And the greater the number and variety of
24 applications available in the market for a given platform, the greater the commercial appeal, and
25 thus the demand for that platform. As illustrated in *Figure 8*, these business incentives create a
26 “feedback cycle” that further entrenches the competitive position of the already dominant
27 platform.

28 **35.** Because the sales of Microsoft’s Windows platform reportedly exceed 90% of all

1 desktop computers sold, and have done so for a substantial number of years, desktop application
2 developers have little or no economic incentive to create applications for any desktop platform
3 other than Windows. *See* FOF ¶ 35. Even though the Windows operating system is not always the
4 most optimal platform for desktop applications, its overwhelmingly dominant installed base
5 enables it to provide developers the highest potential return on desktop applications. That is why it
6 also attracts the greatest amount of third party application development for desktop computers, and
7 has far more applications than any other desktop operating system. This entrenches Windows's
8 dominance because consumers want to purchase an operating system with a large variety of
9 applications, and no other operating system can match the number and variety of applications
10 developed for the Windows operating system.

11 **III. THE PARADIGM SHIFT THAT THREATENED MICROSOFT'S** 12 **MONOPOLY**

13 **36.** In the mid to late 1990's a new computing paradigm, one based on distributed
14 network computing, threatened to erode and potentially eliminate many of the incentives and
15 dynamics that sustain Microsoft's monopoly. That shift in paradigms was fueled by the advent of
16 the Internet and the World Wide Web, the web browser, and especially the Java platform. To
17 understand why Microsoft sought to destroy the Java platform and why reviving desktop
18 distribution of the Java platform would prevent Microsoft from heading off yet another threat to its
19 operating system monopoly, it is important first to understand the nature and magnitude of this
20 paradigm shift, why it threatens Microsoft, and why the Java platform figures so prominently in
21 the threat perceived by Microsoft to its Windows monopoly.

22 **A. THE DESKTOP COMPUTING PARADIGM**

23 **37.** The traditional paradigm of desktop computing exemplified by Windows entails the
24 interaction of one user with one computer through a variety of applications. As depicted in *Figure*
25 *I*, a user interacts directly with computer hardware by means of specific platform-dependent
26 applications running on a specific operating system. The computer principally serves as a tool to
27 create, format or analyze information that has been input directly by the user into his or her
28 computer. The range of possible interactions are limited to the user and the particular capabilities

1 or information provided by the applications that run on his or her specific computer.

2 **B. THE DISTRIBUTED NETWORK COMPUTING PARADIGM**

3 **38.** In stark contrast to the desktop computing paradigm, the distributed network
4 computing paradigm is based on information sharing and collaboration among different devices
5 distributed across vast public and private networks. As shown in *Figure 9*, distributed network
6 computing requires the compatible interaction of individuals with a vast array of different devices,
7 data, applications, and services. It does not attempt to replace the PC. Rather, it adds network
8 connectivity, information exchange, applications sharing and communication to desktop
9 computers, server computers, telephones, televisions, digital assistants and virtually any other
10 electronic device.

11 **39.** In this way, distributed network computing allows each user to benefit from the
12 sum of the value created by all possible interactions and communications with all compatible
13 users, devices and applications present on a network. The networked computer moves beyond an
14 isolated tool for personal productivity or amusement to one of thousands or indeed millions of
15 interacting devices in which the range of possible interactions is commensurate with the ability of
16 every device, application or service on the network to interact with and exchange data or services
17 with every other device and participant on the network.

18 **40.** The promise and power of networked interactions is captured in an equation
19 popularized by Robert Metcalfe, the inventor of Ethernet: the usefulness, or utility, of a network
20 equals the square of the number of users able to interact over the network. This principle is often
21 illustrated by analogy to the telephone network. The variety and number of possible interactions
22 over a telephone network that connects only 5 participants is many times less than the possible
23 interactions and resulting utility of a network that connects 10 participants, and far, far less than
24 that of a network that connects thousands of participants.

25 **C. THE INTERNET**

26 **41.** The internet provided the core designs and services needed for distributed network
27 computing. Initiated and funded by U.S. government research, the internet is based on an open set
28 of public standards for the exchange of data and the interaction of applications. It is built to fulfill

1 the requirements of open access and scalability. It provides the essential foundation for
2 widespread adoption and growth of network computing.

3 **D. THE WORLDWIDE WEB**

4 **42.** The worldwide web also provided pivotal contributions to the adoption of
5 distributed network computing. As with the internet, the web provided open public standards for
6 data formats (HTML), data communication (http), service access (DNS) and information
7 publishing (web servers).

8 **E. THE BROWSER**

9 **43.** The web browser introduced an efficient, effective means to navigate the web and
10 interact with any other device connected to the internet. The browser provided powerful means to
11 access and present network information in a simple to use application that rapidly accelerated the
12 number of participants who could productively participate in network computing.

13 **F. THE JAVA PLATFORM**

14 **44.** Just as the web created a standard for network data creation and access, the Java
15 platform provided a rapid, cost-effective means for the development, distribution and secure
16 execution of distributed applications across multiple devices connected to a public network.

17 **45.** In the world of distributed network computing, the overriding objective is to
18 eliminate or at least reduce the obstacles to the fullest possible set of meaningful interactions
19 between remote devices, systems, and participants. That is why the cost and complexity of
20 creating, testing and distributing a different application version for each different device on a
21 network makes the development of such networks extremely costly and cumbersome, if not
22 impossible. Because programs distributed across a public network cannot know what type of
23 devices are out there, some commonly shared means is required to enable all devices and
24 participants to fully interact with one another irrespective of their unique identity or design.

25 **46.** Prior to the Java platform, a different version of each application had to be written,
26 tested and distributed for each device on a network that used a different microprocessor and
27 operating system. *See Figure 10.* Consequently, the cost and burden of porting every application
28

1 to every different device on a network greatly increased the cost and burden of distributed
2 networked computing. When that cost and burden are combined with the need to resolve all
3 potential incompatibilities that can occur between two or more different applications running on
4 two or more different operating systems and devices, the barriers to efficient distributed computing
5 across public networks were prohibitive.

6 **47.** The Java platform was designed to make distributed computing on public networks
7 feasible and attractive by eliminating the need to create and distribute different versions of
8 application programs for different operating systems and devices. Instead, as illustrated in *Figure*
9 *11*, the Java platform eliminates the need to port every application to every device by instead
10 porting a single application – the Java Runtime Environment – to each device on which the
11 application will run.

12 **48.** As illustrated in *Figure 12*, the JRE is an intermediate “middleware” layer of
13 software that acts much like a virtual computer,⁴ in that it exposes its own set of application
14 programming interfaces (the “Java Class Libraries”) that are separate from and independent of the
15 APIs of the particular operating system or device on which the JRE may be installed, and it
16 provides its own runtime environment and services (the “Java Virtual Machine”) for applications
17 that call the Java APIs. The Java Virtual Machine (“JVM”), interprets a Java-based application,
18 then directs the underlying operating system and hardware platform on which the JVM runs to
19 perform the functions needed to execute the application.⁵ Consequently, applications developed
20 for the Java platform differ from traditional, platform-dependent software in that Java-based
21 applications need not interact directly with the specific operating system or hardware of a given
22 computer.

23 **49.** By porting and distributing a JRE to multiple devices on a network, as illustrated in
24 *Figure 13*, it is possible to develop a single application that can be distributed for and run on all
25

26 ⁴ Hence the name “Java Virtual Machine.”

27 ⁵ The JVM is also designed to protect the computer on which it is installed, as well as any other
28 applications on that computer, from security risks or program errors by isolating the Java-based
application from the host system in a “sandbox” created by the JVM. This “sandbox” security
model prevents the transmission of viruses or other security risks across the network to or from the
host computer.

1 such devices, irrespective of the differences that may exist between the underlying operating
2 systems or hardware on the respective devices to which the application is distributed. Software
3 developers are thus freed from the need to develop, compile and debug a different version of each
4 application for each different operating system or networked device on which the application may
5 run.

6 **50.** The benefits of the Java platform are not limited simply to devices connected to a
7 network. As *Figure 14* illustrates, it has also been ported to and installed on multiple desktop
8 operating systems and/or browser products. In this regard, a major benefit of the platform-
9 independent design of the Java platform is the ability it affords platform and system vendors to add
10 the value of thousands or even millions of Java-based applications to their platform by porting
11 only one application – the JRE – to their system. Once ported to a specific operating system or
12 device, the JRE enables that operating system or device to execute any Java-based application, and
13 to interoperate fully with every other operating system or device that supports a compatible JRE.
14 *See Figure 15.*

15 **51.** As illustrated in *Figure 16*, another benefit of the Java platform’s innovative design
16 is the competitive choice it affords developers. So long as an operating system supports a JRE,
17 developers are free to choose whether to use the platform’s specific APIs, or alternatively to
18 reduce porting costs and increase the applications’ total market potential by developing a Java-
19 based application to run on the system’s JRE.

20 **52.** Yet another benefit of the Java platform is the business incentive it provides
21 application developers to create applications that are not tied to a particular operating system, such
22 as Windows. Because a single version of a Java-based application can run with little or no porting
23 cost on multiple devices, the size of the potential market for each such application is the sum of the
24 installed bases of all devices that have a compatible JRE. In other words, the size of the potential
25 market for each such application is larger than the potential market for applications written to
26 device-specific APIs. If, for example, the Java platform were distributed on both Windows and the
27 Macintosh, the size of the market for Java-based applications would be the sum of the installed
28 bases of Windows and Macintosh, whereas the size of the potential market for a Windows

1 application would be the installed base of Windows devices only. The opportunity to create a
2 single application that can be distributed to the installed bases of different operating systems and
3 devices with little or no porting costs is an enormously attractive commercial opportunity for
4 developers, one that rivals if not surpasses the business opportunity to develop applications that
5 can be distributed for the installed base of Windows platforms.

6 **53.** To fully achieve the cross-platform objective of the Java platform, the JRE installed
7 on a particular system or device must be fully compatible with every other JRE. That is, it must
8 implement and conform to the Java compatibility standards established by the Java Community
9 Process.

10 **54.** The Java platform provides several benefits when compared with traditional
11 platform-specific software:

- 12 a. It reduces the cost of “porting” applications, because it is the Java platform itself
13 (not the applications written to it) that is ported to multiple operating systems.
14 Thus, developers can support many platforms with substantially less effort and cost
15 than is necessary when writing platform-specific code.
- 16 b. Because the Java platform enables developers to create a single version of an
17 application that can run on different JVMs installed on different platforms, it
18 provides consumers greater choice in applications, operating systems, and
19 hardware. It has the potential not only to free individual consumers from concern
20 about whether the software they want to run is supported by a given operating
21 system, but also to permit corporations and Internet users more easily to mix
22 different types of computing systems across a network.
- 23 c. As more Java-based programs are developed, distributed and used, non-Microsoft
24 operating systems can become increasingly competitive so long as there is a JVM is
25 supported by those operating systems. This increases the ability of new operating
26 systems and hardware platforms to compete in markets previously dominated by a
27 particular vendor.
- 28 d. The Java platform allows different types of computing devices to operate the same
software program and to more easily to share software and data, which is especially
important in networks with different computing devices on them.

55. The Java platform is attractive to software developers in part because it is
specifically designed and ideally suited for the creation, distribution and secure execution of
applications for distributed network computing. The distributed network computing paradigm is
based on a small but very important set of design principles that, when properly implemented,
fulfill the promise of distributed network computing.

1 **56.** One important design principle is commonly referred to as “opaque access.” In
2 short, a device must be able to interact with another device, operating system or application
3 without any need to know the specific details of the device, operating system or application with
4 which it is interacting. Any design that requires knowledge of the implementation details of the
5 device, application, or service with which it seeks to interact (*e.g.* operating system requirements
6 or data formats) unduly taxes each participant in the network and inhibits the growth of and
7 participation in the network.

8 **57.** Another important design principle is the need to protect the security of each user’s
9 information and identity. This ensures that users will be able to carry out the greatest variety of
10 interactions using networked computing services.

11 **58.** Standardization is another important principle of distributed network computing.
12 Since the implementation details of each computer type, operating system, application and service
13 participating in the network must be opaque and secure, participants in the network must rely on
14 an accepted set of standards to structure and guide their interactions.

15 **59.** Another important principle is the need to provide efficient and consistent access to
16 information and applications across the network. Ensuring that networked computers have no
17 technical barriers to effective interaction or information exchange allows maximum participation
18 of users on a network. It also expands the total number of users and the variety of interactions that
19 can participate in the network.

20 **60.** Information and application sharing is another important principle of distributed
21 network computing. Data or applications that cannot be shared because they have to be
22 reformatted or regenerated (ported) impede the use and expansion of network computing.

23 **61.** Another important principle is scalability, that is, the ability to expand processing
24 services commensurate with the level of demand for such services. Since there is limited ability to
25 plan or control access to applications or services at any given point in time, computer systems and
26 applications must be able immediately to scale in order to meet unanticipated usage patterns and to
27 prevent interruptions in service.

28 **62.** Finally, reliability is important. Because information is being shared, applications

1 are being delivered to remote devices on the network, and applications are communicating or
2 interacting with other applications across the network, the level of reliability must be unerringly
3 high. Networks and networked applications should be designed so that a single, unseen failure by
4 any device or application will not cripple the network.

5 **63.** Distributed network computing has given birth to new categories of applications
6 that utilize the design principles of network computing and the resources of computer networks,
7 including the internet, in novel ways. Such applications include distributed applications, dynamic
8 web services, applets, and peer-to-peer applications, and currently comprise the fastest growing,
9 most innovative segment of software development.

10 **64.** Distributed applications are commonly designed to utilize the processing resources
11 of a network efficiently by dividing the work of processing a given application among various
12 devices connected to the network. Thus, as illustrated in *Figure 17*, one component of a
13 distributed application may be designed to run on a cellular telephone, while the other components
14 of the application may be designed to run on a desktop computer and one or more server
15 computers. To do so effectively, however, the developer must necessarily treat the set of devices
16 on which the various components of a distributed application are designed to run as though they
17 comprise a single processing environment, such that each device can access and support the
18 appropriate set of functionality of each other device on which the application will run. Only
19 insofar as each device can access and support the necessary set of functionality of each other
20 device on which the application will run, can developers take full advantage of the sum of the
21 features and functions of the different devices on which the distributed application is to be run.

22 **65.** For example, a networked transaction might involve interactions between a wireless
23 device (*e.g.*, cell phone), a desktop computer and one or more server computers. If the phone
24 cannot access the full set of functionality and support provided by the server, the ability to develop
25 a useful distributed application between the wireless device and the server is necessarily limited to
26 the set of interactions the device can access and support. Needless to say, the larger the number
27 and variety of potential interactions that can be accessed and supported between the phone and the
28 server, the greater the potential for developing useful applications among such devices.

1 **66.** Distributed applications require compatible runtime support on each device on
2 which the application or any portion of the application may run. Such support is not effectively
3 provided by conventional desktop operating systems.

4 **67.** Distributed computing runtimes must be portable and scalable, in order to provide
5 the ability to execute a given application on many different computing platforms comprised of
6 different operating systems and different hardware. Such runtimes must also provide an
7 environment that safely executes new downloaded code and enables applications to provide
8 services to remote devices or applications that request such services. Such runtimes must also
9 make it possible for services to scale with demand, and these runtimes need to be available on a
10 wide variety of devices – from cell phones to corporate servers.

11 **68.** As illustrated in *Figure 18*, the Java platform was specifically designed to meet the
12 needs of distributed network computing. For example, the Java platform makes it easy to
13 construct programs that can run with little or no change on different operating systems and
14 computing hardware. The JRE also provides a secure environment for downloading code across
15 the network, and it includes security features that permit such downloaded code to be executed
16 safely. The strongly typed design of the Java language precludes many bugs from compiled code,
17 and its object oriented design permits rapid application development. In addition, the class-based
18 architecture of the Java platform provides JREs for all types and sizes of computing devices.
19 Finally, the server facilities of the Java class libraries are designed to scale easily.

20 **69.** By comparison, conventional desktop operating systems such as Windows fail to
21 address many of these needs. Microsoft attempted to extend the Windows operating system to
22 provide support for networked applications by developing its ActiveX framework. However,
23 ActiveX does not meet many of the requirements for distributed network computing. In particular,
24 Windows-based ActiveX programs will only run atop the Windows operating system on Intel-
25 compatible PCs. In addition, although ActiveX controls can be downloaded across the network,
26 such programs cannot be executed safely, because Windows does not provide a secure
27 environment for executing downloaded code. Moreover, ActiveX programs are as difficult to
28 develop, if not more so, than conventional Windows programs, and ActiveX is not designed for

1 server applications.

2 **IV. THE POTENTIAL BENEFITS TO COMPETITION PROVIDED BY THE**
3 **JAVA PLATFORM**

4 **70.** The Java platform helps to erode the applications barrier to entry for desktop
5 operating systems by eliminating or steeply reducing the costs of porting PC applications to
6 operating systems other than Windows. Because the Java platform is delivered on virtually all
7 desktop operating systems except Windows XP, a developer who uses the Java platform to create a
8 desktop application will incur much lower costs to port a Java-based application from one
9 operating system to another. The reduction in porting costs greatly increases the economic
10 incentive of developers to develop and distribute applications that can run on multiple desktop
11 operating systems. The increased availability of applications for operating systems other than
12 Windows increases the commercial appeal, and thus the competitiveness, of all those operating
13 systems whose sales would otherwise be insufficiently large to attract large numbers of application
14 developers.

15 **71.** Because the Java platform provides developers the ability to build a wide range of
16 applications that can run on a broad variety of operating systems, with little or no porting costs, it
17 has the ability to expand greatly the number and variety of applications available for all desktop
18 operating systems, not just Windows. And because the Java platform dramatically reduces the cost
19 to consumers of substituting a different, non-Microsoft operating system for Windows, it has the
20 ability to increase competition in the market for PC operating systems. In other words, the Java
21 platform goes a long way toward solving the preeminent problems posed by Microsoft's PC
22 monopoly: it enables developers to write applications that can run on all platforms; and it provides
23 end users the ability to switch operating systems without having to purchase new applications or
24 re-train their staffs.

25 **72.** Because a Java-based application can be distributed with little or no additional cost
26 for use with every operating system that supports the Java platform, the total size of the market
27 opportunity available to Java developers grows to equal the sum of all Java-enabled operating
28 systems. Thus, Java developers can achieve a significantly enhanced return on their economic

1 investment in applications development, one that rivals and potentially exceeds the returns
2 otherwise available to developers who create applications that only run on Windows.

3 **73.** For those consumers of PC computers who deploy and use systems that include the
4 Java platform, the Java platform greatly reduces the cost to switch from the Windows operating
5 system to any other operating system, and greatly increases the range of competitive choices
6 available to such consumers for their operating system needs.

7 **74.** For those consumers who deploy and use the Java platform for their application
8 needs, the cost of switching operating systems is effectively reduced to the cost of the replacement
9 operating system alone. There is virtually no need to replace applications, transfer data from prior
10 applications to replacement applications, re-train staff, or unravel interoperability problems.
11 Consequently, consumers are freed to choose among competing operating systems based upon
12 their respective performance, reliability, security or cost characteristics, without concern for the
13 high cost of switching applications, data and highly skilled personnel from one vendor's system to
14 another vendor's system.

15 **75.** Today, more and more applications are being developed either as server based
16 applications, or as distributed applications in which the execution of the application is performed
17 partly on one computer, and partly on another, networked computer. These applications are run
18 via networks and the Web. This means that a platform environment must run not only on PC
19 clients, but also on servers and on other client-devices such as handhelds and cellular telephones.
20 This has profound implications for the development of complex applications for small devices
21 such as handhelds and cell phones, and also for the development of Web-based programs to run on
22 desktop computers.

23 **76.** The rise of distributed network computing, in which developers decide what portion
24 of an application will reside or run on a client or desktop device, and what portion will reside or
25 run on a networked server, has greatly increased the demand for Java-based applications on
26 servers.

27 **77.** Developers can create applications that are stored on Web servers, but can be
28 completely or partially loaded onto a desktop computer that is connected to the internet. Once

1 loaded, the code runs on the desktop computer rather than the Web server, which means that the
2 processing and other functions are performed locally on the desktop computer, without the need
3 for time-consuming transmission back and forth to the Web server. This feature is especially
4 attractive to application developers seeking to provide interactive Web functionality to desktop
5 computer users.

6 **78.** Alternatively, as illustrated in *Figure 19*, developers can create distributed
7 applications in which different components of an application are designed to run on different
8 devices, yet still interoperate with one another to increase the efficiency and/or the utility of the
9 network and the set of devices on which the application runs.

10 **79.** Many Java-based applications execute business logic on desktops, servers and other
11 networked devices. This offers significant advantages over applications that execute business
12 logic only on a single computer. For example, an e-commerce application that merely displays
13 goods for sale on a website might execute business logic only on the server. However, an e-
14 commerce application that allows the consumer to view the goods from different angles or in more
15 detail, thereby promoting increased sales, might execute business logic on both the desktop and the
16 server. For an advanced Java-based distributed application to function optimally, a current JRE
17 must be present on both the desktop computer and the server computer.

18 **80.** So long as a compatible JRE is installed on PCs, the Java platform's distributed
19 network computing paradigm potentially threatens Microsoft's continuing monopoly over PC
20 operating systems by providing a means for non-Microsoft vendors to deliver functionality and
21 services to Windows PCs that rivals or competes with the functionality and services delivered by
22 Windows. The presence of a JRE installed on the desktop provides Java developers the ability to
23 create and distribute the full range of distributed applications for desktop computers.

24 **81.** The competitive challenge posed by the Java platform was particularly threatening
25 to Microsoft due to the rapidly increasing demand for distributed network computing.

26 **82.** The threat posed by the Java platform was particularly dangerous to Microsoft
27 because the Java platform – unlike Windows or any other Microsoft platform – was specifically
28 designed for distributed network computing, and is far better suited to meet the fastest growing

1 demands of developers and consumers alike. And that is why Microsoft first attempted to
2 appropriate the Java platform. When this Court halted that attempt, Microsoft instead acted to
3 impede the commercial success of the Java platform and forestall its adoption while it struggled to
4 build an imitation of the platform tied to the Windows platform: .NET.

5 **V. THE BENEFITS TO COMPETITION OF SUN'S LICENSING AND**
6 **DISTRIBUTION MODEL FOR THE JAVA PLATFORM**

7 **83.** By reducing porting costs for developers and switching costs for end users, the Java
8 platform frees developers and consumers to select an operating system based on the system's
9 features and capabilities, rather than the number of machines that support it, or the number of
10 applications that run on it. Because Sun has licensed hundreds of industry participants to make
11 and distribute the Java platform, the widespread distribution and use of the Java platform creates
12 an application-development market not only for Sun, but for its operating system competitors as
13 well.

14 **84.** In furtherance of this objective, Sun developed full functional specifications for the
15 Java platform and published them under a license that authorizes anyone to implement the
16 technology in a compatible form, without fee. In addition, Sun developed reference
17 implementations of the Java platform for three existing operating systems: Windows, Solaris and
18 Macintosh. A binary code (machine readable only) version of each reference implementation was
19 made freely available by Sun to anyone who wished to use or distribute it without modification.

20 **85.** In addition, Sun made the source code (human-readable and modifiable version of
21 the software) for its Java reference implementations available for license by competing systems
22 vendors, tools developers, browser vendors, application developers, indeed, anyone willing to
23 agree that each implementation they distributed would be compatible and thus fully interoperable
24 with every other licensee's implementation of the platform. In this way, Sun's licensing program
25 preserves the cross-platform character of the Java development and runtime environment, thereby
26 reducing porting and switching costs among competing implementations to an absolute minimum.
27 Licensees were encouraged to port the Java platform to all other operating systems and to optimize
28 the performance of the Java platform on each operating system to which they ported it.

1 **86.** Under the terms of Sun’s standard Technology License and Distribution
2 Agreement, anyone may license the right to use Sun’s Java source code to sell implementations of
3 the Java platform, so long as the products they distribute pass Sun’s compatibility tests suites. As I
4 describe above, this ensures that each licensee’s implementation is fully interoperable with all
5 other licensees’ implementations, and that the cross-platform character of the Java platform is
6 preserved.

7 **87.** The list of industry participants who have licensed the Java source code quickly
8 included virtually all of the major computer companies, including IBM, HP, Compaq, Apple,
9 Microsoft, Fujitsu, Toshiba, Hitachi, SGI, and DEC, and each has ported a version of the Java
10 platform to its machines. For example, IBM created an implementation of the Java platform for its
11 AIX operating system, HP for its HP-UX operating system, and Microsoft for Windows. As
12 reflected in *Figure 20*, Sun’s licensing model has enabled its rivals to make and distribute highly
13 competitive yet compatible versions of the latest, most advanced Java platforms.

14 **88.** The Java platform was initially released with the basic platform functionality
15 needed by software developers. With each subsequent release of a new version of the platform,
16 additional functionality and performance has been introduced into the platform. Each source code
17 licensee receives a new release of the Java technology at the same time, and then begins its efforts
18 to port and tune its implementation for its operating system.

19 **89.** Although the Java platform was initially created by Sun, it has developed into a
20 platform that evolves and develops through the contribution of a large number of software
21 developers under an industry standards group known as the Java Community Process. The Java
22 Community Process presides over the evolution of the Java platform, and establishes rules for the
23 acceptance of proposed modifications to the standard. It consists of two Executive Committees
24 with sixteen industry members each, and hundreds of member companies who cooperate to form
25 working groups to improve or refine portions of the platform. Working groups in turn develop and
26 revise Java technology specifications, reference implementations, and compatibility test suites to
27 preserve compatibility among all implementations of the Java specifications.

28 **90.** Under the Java Community Process, the creator(s) of any new specification,

1 reference implementation and test suite retains whatever intellectual property rights may arise from
2 its creation, subject, however, to their agreement to license third parties the right in perpetuity to
3 use all such intellectual property to make and distribute compatible implementations of such
4 specification(s) on reasonable, nondiscriminatory terms. Consequently, Sun is no longer the sole
5 holder of intellectual property rights in the Java platform, since entities that have created or
6 modified a Java specification approved by the Java Community Process also own intellectual
7 property rights in the Java platform. Sun does retain ownership of the JAVA[®] trademark, but
8 freely licenses that trademark to any Java licensee who creates a compatible implementation in
9 accordance with the compatibility standards set by the Java Community Process.

10 **91.** The Executive Committee has the final vote on whether to accept new or modified
11 specifications as a part of the Java platform. Sun is a member of each Executive Committee but
12 does not have veto power over most of their decisions. In fact, certain specifications have been
13 submitted to and accepted by the Java Community Process for inclusion in the Java platform over
14 Sun's objection.

15 **92.** In two very limited circumstances, the governing document controlling the JCP
16 process provides: "EC [Executive Committee] ballots to approve UJSRs for new platform Edition
17 Specifications or UJSRs for J2SE that propose changes to the Java language, are approved if (a) at
18 least a two-thirds majority of the votes cast are 'yes' votes, (b) a minimum of 5 'yes' votes are
19 cast, and (c) Sun casts one of the 'yes' votes. Ballots are otherwise rejected." *See*
20 <http://www.jcp.org/procedures/jcp2/index.en.jsp>.

21 **93.** The first instance requiring approval by a super-majority of the Executive
22 Committee and Sun concerns proposals to define or revise an entirely new platform ("UJSR").
23 The second instance concerns any UJSR to make a Java language change in the Java2 Standard
24 Edition platform. In my experience, proposals for such changes rarely occur. To date, nearly 200
25 different proposals for changes to the Java platform have been submitted to the JCP. *See*
26 <http://www.jcp.org/jsr/all/index.en.jsp>. Of the 200 proposals submitted, I understand that less than
27 10 proposed changes in the Java language, and none of those have been disapproved by Sun. I
28 further understand that there have been less than 5 UJSRs for Platform Edition Specifications, and

1 that Sun has not disapproved any.

2 **94.** Sun offered to nominate Microsoft to become a member of the Executive
3 Committee of the Java Community Process, but Microsoft refused to join.

4 **VI. THE HARM TO COMPETITION AND TO THE JAVA PLATFORM**
5 **CAUSED BY MICROSOFT'S ANTICOMPETITIVE ACTS**

6 **95.** If Microsoft could prevent the Java platform from interoperating properly with
7 Windows, developers would be forced to write solely to Windows, because they generally cannot
8 afford to write to more than one platform, and Windows accounts for a far greater percentage of
9 end-user machines than any other platform.

10 **96.** As I discuss below, Microsoft breached its license agreement with Sun by
11 distributing incompatible Java-based products to maintain its hold on the market. It also misled
12 Java developers by misrepresenting its Java-based tools and runtime products to be compatible
13 with the Java platform when in reality the developer tools distributed by Microsoft produced by
14 default incompatible programs that were tied to Microsoft's incompatible JRE.

15 **97.** The ultimate success of the Java platform is dependent on broad distribution of
16 compatible JREs that conform to the specifications for the Java platform. Without a large base of
17 users able to run Java programs, there will be little incentive for developers to create Java-based
18 applications for the desktop; without Java-based applications for the desktop, the Java platform
19 will be unattractive to desktop users.

20 **98.** In 1995, Netscape Communications Corporation ("Netscape") became one of the
21 first source code licensees for the Java platform, and the first browser manufacturer to distribute a
22 JRE in its browser.

23 **99.** The Netscape license was of great significance to the success of the Java platform
24 because Netscape agreed to include the Java platform in every copy of its Navigator browser
25 product, thereby affording the Java platform a distribution channel that commanded a greater than
26 80% market share of internet-connected PCs. Shortly thereafter Microsoft entered into a source
27 code license for inclusion of the Java platform in its Internet Explorer browser, thereby providing a
28 second powerful distribution channel through which the Java technology was able to gain access

1 onto virtually all remaining internet-connected PCs. When Sun subsequently licensed Microsoft in
2 1996 to distribute the Java platform with Internet Explorer and Windows, Sun had secured
3 distribution of the Java platform to virtually 100% of Web-enabled PCs.

4 **100.** At the time Netscape licensed the right to distribute the Java platform in Navigator,
5 Netscape was the highest volume distributor of Java platforms for desktop computers. Because
6 Netscape co-packaged the Java platform in every copy of Navigator, the market penetration of the
7 Java platform was equal to that of Navigator. *See Figure 21.*

8 **101.** In order to compete with Navigator, Microsoft used the same packaging strategy for
9 Internet Explorer: it bundled the Java platform with Internet Explorer, thus ensuring that the
10 installed base of Java platforms on desktop computers would equal the overlapping sum of
11 Navigator's and IE's combined distribution. *See Figure 22.* However, once Microsoft breached
12 its license agreement with Sun by distributing its incompatible Java runtime and tools, the market
13 for Java-based applications fragmented into compatible and incompatible markets, and the benefits
14 of low porting and low switching costs no longer extended to the combined distribution of
15 Navigator and IE.

16 **102.** While the market share of IE was initially much less than that of Navigator,
17 Microsoft's anticompetitive acts soon caused its share of the market to surpass and ultimately
18 eclipse that of Navigator, thus growing the share of Microsoft-dependent Java platforms as well.
19 (FOF 375) By eliminating any competitive challenge from Navigator, Microsoft also
20 simultaneously eliminated the most important non-Microsoft channel for desktop distribution of
21 the Java platform.

22 **103.** Once Microsoft destroyed the first and most important channel for desktop
23 distribution of the Java platform – Netscape's Navigator browser – Microsoft's resulting control
24 over the distribution channels to desktop computers rendered Microsoft the only large volume
25 distributor of the Java platform for desktop computers. *See Figure 23.* Absent any meaningful
26 competition for the distribution of Java platform to desktop computers, Microsoft is now abusing
27 its monopoly to end the ability of the Java platform to compete for desktop application
28 development and use by excluding the platform from Windows XP. *See Figure 24.*

1 **104.** By destroying Navigator's distribution,⁶ and distributing an incompatible Java
2 platform in Microsoft's products, Microsoft prevented the Java platform from gaining the desktop
3 market presence and ubiquity it needed to succeed. While many others have licensed the Java
4 platform since that time, the overwhelming majority of JREs that have been distributed to end
5 users to date have been distributed with Internet browsers, specifically Netscape's
6 Navigator/Communicator and Microsoft's Internet Explorer. No other distribution mechanism for
7 JVMs has reached even a small percentage of the distribution achieved through the distribution of
8 Internet browsers.

9 **105.** Other means of distributing JREs have not been nearly as numerically significant.
10 It is not feasible in my experience to achieve the installed base necessary to attract competitive
11 application development by distributing a JRE for download over the internet. Even a compressed
12 JRE with associated class libraries typically ranges from 5 to 12 megabytes in size. Downloading
13 a JRE of that size would unacceptably slow for transmission of the program at speeds available to
14 most end users today on the Internet. For example, with a typical dial-in Internet connection, it
15 would take from approximately 15 minutes to an hour or longer to download a program that also
16 distributed such a JRE. For this reason, many developers allow their Java-based applications to be
17 downloaded, but do not include the JRE itself as part of the download. Indeed, current popular
18 Java-based programs that are downloaded over the Internet without also distributing a JRE include
19 Yahoo Chat, Quicken Quote Tracker, Sabre's QIK-ACCESS airline reservation software, and
20 NASA's JTRACK satellite tracking system.

21 **106.** Nor is it feasible to depend on OEMs to distribute JREs. The number of OEMs
22 required to achieve distribution comparable to Windows (which is distributed on more than 90% of
23 all desktop computers) makes its prohibitively expensive and ultimately futile to contact each
24 distributor individually. While the top five OEMs account for perhaps 60-70% of all PCs
25 distributed, thousands of OEMs account for the remaining 30-40% of the market. Even if the Java
26 platform could achieve distribution on 70% of PCs, developers would still have a stronger
27 incentive to develop applications for Microsoft products, which have over 90% distribution.

28 _____
⁶ See *U.S. v. Microsoft*, FOF Part V.

1 **107.** In addition, the OEM's economic interests are tied to the success of Windows.
2 Since a compatible JRE would enable Apple's Macintosh and other non-Microsoft desktop
3 computers to run the same Java-based applications as a Windows PC, it will increase competition
4 with PCs manufactured by the OEMs. Moreover, based on my experiences with OEMs, I believe
5 that the OEMs fear retaliation from Microsoft if they distribute a competing platform. Since
6 Microsoft's Internet Explorer browser has the largest install base of browser products, many Java-
7 based desktop applications have been written for (and run only on) Microsoft's outdated,
8 incompatible JRE. Consequently, there is less incentive for OEMs to distribute a compatible JRE.

9 **108.** There are no other economically feasible distribution channels for the Java platform
10 that would achieve anything close to the levels of distribution currently enjoyed by Microsoft
11 platforms such as Windows.

12 **109.** In short, Microsoft has either destroyed or shut down all of the most efficient and
13 effective desktop distribution channels for the Java platform. Its exclusion of the JRE from
14 Windows XP could be the final nail in the coffin.

15 **110.** Microsoft cannot justify its decision to exclude the Java platform from Windows
16 XP on the bases of cost. Microsoft has a fully paid-up license to distribute its existing JRE for 7
17 years, and could distribute the binary version of Sun's most current, compatible JRE for free.

18 **111.** Nor can Microsoft justify its decision on the basis of customer demand. The
19 decisions announced by OEMs to independently install a copy of the Microsoft VM on the PCs
20 that they make and sell indicates that there is ample consumer demand for the Java runtime.

21 **A. MICROSOFT'S ANTICOMPETITIVE ACTS WERE DESIGNED TO UNDERMINE**
22 **THE ECONOMIC INCENTIVE OF DEVELOPERS TO DEVELOP AND**
23 **DISTRIBUTE JAVA-BASED APPLICATIONS FOR THE PC**

24 **112.** By breaching its license agreement with Sun to distribute an incompatible version
25 of the Java platform in Windows, in IE and in its tools products, Microsoft signaled the
26 development community that the standard Java platform would not achieve ubiquity, and that it
27 would not be possible to distribute the same Java-based application for all desktop platforms.
28 This, in turn, greatly reduced the incentive to develop Java-based desktop applications.

113. The key parts of the Java platform – the programming language, the class libraries

1 and APIs, the compiler, and the JRE – were each altered in Microsoft’s implementation in ways
2 that impaired and undermined the commercial appeal of the Java platform.

3 **114.** Microsoft also undertook to deceive Java developers so they would develop
4 applications for Microsoft’s incompatible JRE to the exclusion of the compatible Java platform by
5 misleading developers to believe that Microsoft’s Java-based tools and runtime products were
6 compatible with the Java platform, when in fact they were not.

7 **115.** In addition, Microsoft entered into a succession of “first-wave” agreements in
8 which Microsoft required third party software developers to develop applications exclusively using
9 Microsoft’s incompatible version of the Java platform in order to receive preferential access to
10 Microsoft’s technical information. These agreements served to further fragment the Java platform
11 and prevent the creation of compatible Java-based applications.

12 **116.** By breaking compatibility, Microsoft undermined the promise of the Java platform
13 to reduce porting and switching costs. Microsoft’s changes precluded the execution of
14 applications built using Microsoft’s tools on other Java platforms, and in so doing precluded the
15 pooling of Java-based application content across multiple platforms.

16 **117.** To reduce the commercial appeal of the compatible Java platform, Microsoft also
17 coerced Intel to stop development of improved class libraries for the Java platform by threatening
18 to withdraw its support for Intel’s hardware.

19 **118.** As discussed above, Microsoft performed a variety of actions to delay and impede
20 the distribution and success of the Java platform. Microsoft has used this delay to its advantage by
21 using the time gained to develop its own product that is comparable to and competitive with the
22 Java platform. The .NET Framework copies many of the principles and features of the Java
23 platform, with one important difference – it runs only on Windows, thereby maintaining the
24 applications barrier to entry.

25 **119.** For Java-based applications to run properly on a user’s computer, there must be a
26 compatible JRE installed on that computer. Each computer with a compatible JRE is in effect
27 another potential customer of Java programs. By limiting the number of potential customers for
28 compatible Java programs the ultimate effect of Microsoft’s actions was to destroy the economic

1 incentive for ISVs to create applications for compatible Java platforms.

2 **120.** Microsoft undermined the compatibility of the Java platform, prevented the
3 distribution of compatible Java-based applications, and worked to stop progress on improved
4 functionality for the platform. Its actions both harmed and delayed the wide scale deployment and
5 adoption of the Java platform for desktop computers. Microsoft used this delay to build and
6 launch the Common-Language Runtime (“CLR”), which imitates the Java platform and runs only
7 on Windows. Microsoft has thus severely damaged Java’s ability to be a successful cross-platform
8 development environment, while mimicking Java’s best features in Microsoft’s own CLR, which
9 will steer developers to write only to Windows.

10 **121.** The Java platform is successful everywhere Microsoft does not exercise monopoly
11 power – that is, it enjoys wide distribution and a robust developer community in every major
12 market other than the PC market.

13 **122.** The Java platform is exceptionally important for software developers because it
14 enables software developers to learn one language and use it to write programs that could run on
15 any platform.

16 **123.** The Java platform is widely distributed with server operating systems other than
17 Windows XP’s server operating system, and it is a very successful development platform for
18 developers writing server-based applications, including enterprise applications.

19 **124.** The Java platform is a leading platform technology for application servers and
20 server-based dynamic Web content.

21 **125.** The Java platform also now has a high percentage of the market for so-called
22 “smart phones” – cellular telephones that can run Web server-based applications. This year alone,
23 roughly 100 to 125 million Java-powered cell phones will be distributed.

24 **126.** The Java platform is being deployed in many interactive television set-top boxes.

25 **127.** A micro-version of the Java Virtual Machine is being burned onto so-called “smart
26 cards” – credit cards that can interact with computers using a microchip embedded in the card.

27 **128.** The Java platform has the second-highest number of skilled developers writing to it
28 (2½ to 3 million, as compared to the 6-7 million who write to Microsoft’s Visual Basic). Further,

1 56% of all developers use the Java language in at least some part of their work. Although most
2 developers who write Java-based applications tend to write server-based applications, a developer
3 who can write to server-based Java can also write to PC-based Java, because the same language
4 can produce programs compatible with any platform.

5 **129.** The features of the Java platform, which I discussed above, combined with its
6 “write once, run anywhere” paradigm, make the Java platform an incredibly attractive platform for
7 software development.

8 **130.** In short, the Java platform offers enormous potential for cross-platform, desktop
9 applications development if only developers can be assured that it will be distributed on a
10 sufficient number of desktop computers for a sufficient amount of time to generate competitive
11 returns on their investment in applications development.

12 **B. WHY AN ORDER REQUIRING MICROSOFT TO DISTRIBUTE A COMPATIBLE**
13 **JRE WITH WINDOWS AND IE IS URGENTLY NEEDED**

14 **131.** As I discussed above, the Java platform is poised to be a successful desktop
15 application development environment, and could pose a genuine threat to Microsoft’s operating
16 system monopoly by causing developers to be indifferent as to which operating system is running
17 on the computers for which the applications are written.

18 **132.** But the success of the Java platform is wholly dependent upon its ubiquity, *i.e.*, it
19 must be distributed as broadly as Microsoft’s platforms in order to induce software developers to
20 create Java-compatible applications. Prior to Microsoft’s anticompetitive conduct, the Java
21 platform had enjoyed ubiquitous distribution, owing to its popularity among developers and
22 Microsoft’s and Netscape’s agreements to distribute an implementation of the Java runtime. It was
23 due only to Microsoft’s anticompetitive conduct – including destroying the Java distribution
24 channels, breaching its license agreement by distributing incompatible Java-based products,
25 deceiving developers, and fragmenting the Java programming environment – that the Java
26 platform has struggled on the desktop.

27 **133.** In order to restore the Java platform’s ability to challenge Microsoft’s desktop
28 operating system monopoly, Sun seeks an order to restore and preserve the possibility of

1 ubiquitous desktop distribution for the Java platform, such that developers may once again have an
2 economic incentive to create Java-based desktop applications.

3 **134.** By requiring Microsoft to distribute the most current, compatible JRE with its
4 operating systems and browsers, the Court will prevent Microsoft from using its monopoly power
5 to irrevocably tip the rapidly emerging distributed network computing industry its .NET platform
6 before a trial can be held. Such a remedy would prevent Microsoft from using its domination over
7 the distribution networks to promote its own products while cutting off the Java platform's threats
8 to Microsoft's PC-based OS monopoly.

9 **135.** Microsoft should be required to carry Sun's version of the Java runtime
10 environment with all versions of Windows and Internet Explorer (or other browser) that Microsoft
11 distributes. (It is important to note that runtime environments, such as the JRE and the Common
12 Language Runtime component of the .NET framework, make use of interfaces and functionality
13 provided by both operating systems and browsers. In any event, Microsoft distributes a copy of
14 Internet Explorer with each copy of its Windows operating system.)

15 **136.** Sun's Java runtime environment would be provided to Microsoft free of charge as
16 per the standard Java license, so all Microsoft would need to do would be to include the binary
17 code on the disk it ships to OEMs, end users, or other Windows licensees. Sun is willing to
18 provide a version of its JRE to Microsoft at least 90 days before Microsoft is due to ship the
19 relevant version of Windows.

20 **137.** As described earlier, the Java platform has enjoyed notable success in the
21 application server, server-based dynamic content, smart phone, television set-top box, and smart
22 card markets. These are all markets in which Microsoft has no monopoly.

23 **138.** Microsoft's anticompetitive acts have prevented the widespread distribution of
24 current, compatible versions of the Java platform to Windows PCs. As a consequence, the Java
25 platform has attracted much less developer interest, and thus enjoyed less success than it would
26 have enjoyed on desktop computers if Microsoft had not acted unlawfully.

27 **139.** Despite Microsoft's illegal conduct, the Java platform remains popular. For
28 example, millions of web pages still contain Java applets, though these are based on the outdated

1 four-year-old version of the platform that Microsoft has distributed. More advanced, higher-
2 performance Java desktop applications have been developed and used in companies with current
3 versions of the Java platform.

4 **140.** By eliminating the alternative channels of desktop distribution for the Java
5 platform, Microsoft has obtained the power to force the adoption of its ubiquitously distributed
6 .NET Framework.

7 **141.** In my experience, application developers prefer to write applications for those
8 platforms that are most certain to have a large, future installed base. Microsoft's closure of the
9 Java platform's distribution channel, dropping support for the Java platform, and public
10 announcement that its future products will include the .NET applications and web services will
11 influence developers to choose the .NET Framework over the Java platform.

12 **142.** Recently, I became aware that the number of web pages with Java applets has
13 decreased dramatically since Microsoft announced in October 2001 that it would not support the
14 Java platform in Windows XP. At that time, well over 7 million web pages contained Java applets.
15 Last month, the number of web pages with Java applets had dropped to 5.8 million. Today that
16 number has decreased to 5.2 million pages. This erosion in the number of Java applets used on the
17 web coincides with Microsoft's exclusion of the Java platform from Windows XP and illustrates
18 the enormous power that Microsoft's desktop monopoly entails.

19 **143.** If the distributed network computing industry tips toward .NET, Sun will suffer
20 irreparable and far-reaching harm. The future value of the Java platform would be greatly
21 diminished and possibly destroyed. The gravity of harm that will result from such tipping is
22 multiplied many times over if Microsoft succeeds in tipping the market to .NET and excluding
23 Sun's microprocessor, operating system, desktop, and server products from competing throughout
24 the distributed network computing industry. Not only will the future value of the Java platform be
25 greatly diminished, but the opportunity to interoperate across the full panoply of networked
26 opportunities free from the stifling hand of Microsoft's monopoly power will be lost.

27 **144.** Sun has licensed the Java platform to thousands of companies. There are millions
28 of Java developers. Consumers use millions of web pages containing Java applets. These

1 licensees, developers, and consumers will also suffer injury if the market tips toward .NET
2 because they will be denied the benefits of competition, including improved performance, a wider
3 variety of applications, richer features and functionality, as well as increased choice.

4 **145.** Beyond the irreparable harm to Sun, its Java licensees, Java platform developers,
5 and consumers, Microsoft's anticompetitive actions prevent innovation in the distributed
6 application industry. Microsoft's exclusion of the Java platform deprives software developers of
7 the opportunity and ability to create new types of applications, applets, features, interfaces and
8 functions that would enrich a consumer's Internet experience.

9 **146.** In my experience, I have also observed that software applications move toward a
10 standard in a fairly rapid manner. This heightens the danger of irreparable harm to Sun if
11 Microsoft succeeds in tipping the market toward .NET. If the market tips, there can then be no
12 adequate remedy for Sun, its licensees, Java developers, and consumers. Even if this Court
13 ordered the must-carry remedy at that time, it would not restore the competitiveness of the Java
14 platform because software developers would already have moved on to writing applications for
15 .NET. This proliferation in applications for the .NET Framework will raise an applications barrier
16 to entry, but this time the barrier will protect Microsoft's new monopoly over the markets for web
17 applications, services, and content.

18 **147.** Because the JRE would be required to be distributed along with Windows and
19 Microsoft's browser, developers can be confident that the Java platform will be present on all
20 Microsoft machines, and therefore they can write to the JRE if they choose to do so. Of course,
21 developers would still be free to write to other environments, such as Microsoft's .NET
22 framework, if they so desire. But they would not be constrained to do so by reason of Microsoft's
23 control over the distribution channels.

24 **148.** Microsoft's .NET framework – which runs only on Windows and is not ported to
25 other platforms – imitates aspects of the Java platform, so that, absent the Java must-carry
26 distribution remedy, many developers would see no point in writing to the Java platform if they
27 could write to .NET. This is because the .NET framework is distributed with all new versions of
28 Windows, and it is therefore guaranteed to be on every PC, whereas the JRE currently is not.

1 **149.** In the near future, developers will have to make a choice between writing to the
2 Java or .NET platform, and if their choice is tilted toward .NET merely because Microsoft has
3 used its monopoly to bias the choice, then Microsoft will have successfully headed off the threat
4 that the Java platform poses to its PC operating system monopoly.

5 I declare under penalty of perjury under the laws of the United States that the foregoing is
6 true and correct to the best of my knowledge.

7 Signed this 26th day of March 2002.

8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

By: _____
Richard Green