Case Nos. 2013-1021, -1022

UNITED STATES COURT OF APPEALS

FOR THE FEDERAL CIRCUIT

ORACLE AMERICA, INC.,

Plaintiff-Appellant,

v.

GOOGLE INC.,

Defendant-Cross Appellant.

Appeal from the United States District Court for the Northern District of California in Case No. 10-cv-3561, Judge William H. Alsup

---

BRIEF OF AMICI CURIAE COMPUTER SCIENTISTS IN SUPPORT OF DEFENDANT-CROSS APPELLANT AND AFFIRMANCE

Julie P. Samuels
(*Principal Attorney of Record*)
Michael Barclay
ELECTRONIC FRONTIER FOUNDATION
815 Eddy Street
San Francisco, CA 94109-7701
Tel: (415) 436-9333
Fax: (415) 436-9993
julie@eff.org

Jason M. Schultz
NYU Technology Law and Policy Clinic
NYU SCHOOL OF LAW
40 Washington Square South
New York, NY 10012
Tel: (212) 998-6100

Attorneys for Amici Curiae Computer Scientists

May 30, 2013

CERTIFICATE OF INTEREST

Pursuant to Federal Circuit Rules 29(a) and 47.4, counsel for Amici Curiae certifies that:

1. The full names of the amici represented by me are:

See the Attachment on the next page.

2. The name of the real party in interest (if the party named in the caption is not the real party in interest) represented by me is:

N/A

3. All parent corporations and any publicly held companies that own 10 percent or more of the stock of the amicus curiae represented by me are:

None.

4. The names of all law firms and the partners or associates that appeared for the party or amicus now represented by me in the trial court or are expected to appear in this Court are: Julie P. Samuels and Michael Barclay, Electronic Frontier Foundation, San Francisco, California; Jason M. Schultz, NYU Technology Law and Policy Clinic, New York, New York.

May 30, 2013

/s/ Julie P. Samuels
Julie P. Samuels
Attorneys for Amici Curiae Computer Scientists

ATTACHMENT TO CERTIFICATE OF INTEREST
LIST OF AMICI CURIAE
(In alphabetical order)

1. John Perry Barlow
2. Brian Behlendorf
3. Richard A. Belgard
4. Jon Bentley
5. Matthew Bishop
6. Frederick Brooks
7. David Dill
8. Les Earnest
9. Brendan Eich
10. Dave Farber
11. Jeremiah Flerchinger
12. Martin Fowler
13. Allan Gottlieb
14. David Klausner
15. Kin Lane
16. Doug Lea
17. Paul Menchini
18. Martin Odersky
19. Tim Paterson
20. Tim Peierls
21. Simon Phibbs
22. Bill Pugh
23. Larry Roberts
24. Bruce Schneier
25. Curtis Schroeder
26. Dave Snigier
27. Bjarne Stroustrup
28. Paul Sutter
29. Michael Tiemann
30. Andrew Tridgell
31. Josh Triplett
32. Phil Wadler

An expanded list of amici curiae, with short biographies of each, can be found at https://www.eff.org/cases/oracle-v-google/amici.

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page(s)

Federal Cases

Federal Statutes

Other Authorities

Legislative Materials

STATEMENT OF IDENTITY AND INTEREST OF AMICI CURIAE[1]

Amici are numerous individual computer scientists who believe the District Court correctly decided this case and who urge this Court to uphold that ruling. The signatories to this brief include some of the leading, pioneering scientists in the computer industry. They have invented or contributed to the authorship of numerous computer programs. They have joined this brief because they believe the District Court correctly rejected Oracle's attempt to overextend copyright coverage in a manner irreconcilable with the purpose of copyright law and the nature of computer science.

As computer scientists, amici have relied on the open nature of APIs and the programs built on them to create and operate new software. Amici depend on APIs remaining open to sustain widespread compatibility standards used by

[1] No party's counsel authored this brief in whole or in part. Neither any party nor any party's counsel contributed money that was intended to fund preparing or submitting this brief. No person other than amici, their members, or their counsel contributed money that was intended to fund preparing or submitting this brief. At one point, the District Court ordered the parties to disclose any financial relationships with commentators about this case. Dkt. No. 1229 (Order dated August 7, 2012). In response, Google identified EFF as an organization to which it has contributed, and specifically identified two of EFF's lawyers who are counsel on this brief. Dkt. No. 1240 (Google's Response to Order to Supplement at 7-8 (August 24, 2012)). The district court took no further action. Dkt. No. 1242 (Order dated September 4, 2012). To make it clear, under Fed. R. App. P. 29(c)(5), Google's general contributions to EFF were not intended to fund preparing or submitting this brief.

NYU law students Sam Zeitlin and Charlotte Slaiman assisted in the preparation of this brief. Web sites cited in this brief were last visited on May 28, 2013.

startups and incumbents alike. Reversing the District Court would dangerously undermine the settled expectations of computer scientists who rely upon the open nature of APIs.[2]

Pursuant to Fed. R. App. P. 29(a), all parties have consented to the filing of this brief.

INTRODUCTION AND SUMMARY OF ARGUMENT

The freedom to reimplement and extend existing APIs has been the key to competition and progress in the computer field—both hardware and software. It made possible the emergence and success of many robust industries we now take for granted—such as industries for mainframes, PCs, peripherals (storage, modems, printers, sound cards, etc.), workstations/servers, and so on—by ensuring that competitors could challenge established players and advance the state of the art.

Thus, excluding APIs from copyright protection has been essential to the development of modern computers and the Internet. For example, the widespread availability of diverse, cheap, and customizable personal computers owes its existence to the lack of copyright on the specification for IBM's Basic Input/Output System (BIOS) for the PC. Companies such as Compaq and Phoenix reimplemented IBM's BIOS without fear of copyright claims, making

[2] An expanded list of amici, with short biographies of each, can be found at https://www.eff.org/cases/oracle-v-google/amici.

PC clones possible. And the open nature of APIs was essential to many modern computing developments, including those of operating systems such as UNIX, programming languages such as "C", the Internet's network protocols, and cloud computing.

The uncopyrightable nature of APIs spurs the creation of software that otherwise would not have been written. When programmers can freely reimplement or reverse engineer an API without the need to negotiate a costly license or risk a lawsuit, they can create compatible software that the interface's original creator might never have envisioned or had the resources to create. Moreover, compatible APIs enable people to switch platforms and services freely, and to find software that meets their needs regardless of what browser or operating system they use. Without the compatibility enabled by the open nature of APIs, consumers could be forced to leave their data behind when they switch to a new service.

The freedom to reimplement APIs also helps rescue "orphan" software or data—systems whose creators have either gone out of business or abandoned their product in the marketplace. Government entities and non-profits are especially susceptible to the orphan programs problem as they often cannot afford to upgrade and are left using legacy technologies for years or decades.

When a popular computer platform or service shuts down, the ability to freely reimplement APIs protects the communities that rely on that software.

This Court's decision will determine whether or not APIs continue to remain open and thereby enable competition and innovation among software developers, and enable access to information and affordable technology for users. Should the Court reverse Judge Alsup's well-reasoned opinion, it will hand Oracle and others the ability to monopolize any and all uses of systems that share their APIs. API creators would have veto power over any developer who wants to create a compatible program—regardless of whether she copies any literal code from the original API implementation. That, in turn, would upset the settled business practices that have enabled the American computer industries to flourish, and choke off many of the system's benefits to consumers. Therefore, we respectfully request that this Court affirm the District Court's decision.

ARGUMENT

I. UNCOPYRIGHTABLE INTERFACES WERE ESSENTIAL TO THE DEVELOPMENT OF MODERN COMPUTERS AND THE INTERNET

The free and open use of APIs has been routine in the computer industry since its beginning. As explained below, the uncopyrightable nature of APIs and other interfaces was essential to the development of the home computer, operating systems, programming languages, the Internet, and cloud computing. These examples represent an important principle: the success of the integrated

system of technology we take for granted today depends on open interfaces. The more that individual companies can lock down communication standards that connect programs, the more information technology becomes fragmented, cutting users off from each other and from the benefits of computer innovation.

As Oracle's lead trial lawyer noted before this dispute began, "compatibility in the personal computer arena has fostered innovation and competition—two critical policy objectives." Michael Jacobs, *Copyright and Compatibility*, 30 Jurimetrics J. 91, 93 (1989-90). The uncopyrightability of APIs was key to the development of each major layer of computing that we use today, from the most fundamental to the cutting edge. The Court should consider this history carefully before changing the law and disrupting the settled expectations of the development community and of millions of programmer/developers.

A. The BIOS of the Original IBM-Compatible PC

One can trace the importance of keeping APIs free from copyright back to home computing's origins. In 1981, IBM released its first home computer, the PC. Charles H. Ferguson & Charles R. Morris, *Computer Wars: The Fall of IBM and the Future of Global Technology* 27–28 (1994). Unlike prior offerings, the IBM PC soon dominated the computer market because of its open design, which allowed entire industries of PC software and hardware peripheral producers to

emerge. *Id*. at 28–29. IBM-exclusive software, like the popular spreadsheet program Lotus 1-2-3, forced many users to buy IBM computers. *Id*. And although other computer manufacturers could run the same MS-DOS operating system that IBM used, many best-selling programs required complete hardware and Basic Input/Output System (BIOS) firmware[3] compatibility as well, thus making the IBM model the *de facto* standard. *Id*. at 51–53; *see, e.g.*, Stephen Satchell, *The Corona ATP is Faster than the IBM PC AT, But it Has Flaws*, InfoWorld, Jan. 1986, at 50 (using Microsoft Flight Simulator and Lotus 1-2-3 to test PC compatibility).

Thus, in order to create a computer that was truly competitive with the IBM PC, other manufacturers needed to duplicate the functionality of IBM's BIOS firmware. *See* Ferguson, *supra*, at 55–53. To avoid exposing themselves to copyright liability, Phoenix, Compaq, and other hardware manufacturers assembled "clean" teams of programmers who had never seen the BIOS source code. Van Lindberg, *Intellectual Property and Open Source: A Practical Guide to Protecting Code* 240–41 (2008). The clean teams created new software from scratch using the interface specifications needed to interact successfully with the

[3] Firmware is software stored in read-only memory that stays intact even when a computer is switched off. Microsoft Dictionary (2005) at 357. Firmware holds the most basic pieces of software in a computer, like startup routines and the interface that allows the operating system to interact with the computer hardware. *See generally* Jeff Tyson, *How BIOS Works*, HowStuffWorks, http://computer.howstuffworks.com/bios1.htm.

IBM PCs: the BIOS API, including its structure, sequence, and organization. *Id*. Once these firms developed their own BIOS firmware, they were able to produce cheaper, faster IBM-compatible computers, and market innovations like the first portable PC. Ferguson, *supra*, at 53–55; *see also Compaq Computer Corporation: Portable Computer*, Encyclopedia Britannica.[4] With more computers and customers now available to them, software developers began to write and distribute more software than ever, innovating with new features and functionality and competing directly on price. The age of home computing began in earnest.

The story of Compaq's and Phoenix's creation of PC clones embodies one of the core copyright principles at issue in this case, as laid out by the Supreme Court more than a century ago in *Baker v. Selden*, 101 U.S. 99 (1879). There, the Supreme Court held that while the author of a book on accounting owned a copyright on his text, copyright could not grant a monopoly over the accounting system contained within. *Id*. at 104. In fact, it would be "a surprise and fraud upon the public" if copyright could grant exclusive rights to a system without any examination of its novelty. *Id*. at 102. Congress incorporated this insight into law at 17 U.S.C § 102(b). Nat'l Comm'n on New Technological Uses of

[4] *See* http://www.britannica.com/EBchecked/media/19722/The-Compaq-portable-computer-Compaq-Computer-Corporation-introduced-the-first.

Copyrighted Works, *Final Report* 19 (1979) [hereinafter CONTU].[5] Section 102(b), which denies copyright to any "process, system, or method of operation," ensures that a programmer's copyright protects his code, but does not protect the processes or methods that code enables a computer to perform. H.R. Rep. No. 94-1476 at 56–57, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5670 [hereinafter House Rep.]; S. Rep. No. 94-473, at 54 (1975) (found under heading "Nature of copyright" addressing Section 102) [hereinafter Senate Rep.]. IBM owned the copyright on its BIOS source code, but that did not give it a monopoly on the system of commands the operating system used to communicate with the BIOS. Compaq and Phoenix were entitled to reimplement the BIOS interface as long as they did not copy any of IBM's code. *Cf. Lotus Dev. Corp. v. Borland Int'l, Inc.*, 49 F.3d 807, 810 (1st Cir. 1995) (holding that the menu structure and commands of Lotus's interface comprised an uncopyrightable system or method of operation under § 102(b), and that Borland was free to reimplement them), *aff'd by an equally divided Court*, 516 U.S. 233 (1996). Notably, it is irrelevant under § 102(b) whether the API's creator intends to keep its API proprietary or freely dedicate it to the public—either way, APIs are unprotectable methods of operation.

[5] Available at: http://digital-law-online.info/CONTU/PDF/Chapter3.pdf.

B. Major Modern Operating Systems Reimplement the Groundbreaking UNIX API

Many popular operating systems today reimplement the APIs of one of the earliest operating systems created. In 1969, Ken Thompson, Dennis Ritchie, and other computer scientists at AT&T Bell labs developed the first modern operating system. Heather J. Meeker, *The Open Source Alternative* at 3–4 (2008). They called it UNIX. *Id.* Predating the home computer by a decade, UNIX ran on large mainframe computers owned by corporations, universities, and the government. *Id.* At the time AT&T developed UNIX, the company was operating under a 1956 consent decree (the result of an antitrust suit) that forbade it from monetizing any project outside of telecommunications and special federal contracts. *Milestones in AT&T History*, ATT.com.[6] To comply with the decree, AT&T licensed UNIX to any interested party for a nominal fee. Meeker, *supra*, at 4. Computer scientists embraced UNIX, making it the dominant operating system of its day. *Id.* Programmers shared their source code and programming innovations freely, developing and releasing new versions of the operating system. *Id.*

The original versions of UNIX became obsolete as the computers that ran them changed, but the UNIX platform could always return in new forms because AT&T's copyright in the UNIX code didn't extend to its API. Software

[6] *See* http://www.corp.att.com/history/milestones.html.

developers dissatisfied with available operating systems such as MS-DOS, Windows, and Apple's system, along with UNIX users, reimplemented the UNIX API to run on a PC.[7] Meeker, *supra*, at 6. They wanted to create a new operating system that would run software made for UNIX, but was also free of AT&T's (or anyone's) intellectual property, specifically a system comprising only free software. *Id.* The GNU project, together with the Finnish programmer Linus Torvalds, produced the Linux operating system, which shares the UNIX API—including its structure, sequences, and organization—but uses entirely original code. *Id.* Today, tens of millions of servers run Linux. Steven J. Vaughan-Nichols, *How Many Linux Users Are There (Really)?* Linux Planet (Feb. 18, 2009).[8] Countless Internet-based services from Facebook to ATMs rely on Linux-based high-speed networking systems. *Id.* Because the API was open, it took a minimal amount of work to make pre-existing software run on subsequent systems.

Another successful reimplementation of the UNIX API came from Apple Inc. During the 1980s and 90s, Apple was known for its isolated computing system incompatible with mainstream software development. Daniel Eran

[7] MS-DOS itself reimplemented the API of an earlier operating system, CP/M. *Paterson v. Little, Brown & Co.*, 502 F. Supp. 2d 1124, 1128 (W.D. Wash. 2007).

[8] Available at: http://www.linuxplanet.com/linuxplanet/reports/6671/1.

Dilger, *How CPR Saved Apple,* Roughly Drafted Magazine (Oct. 23, 2006).[9] Over time, the lack of software began to choke off Apple's user base. *Id.* In 1999, Apple abandoned the OS it had developed through nine versions since 1984. *Id*. Its tenth operating system, OS X, had a new base: the UNIX API. *Id.* By using the UNIX API, Apple hoped to win over the UNIX user base of sophisticated technologists and attract the UNIX developer community to write software for Macs. *See* Joe Wilcox, *Will OS X's Unix Roots Help Apple Grow?*, CNET.com (May 21, 2001).[10] Apple is now the world's largest computer company, and OS X the most popular UNIX-compliant personal computer operating system. Nick Wingfield, *Apple Becomes the Most Valuable Public Company Ever, With an Asterisk*, N.Y. Times (August 20, 2012);[11] *Top Operating System Share Trend*, Netmarketshare (April 2003).[12]

The many implementations of the UNIX API represent exactly the kind of innovation and competition that Congress envisioned when it passed § 102(b). Congress wrote that section to respond to concerns that computer copyright would "extend protection to the methodology or processes adopted by the

---

[9] Available at: http://www.roughlydrafted.com/RD/Q4.06/469013E9-454C-42F0-AFB1-FA75871A028B.html.
[10] Available at: http://news.cnet.com/Will-OS-Xs-Unix-roots-help-Apple-grow/2100-1040_3-257982.html.
[11] Available at: http://bits.blogs.nytimes.com/2012/08/20/apple-becomes-the-most-valuable-public-company-ever-with-an-asterisk/.
[12] Available at: http://www.netmarketshare.com/os-market-share.aspx?qprid=9.

programmer, rather than merely to the 'writing' expressing his ideas." House Rep. at 56–57; Senate Rep. at 54. The law has thus allowed many individuals and companies to each provide its own code behind a UNIX interface, letting consumers adopt the right one to fit their needs.

C. The C Programming Language Became Universal Because of Its Uncopyrightable Interface

One of the most important contributions of open interface specifications to computer science was enabling software written in one programming language to run on any operating system. Dennis Ritchie, one of the computer scientists who invented UNIX, also co-invented a new language in which to code it, called "C". P.J. Plauger, *The Standard C Library* 3 (1991). Because C has been used so widely, the C Standard Library API has been reimplemented countless times to allow different operating systems to work with programs written in C. For example, Microsoft reimplemented the C Standard Library for Windows as part of the Microsoft C Run-Time Library. *C Run-Time Libraries*, Microsoft Developer Network.[13] Google's reimplementation of the same for Android is called Bionic. *The Native Android API*, Mobile Pearls.[14] Another significant reimplementation was the GNU C Library, which was essential to the

[13] Available at: http://msdn.microsoft.com/en-us/library/abx4dbyh(v=vs.80).aspx.
[14] Available at: http://mobilepearls.com/labs/native-android-api/.

GNU Project's effort to create a free UNIX-compatible operating system. *The GNU C Library (glibc)*, The GNU Project.[15]

Programs written in C use the C standard library to execute their functions and operate the computer on which they run—including tasks as basic as opening and closing files on the hard drive.

Limiting the ability to reimplement the C Standard Library would have likewise severely limited the range of systems on which C programs could run. Each operating system would require a unique set of libraries for C-based programs to call on, effectively creating a new, incompatible version of the language. Today, once a programmer learns C, he can write code that will run on any operating system that can run a reimplementation of the C standard library. API copyright would turn universal programming languages like C into narrow dialects, usable only on a specific operating system. Many innovative software projects would be restricted to a single operating system, or simply never get off the ground at all. Old programs would run the risk of becoming obsolete whenever a new operating system came into use, and new operating systems would be unable to take advantage of the thousands of existing C programs.

[15] Available at: http://www.gnu.org/software/libc/.

D. Computers Rely on the Uncopyrightable Nature of APIs and Network Protocols to Communicate Over the Internet

Open interface standards have been particularly important to the development of the Internet because the Internet's entire purpose is to let computer systems around the world communicate with each other. The many reimplementations of the Berkeley Systems Distribution (BSD) "sockets" API ensure that application developers only have to write the networking sections of their program once for it to function on almost every operating system. BSD sockets, designed to help computers connect to the Internet, were one of the great innovations of the early UNIX diaspora. Kaare Christian & Susan Richter, *The UNIX Operating System* 6 (3d ed. 1994). In 1983, the Computer Systems Research Group at the University of California-Berkeley created BSD sockets for its new version of UNIX. Mark Muggeridge, *Programming with TCP/IP—Best Practices*, 3 HP OpenVMS Technical Journal 3, 5 (2004).[16] BSD's version of UNIX used the new API to control and operate network sockets—the starting and ending point for any communication over the Internet. Christian, *supra*, at 502. Today, every major operating system, including Windows, allows

[16] Available at: http://h71000.www7.hp.com/openvms/journal/v3/tcpip.pdf.

applications to connect to the Internet via an implementation of the BSD sockets API. *See, e.g.*, *About Berkeley Sockets and Winsock*, VMware.[17]

System designers and application programmers alike rely on the BSD sockets API as a de facto industry standard. *See, e.g.,* Johnson M. Hart, *Windows System Programming*, ch. 12 (4th ed. 2010). Because the BSD sockets API is free of copyright, OS developers can redesign the implementation of the API to improve its efficiency, secure in the knowledge that consumers will be able to keep using all their programs. Network software specialists can write new implementations of the API, which can then be integrated into any operating system. Application programmers only have to write the networking section of their programs once and it will work on any operating system, making it cheaper and easier to make software available for multiple platforms.

Even more crucial to the development of the Internet than the Berkeley sockets API were network protocols. Network protocols are computer interfaces very similar to APIs. APIs are rules for how programs communicate within a single computer, while network protocols are rules for how programs communicate between computers. One of the most important network protocols is the hypertext transfer protocol (HTTP). Larry L. Petersen & Bruce S. Davie, *Computer Networks: A Systems Approach* 640–42 (2d ed. 2000). Whenever an

[17] Available at: http://pubs.vmware.com/vsphere-51/index.jsp?topic=%2Fcom.vmware.vmci.pg.doc%2FvsockAppendix.8.2.html.

Internet user clicks a link, her computer sends an HTTP command requesting the appropriate webpage from the server on which it is stored. *Id.* Because no one has ever asserted copyright in this protocol, anyone is free to write an implementation of the HTTP interface—meaning that his or her program can send and respond to HTTP requests.

HTTP was created by Tim Berners-Lee, the inventor of the Web, and developed as a standard in large part by the World Wide Web Consortium, an organization he founded and runs. Tim Berners-Lee, *Biography*, World Wide Web Consortium;[18] Roy Fielding, *et al*., *RCFC 2616: Hypertext Transfer Protocol—HTTP/1.1*, Internet Engineering Task Force (June 1999).[19] If interface specifications were copyrightable, Berners-Lee could have used copyright to determine which software and hardware manufacturers could use the Web. Mutually incompatible networking protocols would cause the Internet to fragment into isolated online communities incapable of exchanging information.

Microsoft's Server Message Block (SMB) system nearly caused a similar kind of fragmentation to happen to local networks. The SMB network protocols govern local area networks, often used in homes and offices to allow local users to share files and printers. Richard Sharpe, *Just What is SMB?* Samba.org (Oct.

[18] Available at: http://www.w3.org/People/Berners-Lee/Overview.html#Bio.
[19] Available at: http://tools.ietf.org/html/rfc2616.

8, 2002).[20] When Microsoft added SMB to Windows it released very little documentation for the SMB interface specification, making it difficult to reimplement SMB on any other operating system. *See* Andrew Tridgell, *How Samba was Written*, Samba.org (Aug. 2003).[21] This meant that there was no way to link Windows computers together with computers running other operating systems in the same network. *See* Chris Hertel, *Samba: An Introduction*, Samba.org (Nov. 27, 2001).[22]

Beginning in 1991, however, amicus computer scientist Andrew Tridgell successfully deduced the interface specifications for SMB. *See* Tridgell, *supra.* Once Tridgell had discovered the commands that software would need to send and receive in order to be SMB-compatible, he wrote a UNIX-compatible open-source reimplementation called Samba. *See* Hertel, *supra.* Today, Samba is used by most UNIX-compatible operating systems. Gerald Carter, *et al*., *Using Samba: A File and Print Server for Linux, Unix & Mac OS X* 3 (3d ed. 2007). And in 2011, Apple created a new reimplementation of SMB for OS X. David Morgenstern, *Samba Growing Pains Continue in OS X Lion*, ZDNet, (July 23, 2012, 7:21 p.m.).[23]

---

[20] Available at: http://www.samba.org/cifs/docs/what-is-smb.html.
[21] Available at: http://www.samba.org/ftp/tridge/misc/french_cafe.txt.
[22] Available at: http://www.samba.org/samba/docs/SambaIntro.html.
[23] Available at: http://www.zdnet.com/samba-growing-pains-continue-in-os-x-lion-7000001353/.

BSD sockets and network protocols are industry standard systems of communication. Although many programs reimplement these interfaces, § 102(b) ensures that such uses will not create copyright liability. *See Lotus*, 49 F.3d at 807. If the Court finds APIs subject to copyright, it could mean that even interfaces like HTTP might be copyrightable.

E. The Uncopyrightable Nature of APIs Forms the Industry Standards for Cloud Computing

Modern cloud computing providers like Amazon Web Services rely on a reimplementation of one of the oldest APIs: the IBM BIOS. Cloud computing allows users to rent space and processing power on distant servers, accessible from anywhere in the world via the Internet. *What is Cloud Computing?*, Amazon Web Services.[24] At their core, cloud computing clusters act as "virtual machines"—imitations of small computers being run on huge servers. *See id.*; *see also* Margaret Rouse, *Definition: Virtual Machine (VM)*, SearchServerVirtualization (Oct. 2011).[25] Virtual machines call the functions of the BIOS API just like physical computers, but they have no individual physical hardware. *See id.* Instead, a reimplementation of the BIOS API allows the server to answer the API calls of all the virtual machines running on it. *See id.*

[24] Available at: http://aws.amazon.com/what-is-aws.
[25] Available at: http://searchservervirtualization.techtarget.com/definition/virtual-machine.

Cloud computing providers also use an API to govern how their users can interact with their services. *Amazon Web Services (AWS) and Eucalyptus Partner to Bring Additional Compatibility Between AWS and On-Premises IT Environments*, Eucalyptus (March 22, 2012).[26] Most providers rely on Amazon's cloud services API to allow users to control and operate the cloud computers that they rent. The API is ineligible for copyright protection under § 102(b). Just as copyright law allowed Baker to compete with Selden to make the most useful book from Selden's accounting system, § 102(b) ensures that companies like CloudStack and Eucalyptus can compete with Amazon to provide the best implementation of the cloud services API.

Businesses that employ cloud services write or commission their own proprietary software to perform operations on cloud servers. *Business Applications*, Amazon Web Services.[27] This software is built around the cloud service's API. Today, since major cloud service providers like Amazon, Eucalyptus, and CloudStack use the same standard specifications for their APIs, their customers can easily switch from one cloud service to another. Steven J. Vaughan-Nichols, *OpenStack vs. CloudStack: The Beginning of the Open-*

[26] Available at: http://www.eucalyptus.com/news/amazon-web-services-and-eucalyptus-partner.
[27] Available at: http://aws.amazon.com/business-applications.

Source Cloud Wars, ZDNet (Apr. 12, 2012).[28] Software developers can write programs capable of interacting with all three services, creating new ways for users to access and manipulate information spread out across the Internet.

If copyright allowed Amazon to monopolize its cloud storage API, Amazon would be able to use that power to lock in its users and cripple new competitors. Because businesses use custom software built around the cloud service provider's API, switching to a cloud service provider with a different API would require rewriting their cloud software. The cost and disruption of doing so would mean that few businesses would be willing to leave their cloud service provider, leaving late entrants in the cloud service market to find it impossible to build a customer base and leaving consumers with less choice.

Cloud service APIs demonstrate how APIs link the past and the future of computing. Cloud services exist because their creators could build on the openness of the BIOS API. As discussed above, IBM enforced copyright on the BIOS source code and would certainly have used copyright to control reimplementations of the BIOS API if the law allowed. Using this shared resource, cloud service providers created a new service with a new API that is quickly becoming a compatibility standard in their field. Only time will reveal

[28] Available at: http://www.zdnet.com/blog/open-source/openstack-vs-cloudstack-the-beginning-of-the-open-source-cloud-wars/10763.

what new innovations will take advantage of widespread, compatible cloud services.

II. UNCOPYRIGHTABLE INTERFACES SPUR THE CREATION OF SOFTWARE THAT OTHERWISE WOULD NOT BE WRITTEN

When programmers can freely employ any interface without the need to negotiate a costly license or risk a lawsuit, they may create compatible software that the interface's original creator might never have envisioned or had the resources to create.

Copyrightable APIs would discourage this innovation by creating potential infringement liability for the mere act of writing a compatible program, even where no literal code is copied. The Copyright Act protects a programmer's source code as creative expression, but does not cover the processes, systems, and methods of operation that code may employ to interface with other software. 17 U.S.C. § 102(b); *see also* House Rep. at 56-57; Senate Rep. at 54. In the Ninth Circuit, the "functional requirements for compatibility" between computer programs "are not protected by copyright" under § 102(b). *Sega Enters., Ltd., v. Accolade, Inc.,* 977 F.2d 1510, 1522 (9th Cir. 1992); *see also Sony Computer Ent'mt, Inc. v. Connectix Corp.*, 203 F.3d 596, 599–600 (9th Cir. 2000) (describing Sony's Playstation BIOS as a "system interface procedure[]" that Connectix was entitled to reimplement under § 102(b)).

A. Uncopyrightable Interfaces Allow Software that Makes Different Systems Compatible

One straightforward and common reason to reimplement another programmer's API is to make a program compatible with a different application or platform. Small companies and volunteer groups often undertake such projects, but heavy licensing fees or the threat of litigation over API copyright would hinder this work.

Reimplemented APIs create compatibility for users as well as developers. Wine is a compatibility layer that reimplements the Windows API so that Windows programs can run on UNIX-based operating systems like Linux and Mac OS X. *About Wine*, WineHQ.[29] Millions of people use Wine to make their favorite Windows programs work on other operating systems. *Id.* Wine is free and open source software, and volunteers write much of its code. *Id.* Microsoft has no agreement and no contact with the Wine project. Scott Swigart & Sean Campbell, *Interview with Alexandre Juliard, Head of the Wine Project/CTO of CodeWeavers*, How Software is Built (Sept. 8, 2008).[30] In fact, Microsoft has used its anti-piracy tools to prevent Wine users from updating their software. Ingrid Marson, *Microsoft Admits Targeting Wine Users,* ZDnet (Feb. 25,

[29] Available at: http://www.winehq.org/about/.
[30] Available at: http://howsoftwareisbuilt.com/2008/09/09/interview-with-Alexandre-Julliard-Head-of-the-Wine-Project-CTO-of-CodeWeavers/#difficulty.

2005).[31] If copyright law gave Microsoft a monopoly on the Windows API, Microsoft could demand licensing fees from Wine, or sue for statutory damages. Either outcome might lead to the Wine project shutting down permanently, preventing its users from running software they have legally purchased or licensed on their own computers.

In the context of supercomputers, reimplementing an API is often necessary to make new hardware compatible with existing software. Supercomputers typically have unusual, custom-built hardware reflecting their purpose and the state of the art in computer design at the time of their manufacture. In order for supercomputers to operate effectively, they need software written specially for their hardware architecture. *See* National Research Council, *The Future of Supercomputing: An Interim Report* 4, 17 (2003). Supercomputer vendors create specially tuned and optimized implementations of APIs like the Basic Linear Algebra Subprograms (BLAS) library so that scientists and mathematicians can use the API to write code for their research and experiments. *See BLAS Frequently Asked Questions*, Netlib (Jul. 25,

[31] Available at: http://www.zdnet.com/microsoft-admits-targeting-wine-users-3039189180/.

2005);[32] IBM, *Basic Linear Algebra Subprograms Library Programmer's Guide and API Reference* (2008).[33]

A shared API is particularly important for supercomputers, because time on these machines is often limited. *See e.g.*, *Scheduling Policies and Limits*, Ohio Supercomputers Center;[34] *PBS Information for Labs and the Lab Queue*, Minnesota Supercomputing Institute.[35] As a result, supercomputer users must write and test their programs on smaller computers and only use the supercomputer when they wish to run the program for research or experimental purposes. The program must work on both the smaller computer and the supercomputer, even though the two systems have different hardware and demands. BLAS and similar APIs provide a compatibility standard that allows supercomputers to run software written on other machines, as long as both machines have valid implementations of the same interfaces.

Copyright on the APIs for supercomputer libraries like BLAS would burden the academics who rely on them. A research project's codebase (all the code they've written) is a significant investment, tied to the APIs chosen at the

---

[32] Available at: http://www.netlib.org/blas/faq.html.
[33] Available at: http://webpages.uncc.edu/~apanday/documents/BLAS_Prog_Guide_API_v3.0.0.3.pdf.
[34] Available at: https://www.osc.edu/supercomputing/batch-processing-at-osc/scheduling-policies-and-limits.
[35] Available at: https://www.msi.umn.edu/labs/pbs.

start. *See* National Research Council, *supra*, at 21. Moving to an incompatible new supercomputer is a hugely expensive and wasteful proposition. *Id.* If every supercomputer vendor had its own proprietary API, then research groups would find themselves "locked in" to the vendor they worked with at the start of their project. Supercomputer users would be unable to switch vendors to escape poor service or gain access to new technology without making their codebase useless. New vendors with improved supercomputer technology would be unable to attract customers, making it harder to successfully bring hardware innovations to market.

B. Uncopyrightable Interfaces Help Programmers Develop Completely New Capabilities for Software

Developers reverse engineer APIs to write programs that add new features or provide new uses for online services. Roman Irani, *The Curious Case of Unofficial APIs*, Programmable Web (Nov. 15, 2011).[36] For instance, many popular websites have companion mobile apps. *Id.* The website needs an API so that the app can communicate with it, even if that API is internal (*i.e.* unavailable to the public). *Id.* By monitoring traffic between the website and the app, a developer can reverse engineer the internal API, figure out its rules, and write extra code (called a "client wrapper") to make it easier for outsiders to use.

[36] Available at: http://blog.programmableweb.com/2011/11/15/the-curious-case-of-the-unofficial-apis/.

Id. If the unofficial API is posted online, anyone can write their own software compatible with the website. *Id.* For instance, a programmer named Mislav Marohnić discovered and published an unofficial API for the photo sharing service Instagram. Adam DuVander, *The Full-featured, Unpublished Instagram API*, Programmable Web (Dec. 15, 2010).[37] Software soon sprang up to take advantage of the new API, offering new options to Instagram users. Web developers began using the API to integrate Instagram photos into the sites they created. When Instagram saw how much demand there was for Marohnić's API, it launched an official Instagram external API that app and web developers could use directly. Adam DuVander, *Instagram Shuts Down Developers, Plans Official API*, Programmable Web (Jan. 12, 2011).[38] Today, organizations of all stripes use the Instagram API to feature their members' photography and share photos with customers and fans. *Instagram for Business*, Instagram Help Center.[39]

If APIs were not excluded from protection under § 102(b), then when Marohnić publically posted his unofficial Instagram API, he would have

---

[37] Available at: http://blog.programmableweb.com/2010/12/15/the-full-featured-unpublished-instagram-api.
[38] Available at: http://blog.programmableweb.com/2011/01/12/instagram-shuts-down-third-party-developers-plans-official-api.
[39] Available at: http://help.instagram.com/customer/portal/articles/95806-examples-of-how-brands-are-using-instagram (select "Examples of How Brands Are Using Instagram").

infringed Instagram's copyright, and become liable for statutory damages. If the fear of suit deterred people like Marohnić from helping developers produce compatible software or websites, Instagram may never have realized the full potential of its service.

III. COPYRIGHT IN INTERFACES WOULD CREATE AN "ORPHAN SOFTWARE" PROBLEM

Programmers frequently need to reimplement APIs in order to access data or other resources trapped in obsolete software. Software creators go bankrupt or stop supporting their creations, and the intellectual property in software is often bought and sold when startups are acquired or divisions of companies spin off or shut down. These characteristics create a class of "orphan software", whose copyright owners are hard to find. Obsolete software becomes incompatible with modern computers and other modern software as platforms change. For owners of now-orphaned software, reimplementing an API associated with that software may be the only realistic way to reclaim the time and resources they have invested in it.

Section 102(b) solves this problem. When a copyright owner goes missing, it is difficult to get the right to make derivative works of the code. However, a program's interface specifications are part of its system or method of operation, rather than part of its copyrightable expression. By keeping interface specifications free of copyright, Congress allowed other developers to

build compatible systems. An orphan program's original implementation may be lost, obsolete, or inoperable, but any developer is free to build a new compatible program. Allowing copyright law to prevent an entire community of users and third-party developers from switching easily to another service would subvert the purpose of § 102(b).

A. Uncopyrightable Interfaces Protect Both Developers And Users

Platform developers strive to create communities of users and developers around their platform. The ability to freely reimplement APIs protects these communities when the platform for their services is shut down.

For example, when the social bookmarking site Delicious collapsed, a new service reimplemented the API for the apps and users that depended on it. Delicious was a popular site where users could post links to interesting content that they found around the web. Bobbie Johnson, *Oh, Delicious—Where Did It All Go So Wrong?*, GigaOm (Sept. 28, 2011).[40] People used a variety of third-party applications that ran on the Delicious API to read and post information on Delicious. *A Tour of Pinboard*, Pinboard.[41] Yahoo! bought Delicious and slowly phased out its development, hemorrhaging Delicious users along the way. Bobbie Johnson, *supra.* As the size of the community diminished, so did the

[40] Available at: http://gigaom.com/2011/09/28/oh-delicious-where-did-it-all-go-so-wrong.
[41] Available at: http://pinboard.in/tour#api.

usefulness of Delicious. Kristina Dell, *Entrepreneurs Who Go It Alone—By Choice*, Time (Oct. 24, 2011).[42] Eventually Yahoo! sold Delicious, and many users decided to find a new place to go. *Id.*

A new social bookmarking site, Pinboard, offered itself up as a haven for former Delicious users. *Id.* By reimplementing the Delicious API, Pinboard allowed users to keep using their Delicious-based applications, but with Pinboard instead. Pinboard, *supra*. Pinboard was created by one man, Maciej Ceglowski, in his spare time. Dell, *supra*. If Ceglowski had to pay for an expensive license or risk copyright liability in order to reimplement the Delicious API, he probably wouldn't have gone ahead with the project. The Delicious-based applications would have become useless.

As the above indicates, the open nature of APIs protects the investments of users in a platform or service as much as those of software developers. Twitter is a massively popular way to communicate with the world through microblogging. Some great uses of Twitter have come from add-on applications that call the Twitter API to provide additional services. Sickweather tracks Twitter and Facebook for people posting about being sick, and maps their comments so users can find out what illnesses are going around in their area.

---

[42] Available at: http://www.time.com/time/specials/packages/article/0,28804,2094921_2094923_2094924-2,00.html.

How It Works, Sickweather.[43] Flipboard lets users access all their social networks and regular news sources together. *Flipboard*, Flipboard.[44] Third-party Twitter clients, programs that display Twitter feeds in different, user-friendly ways, are especially popular. Erick Schonfeld, *The Top 21 Twitter Clients*, TechCrunch (Feb. 19, 2009).[45] There are so many Twitter-based applications and Twitter clients that Twitter's own developers could not have created and implemented all of them. In fact, Twitter has not built native apps for some hardware devices, and Twitter clients are the only Twitter apps available for users of those devices. John McDermott, *App Developers Shun Microsoft's Surface*, Ad Age (Dec. 4, 2012).[46] If Twitter were to go out of business or stop supporting these applications, many users would lose access to their favorite applications. If this happened, a new competitor could come into the market and support those applications by reimplementing the Twitter API.

Finally, a very present example involves Google's "Reader" service, which Google is shutting down effective July 1, 2013. If the Google reader API was copyrightable, users of Google Reader would be stranded. It is not, and that is one reason a new service called Feedly initially attracted over 500,000 users

[43] Available at: http://www.sickweather.com/how.
[44] Available at: http://flipboard.com.
[45] Available at: http://techcrunch.com/2009/02/19/the-top-21-twitter-clients-according-to-twitstat.
[46] Available at: http://adage.com/article/digital/app-developers-shun-microsoft-s-surface/238602/.

by offering a compatible service. Salvador Rodriguez, *Google Reader's demise means big gains for Feedly*, Los Angeles Times (March 18, 2013).[47] Two months later, Feedly's user base "swelled to seven million." David Pogue, *Google's Aggregator Gives Way to an Heir*, N.Y. Times (May 8, 2013).[48]

B. The Orphan Software Problem Disproportionately Affects the Public Sector

Government entities and non-profits are especially susceptible to the orphan programs problem since their tight budgets often force them to use outdated technology. One of our signatories, Jeremiah Flerchinger, is a developmental engineer with over eight years of service in the Department of Defense and, later, with a machine-tool company. The National Aeronautics and Space Administration (NASA) sought to repurpose old manufacturing robots for a new project, and asked Flerchinger's company to manufacture and program updated memory chips to store the robots' new instructions. Configuring firmware to put on the chips required using obsolete software that wouldn't run on modern computers. Flerchinger reimplemented the software's API, creating modern software that could fulfill the same functions and work alongside old machines that had the same API hard-coded into their electronics.

---

[47] Available at: http://www.latimes.com/business/technology/la-fi-tn-google-reader-demise-feedly-20130318,0,3173230.story.

[48] Available at: https://www.nytimes.com/2013/05/09/technology/personaltech/three-ways-feedly-outdoes-the-vanishing-google-reader.html?_r=1&.

If APIs were copyrightable, Flerchinger's company would have needed a license to reimplement that software's API. Assuming they could afford it, finding the right person to grant permission for a reimplementation would have been extremely difficult. If Flerchinger's company couldn't control its liability, they would not have been able to reimplement the API and complete their contract. Copyright on the API of obsolete software could have forced NASA to spend its limited funding on replacing its perfectly functional manufacturing robots. In these days of budget cuts and the "sequester," such unnecessary spending should be discouraged, not encouraged, by this Court.

CONCLUSION

The freedom to reimplement and extend existing APIs has undoubtedly led to robust software and hardware industries, but also to an explosion of technological advances that do more than merely increase companies' bottom lines. They open the world for the sharing of information, increased communication, and technological advances that could have never been contemplated.

The role of the courts in this progress cannot be understated. Indeed, the progress of technology flourished following the Supreme Court's 1996 affirmance of the First Circuit's holding in *Lotus v. Borland* that the menu hierarchy that controlled Lotus 1-2-3's functional capabilities was a method of

operation and thus uncopyrightable. *Lotus,* 49 F.3d at 807. Programmers and developers have relied on that ruling for the proposition that APIs, like the menu hierarchy in *Lotus,* may not be copyrighted. The *Lotus* Court's ruling thus directly led to many of the innovations highlighted by amici here. Affirmance of the District Court ruling below would ensure that the grounds for technological development remain fertile. We respectfully urge this Court to do that.

May 30, 2013

Respectfully submitted,

ELECTRONIC FRONTIER FOUNDATION

By: /s/ Julie P. Samuels

Julie P. Samuels
Michael Barclay
ELECTRONIC FRONTIER FOUNDATION
815 Eddy Street
San Francisco, CA 94109-7701
Tel: (415) 436-9333
Fax: (415) 436-9993
julie@eff.org

Jason M. Schultz
NYU Technology Law and Policy Clinic
NYU SCHOOL OF LAW
40 Washington Square South
New York, NY 10012
Tel: (212) 998-6100

Attorneys for Amici Curiae
Computer Scientists

CERTIFICATE OF SERVICE

I hereby certify that on May 30, 2013, I caused the foregoing BRIEF OF AMICI CURIAE COMPUTER SCIENTISTS IN SUPPORT OF DEFENDANT-CROSS APPELLANT AND AFFIRMANCE to be electronically filed with the Clerk of the Court using CM/ECF, which will automatically send email notification of such filing to the following counsel of record:

Christa M. Anderson
Robert A. Van Nest
Steven Hirsch
Michael Kwun
Dan Jackson
KEKER & VAN NEST, LLP
633 Battery Street
San Francisco, CA 94111

E. Joshua Rosenkranz
Mark S. Davies
Andrew D. Silverman
Kelly M. Daley
ORRICK, HERRINGTON & SUTCLIFFE LLP
51 West 52nd Street
New York, NY 10019

Dorian E. Daley
Deborah K. Miller
Matthew Sarboraria
Andrew C. Temkin
ORACLE AMERICA, INC.
500 Oracle Parkway
Redwood Shores, CA 94065

Dale M. Cendali
Diana M. Torres
Sean B. Fernandes
Joshua L. Simmons
KIRKLAND & ELLIS LLP
601 Lexington Avenue
New York, NY 10022

Annette L. Hurst
Gabriel M. Ramsey
Elizabeth C. McBride
ORRICK, HERRINGTON & SUTCLIFFE LLP
405 Howard Street
San Francisco, CA 94105-2669

Susan M. Davies
KIRKLAND & ELLIS LLP
655 15th Street, N.W.
Washington, DC 20005

Michael A. Jacobs
Kenneth A. Kuwayti
MORRISON & FOERSTER LLP
425 Market Street
San Francisco, CA 94105

Bruce William Baber
KING & SPALDING LLP
1180 Peachtree Street, NE, Suite 2800
Atlanta, GA 30309

Daryl Joseffer
KING & SPALDING LLP
1700 Pennsylvania Avenue, N.W.
Washington, DC 20006

Ian Charles Ballon
GREENBERG TRAURIG LLP
2450 Colorado Avenue
19th Floor, Suite 400E
Santa Monica, CA 90404

Heather Meeker
GREENBERG TRAURIG LLP
1900 University Avenue, 5th Floor
East Palo Alto, CA 94303

Renny F. Hwang
GOOGLE INC.
1600 Amphitheatre Parkway
Mountainview, CA 94043

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct. Executed on May 30, 2013 in San Francisco, California.

/s/ Julie P. Samuels
Julie P. Samuels

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE REQUIREMENTS PURSUANT TO FED. R. APP. P. 32(a)(7)(C)

I hereby certify as follows:

1. The foregoing Brief of Amicus Curiae of Computer Scientists In Support of Defendant-Cross Appellant and Affirmance complies with the type-volume limitation of Fed. R. App. P. 32(a)(7)(B). The brief is printed in proportionally spaced 14-point type, and there are 6,767 words in the brief according to the word count of the word-processing system used to prepare the brief (excluding the parts of the brief exempted by Fed. R. App. P. 32(a)(7)(B)(iii), that is, the tables of contents and citations, and certificates of counsel, and by Fed. Cir. R. 32(b), that is, the certificate of interest, the statement of related cases, and the addendum in an initial brief of an appellant).

2. The brief complies with the typeface requirements of Federal Rule of Appellate Procedure 32(a)(5), and with the type style requirements of Federal Rule of Appellate Procedure 32(a)(6). The brief has been prepared in a proportionally spaced typeface using Microsoft® Word for Mac 2011 in 14-point Times New Roman font.

May 30, 2013

/s/ Julie P. Samuels
Julie P. Samuels
Attorneys for Amici Curiae Computer Scientists