No. 14-410

IN THE

Supreme Court of the United States

GOOGLE INC.,

Petitioner,

v.

ORACLE AMERICA, INC.,

Respondent.

ON PETITION FOR A WRIT OF CERTIORARI TO THE COURT OF APPEALS FOR THE FEDERAL CIRCUIT

BRIEF OF A*MICI CURIAE* COMPUTER SCIENTISTS IN SUPPORT OF PETITIONER

CORYNNE MCSHERRY
Counsel of Record
MICHAEL BARCLAY
ELECTRONIC FRONTIER FOUNDATION
815 Eddy Street
San Francisco, CA 94109
(415) 436-9333
corynne@eff.org

Attorneys for Amici Curiae Computer Scientists

November 7, 2014

255819

COUNSEL PRESS

(800) 274-3321 • (800) 359-6859

TABLE OF CONTENTS

Table of Contents

Table of Contents

TABLE OF CITED AUTHORITIES

Page

Federal Cases

Statutes

Other Authorities

Cited Authorities

Page

Page

STATEMENT OF IDENTITY AND INTEREST OF *AMICI CURIAE*[1]

Amici are computer scientists who believe the Federal Circuit's ruling will stifle progress in their field. Amici include many of the pioneers of modern computing, who have invented or contributed to the authorship of computing technologies that have been essential to the development of the Internet, personal computing, and the explosion of technology industries in the past four decades. They have joined this brief because they believe the Federal Circuit overextended copyright coverage in a manner that is irreconcilable with both the purposes of copyright law and the reality of computer science.

As computer scientists, amici have relied on decades of custom and precedent that recognized the open nature of APIs and permitted them to reimplement those APIs to create new software. Amici depend on and expect the inherent openness of APIs to support innovation by

---

1. No counsel for a party authored this brief in whole or in part, and no such counsel or party made a monetary contribution intended to fund the preparation or submission of this brief. No person other than the amici curiae, or their counsel, made a monetary contribution intended to fund its preparation or submission. In an abundance of caution and for the sake of transparency, counsel state that Petitioner, Google Inc., has made contributions to the Electronic Frontier Foundation; such funds have been allotted to support specific projects, but not this brief.

Pursuant to Supreme Court Rule 37.2(a), amici curiae provided at least ten days' notice of their intent to file this brief to counsel of record for all parties. All parties have consented to the filing of this brief.

Web sites cited in this brief were last visited on October 28, 2014.

startups and incumbents alike. By reversing or splitting with decades of existing precedent, the Federal Circuit has dangerously undermined those settled expectations.

A list of amici and their short biographies are contained in the appendix.

INTRODUCTION AND SUMMARY OF ARGUMENT

For decades, computer scientists have relied on the open nature of APIs to enable rapid innovation in computer technology. For decades, circuit courts have supported that reliance, concluding that Section 102(b) of 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. The district court correctly followed that precedent and rejected Oracle's claim that the Java APIs could be copyrightable. Sadly, the Federal Circuit chose to split with the other circuits and reverse the district court. That decision upended decades of industry practice and threatens the basic principles upon which our technology sector was built.

Not surprisingly, the Federal Circuit's decision has been harshly criticized. As many commentators have noted, if the Federal Circuit's view had been accepted at the birth of modern computing, many important technologies would never have existed, and/or succeeded. 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. And open APIs

were 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.

Today, open, uncopyrightable APIs continue to spur the creation and adoption of new technologies. When programmers can freely reimplement or reverse engineer an API without obtaining a costly license or risking a lawsuit, they can create compatible software that the interface's original creator might never have envisioned or had the resources to develop. Moreover, compatible APIs help 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 and programs behind when they switch to a new service.

The freedom to reimplement APIs also helps developers rescue "orphan" software or data—systems that are no longer supported by their creators. When a company stops supporting a computer platform or service, the ability to freely reimplement APIs protects the communities that rely on that software. 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.

The Federal Circuit's decision poses a significant threat to the technology sector and to the public. If it is allowed to stand, Oracle and others will have an unprecedented and dangerous power over the future of

innovation. API creators would have veto rights 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 industry to flourish, and choke off many of the system's benefits to consumers.

To forestall this threat, we urge this Court to grant the petition, reverse the Federal Circuit, and reinstate the district court's decision.

ARGUMENT

I. The Federal Circuit's Decision Has Disrupted Decades of Settled Expectations That APIs Are Not Copyrightable

The Federal Circuit's decision shocked many in the computer science community, including the undersigned. The Court's decision could undermine decades of progress and innovation premised on the open nature of computer interfaces. Indeed, the free and open use of APIs has been both routine and essential in the computer industry since its beginning, and that use depended, in turn, on the sensible assumption that APIs and other interfaces were uncopyrightable.

That assumption was well-founded. It was based, in large part, on this Court's own recognition, in 1996, that menu hierarchies that control functional capabilities are a method of operation, and thus uncopyrightable under 17 U.S.C. § 102(b). *Lotus Dev. Corp. v. Borland Int'l, Inc.*, 49 F.3d 807, 815 (1st Cir. 1995), *aff'd by an equally divided Court*, 516 U.S. 233 (1996). Programmers and developers

relied on that ruling for the proposition that APIs, like the menu hierarchy in *Lotus*, may not be copyrighted under § 102(b). Ninth Circuit law—which should have been controlling in this case—was the same: the "functional requirements for compatibility" between computer programs "are not protected by copyright." *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)).

The Federal Circuit's decision marked a dramatic and disturbing departure from this precedent. Noting the industry's long-standing reliance on cases such as *Lotus*, *Sega v. Accolade*, and *Computer Associates v. Altai*, one commenter explained why the Federal Circuit decision was so harmful:

> By ruling that interoperability is relevant only to fair use, and not to protectability, Judge O'Malley would require every developer to perform a fair use analysis before developing an interoperable product. This would place U.S. programmers at a competitive disadvantage to developers in other jurisdictions that recognized that copyright does not protect program elements necessary for interoperability.

Jonathan Band, *The Federal Circuit's Poorly Reasoned Decision in Oracle v. Google*, Disco (May 9, 2014).[2] Others

2. Available at: http://www.project-disco.org/intellectual-property/050914-the-federal-circuits-poorly-reasoned-decision-in-oracle-v-google.

pointed out fundamental flaws in the Federal Circuit's reasoning, suggesting it reflected a "fundamental lack of understanding of how software works." Timothy B. Lee, *The Court That Created the Patent Troll Mess Is Screwing Up Copyright Too*, Vox (May 9, 2014).[3] One observer noted that the court seemed "to think that because the API is 'big' it must therefore be copyrightable as a 'literary work' even though—as [Judge] Alsup rightly pointed out—it's nothing more than a 'system or method of operation' which 102(b) clearly states is uncopyrightable." Mike Masnick, *Appeals Court Doesn't Understand the Difference Between Software and an API; Declares APIs Copyrightable*, Techdirt (May 9, 2014).[4]

Thus, the Federal Circuit's opinion is widely regarded as both insupportable and dangerous. In what follows, amici illustrate that danger by highlighting a few of the technological developments that would not have occurred if the Federal Circuit's misguided view had prevailed at the birth of modern computing.

II. Uncopyrightable Interfaces Were Essential to the Development of Modern Computers and the Internet

A. The BIOS of the Original IBM-Compatible PC

APIs free from copyright restrictions were essential to the emergence of home computing. In 1981, IBM released

---

3. Available at: http://www.vox.com/2014/5/9/5699960/this-court-decision-is-a-disaster-for-the-software-industry.

4. Available at: https://www.techdirt.com/articles/20140509/10133727178/appeals-court-doesnt-understand-difference-between-software-api-declares-apis-copyrightable.shtml.

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 had an open design. Thanks to that design, add-on innovation in PC software and hardware peripherals flourished. *Id.* at 28–29. To use IBM-exclusive software like the popular spreadsheet program Lotus 1-2-3, however, users initially had to buy IBM computers. *Id.* Although other 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[5] compatibility as well. Thus, the IBM model was 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).

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 52-53. 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*

5. 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 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.

240–41 (2008). The clean teams created new software from scratch using the interface specifications needed to interact successfully with the 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.[6] 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.

Key to that development was the fact that IBM owned the copyright on the BIOS source code, but could not claim a monopoly on the system of commands the operating system used to communicate with that code. Thus, Compaq and Phoenix were entitled to reimplement the BIOS interface as long as they did not copy any of IBM's code. *Cf. Lotus*, 49 F.3d at 810 (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). If the law had been otherwise, IBM's ownership of the BIOS code would have given it the ability to stifle competitive innovation, to the detriment of the public.

---

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

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, UNIX. Developed by amicus Ken Thompson, Dennis Ritchie, and other computer scientists at AT&T Bell labs and launched in 1969, UNIX is widely regarded as the first modern operating system. Heather J. Meeker, *The Open Source Alternative* at 3–4 (2008). It ran on large mainframe and minicomputers owned by corporations, universities, and the government. *Id.*

At the time AT&T developed UNIX, however, 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.[7] Thus, to comply with the decree, AT&T licensed UNIX source code to any interested party for a nominal fee. Meeker, *supra*, at 4. Thanks in part to that open license, 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 developers dissatisfied with available operating

7. *See* http://www.corp.att.com/history/milestones.html.

systems such as MS-DOS, Windows, and Apple's system, along with UNIX users, reimplemented the UNIX API to run on a PC.[8] Meeker, *supra*, at 6.

Because the API was open, it took a minimal amount of work to make pre-existing software run on subsequent systems. For example, some developers 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).[9] 37% of Web servers run on Linux.[10] Countless Internet-based services from Facebook to ATMs rely on Linux-based high-speed networking systems. Vaughan-Nichols, *supra*.

---

8. 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).

9. Available at: http://www.linuxplanet.com/linuxplanet/reports/6671/1.

10. *See Usage of operating systems for websites*, W3Techs, available at: http://w3techs.com/technologies/overview/operating_system/all; *Usage statistics and market share of Unix for websites*, W3Techs, available at: http://w3techs.com/technologies/details/os-unix/all/all.

The varied implementations of the UNIX API are textbook examples of the importance of § 102(b) to innovation and competition. Thanks to the prevailing interpretation of that provision, innovators could provide their 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. The evolution of "C" is a textbook example. Dennis Ritchie, one of the computer scientists who invented UNIX, also co-invented a new language, called "C," in which to code it. P.J. Plauger, *The Standard C Library* 3 (1991). 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. Once programmers learn C, they can write code that will run on any operating system that can provide a reimplementation of the C standard library.

Today, those operating systems are legion. 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.[11] Google's

11. Available at: http://msdn.microsoft.com/en-us/library/abx4dbyh(v=vs.80).aspx.

reimplementation of the same for Android is called Bionic. *The Native Android API*, Mobile Pearls.[12] Another significant reimplementation was the GNU C Library, which was essential to the GNU Project's effort to create a free UNIX-compatible operating system. *The GNU C Library (glibc)*, The GNU Project.[13]

Limiting the ability to reimplement the C Standard Library would have 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.[14]

Thus, 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. Old programs could become 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.

---

12. Available at: http://mobilepearls.com/labs/native-android-api/.

13. Available at: http://www.gnu.org/software/libc/.

14. Very briefly and simply stated, computer languages provide "libraries," which are collections of reusable functionality (*e.g.* math, text formatting). Libraries shorten programs and reduce errors by eliminating the need for programmers to "reinvent the wheel." They also offer essential functionality that couldn't be directly implemented in the language (e.g. Input/Output). Libraries include subroutines; in some languages, they are grouped into classes and packages.

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.

Consider, for example, the many reimplementations of the Berkeley Systems Distribution (BSD) "sockets" API. That API, designed to help computers connect to the Internet, was 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).[15] 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 applications to connect to the Internet via an implementation of the BSD sockets API. *See, e.g.*, *About Berkeley Sockets and Winsock*, VMware.[16]

---

15. Available at: http://h71000.www7.hp.com/openvms/journal/v3/tcpip.pdf.

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

It is the de facto industry standard. *See, e.g.,* Johnson M. Hart, *Windows System Programming*, ch. 12 (4th ed. 2010). Application developers only have to write the networking sections of their program once for it to function on almost every operating system. OS developers can redesign the implementation of the API to improve its efficiency, knowing that consumers will be able to keep using all their programs. And none of this would have occurred if developers had to seek out a license for every reimplementation.

Consider also the history of network protocols. Network protocols are computer interfaces very similar to APIs. While APIs provide the framework for communication among or within programs, network protocols provide the underlying framework for communication among computers.

One of the most important of these is the hypertext transfer protocol (HTTP). Larry L. Petersen & Bruce S. Davie, *Computer Networks: A Systems Approach* 640–42 (2d ed. 2000). Whenever an 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;[17] Roy Fielding, *et al.*, *RCFC 2616: Hypertext Transfer Protocol—HTTP/1.1*, Internet Engineering Task Force (June 1999).[18] If interface specifications were copyrightable, Berners-Lee could have used copyright to determine which software and hardware manufacturers could use the Web. But as an open system, the Web became universal.

Microsoft's Server Message Block (SMB) system offers a cautionary tale of that kind of fragmentation—and the importance of keeping APIs free of copyright restriction. 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. 8, 2002).[19] 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).[20] 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).[21]

---

17. Available at: http://www.w3.org/People/Berners-Lee/Overview.html#Bio.

18. Available at: http://tools.ietf.org/html/rfc2616.

19. Available at: http://www.samba.org/cifs/docs/what-is-smb.html.

20. Available at: http://www.samba.org/ftp/tridge/misc/french_cafe.txt.

21. Available at: http://www.samba.org/samba/docs/SambaIntro.html.

Beginning in 1991, however, amicus 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). That reimplementation would never have happened if Tridgell had had to seek a license from Microsoft first.

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. That, in turn, has helped fuel an explosion of innovation. Indeed, this is one area where the purposes of copyright are best served by exclusion from copyright protection.

E. Treating APIs as Copyrightable Would Undermine 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 PC 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.[22]

22. Available at: http://aws.amazon.com/what-is-aws.

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).[23] Virtual machines "call" (or invoke) 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.*

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).[24] For example, most providers rely on Amazon's cloud services API to allow users to control and operate the cloud computers that they rent. Because the API is not restricted by copyright, companies like CloudStack and Eucalyptus can compete with Amazon to provide the best implementation of that API. Businesses that employ cloud services can also write or commission their own proprietary software to perform operations on cloud servers. *Business Applications*, Amazon Web Services.[25] In addition, since major cloud service providers like Amazon, Eucalyptus, and CloudStack use the same

---

23. Available at: http://searchservervirtualization.techtarget.com/definition/virtual-machine.

24. Available at: http://www.eucalyptus.com/news/amazon-web-services-and-eucalyptus-partner.

25. Available at: http://aws.amazon.com/business-applications.

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-Source Cloud Wars*, ZDNet (Apr. 12, 2012).[26] Software developers can write programs capable of interacting with the above three cloud services, creating new ways for users to access and manipulate information spread out across the Internet.

By contrast, 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. Given the cost and disruption of doing so, few businesses would be willing to leave their cloud service provider, meaning late entrants in the cloud service market would be hard-pressed to build a customer base. The ultimate result: less choice, less innovation.

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 and other APIs. 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. It did not, which meant the BIOS API became a kind of shared resource. Using

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

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

III. Uncopyrightable Interfaces Spur the Creation of Software That Otherwise Would Not Be Written

When programmers can freely employ any interface without obtaining a costly license or risking a lawsuit, they can create compatible software that the interface's original creator might never have envisioned or had the resources to develop. Copyrightable APIs would discourage this innovation by creating potential liability for the mere act of writing a compatible program.

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 benefit users as well as developers. In the context of supercomputers, for example, it is often necessary to reimplement an API in order to make new hardware compatible with existing software. Supercomputers typically have unusual, custom-built hardware reflecting both their purpose and the

state of the art in computer design at the time of their manufacture. 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 tailored 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, 2005);[27] IBM, *Basic Linear Algebra Subprograms Library Programmer's Guide and API Reference* (2008).[28]

A shared API is particularly important for supercomputers, because time on these machines is often limited. *See e.g.*, *Scheduling Policies and Limits*, Ohio Supercomputer Center;[29] *PBS Information for Labs and the Lab Queue*, Minnesota Supercomputing Institute.[30] 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. Therefore, programs 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

27. Available at: http://www.netlib.org/blas/faq.html.

28. Available at: http://webpages.uncc.edu/~apanday/documents/BLAS_Prog_Guide_API_v3.0.0.3.pdf.

29. Available at: https://www.osc.edu/supercomputing/batch-processing-at-osc/scheduling-policies-and-limits.

30. Available at: https://www.msi.umn.edu/labs/pbs.

that makes that possible, as long as both machines have valid implementations of the same interfaces.

If every supercomputer vendor had its own proprietary API, then research groups also would find themselves "locked in" to the vendor they worked with at the start of their project. A research project's codebase (all the code they've written) is a significant investment, tied to the APIs chosen at the start. *See* National Research Council, *supra*, at 21. Moving to an incompatible new supercomputer could mean losing that investment. *Id.* Thus, 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).[31] 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

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

API, figure out its rules, and write extra code (called a "client wrapper") to make it easier for outsiders to use. *Id.* If the unofficial API is posted online, anyone can write their own software compatible with the website. *Id.*

For instance, the programmer 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).[32] 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. Instagram recognized the demand for Marohnić's API and 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).[33] Today, organizations of all stripes use the Instagram API to feature their members' photography and share photos with customers and fans. *Getting Started with the Instagram API*, Instagram Help Center.[34]

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

---

32. Available at: http://blog.programmableweb.com/2010/12/15/the-full-featured-unpublished-instagram-api.

33. Available at: http://blog.programmableweb.com/2011/01/12/instagram-shuts-down-third-party-developers-plans-official-api.

34. Available at: https://web.archive.org/web/20121112035903/http://help.instagram.com/customer/portal/articles/95808-getting-started-with-the-instagram-api (archived on Nov. 12, 2012).

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

IV. 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 for many reasons, and rights in software may change hands many times when startups are acquired or divisions of companies spin off or shut down. Over time, "orphan" software often becomes incompatible with modern computers and other software, particularly as platforms change. For owners of that software, reimplementing the software's API may be the only realistic way to reclaim the time and resources they have invested in it. But the copyright owner may no longer be identifiable, much less available to authorize the creation of derivative works.

Section 102(b) helps solve this problem. When a copyright owner goes missing, it is difficult 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 easily 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 § 102(b)'s purposes.

A. Once and Future Orphans: Some Examples

The history of the social bookmarking site Delicious is a case in point. 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).[35] 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.[36] Yahoo! bought Delicious and slowly phased out its development, losing Delicious users along the way. Bobbie Johnson, *supra.* As the size of the community diminished, so did the usefulness of Delicious. Kristina Dell, *Entrepreneurs Who Go It Alone—By Choice*, Time (Oct. 24, 2011).[37] 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

---

35. Available at: http://gigaom.com/2011/09/28/oh-delicious-where-did-it-all-go-so-wrong.

36. Available at: http://pinboard.in/tour#api.

37. Available at: http://content.time.com/time/specials/packages/article/0,28804,2094921_2094923_2094924,00.html.

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. Some of the best uses of Twitter, a massively popular way to communicate with the world, have come from add-on applications that interoperate with the Twitter API to provide additional services that dramatically increases its value to users. For example, 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. *How It Works*, Sickweather.[38] Flipboard lets users access all their social networks and regular news sources together. *Flipboard*, Flipboard.[39] 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).[40]

Left to its own devices, Twitter probably could not have created and implemented all of these services. In fact, Twitter hasn't built native apps for some hardware

38. Available at: http://www.sickweather.com/how.

39. Available at: http://flipboard.com.

40. Available at: http://techcrunch.com/2009/02/19/the-top-21-twitter-clients-according-to-twitstat.

devices, such as Microsoft's Surface tablet. John McDermott, *App Developers Shun Microsoft's Surface*, Ad Age (Dec. 4, 2012).[41] If Twitter were to go out of business or stop supporting its APIs, 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.

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. Amicus Jeremiah Flerchinger is an electrical engineer with over ten years of service in the Department of Defense, after previous experience with a machine-tool company. When the National Aeronautics and Space Administration (NASA) sought to repurpose old manufacturing robots for a new project, they 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.

If APIs were copyrightable, Flerchinger's company would have needed a license to reimplement that software's API. Assuming it could afford it, finding the right person

41. Available at: http://adage.com/article/digital/app-developers-shun-microsoft-s-surface/238602/.

to grant permission for a reimplementation would have been extremely difficult. If the company couldn't control its liability, it would not have been able to reimplement the API and complete the 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, such unnecessary spending should be discouraged, not encouraged, by the courts.

CONCLUSION

The Court should grant the petition for a writ of certiorari.

Respectfully submitted,

Corynne McSherry
Counsel of Record
Michael Barclay
Electronic Frontier Foundation
815 Eddy Street
San Francisco, CA 94109
(415) 436-9333
corynne@eff.org

Attorneys for Amici Curiae
Computer Scientists

November 7, 2014

APPENDIX — LIST OF *AMICI CURIAE*

(In alphabetical order)

Unless otherwise indicated, amici are signing this brief on their own individual behalf and not on behalf of the companies or organizations with whom they are affiliated. Those affiliations are only for identification. This includes those amici indicated by an asterisk (*), who are presently Google employees, consultants, and/or directors. Those amici are signing this brief as individual computer scientists whose work in the field long preceded their affiliation with Google. They are not signing this brief on behalf of Google or at Google's request.

1. **Harold Abelson.*** Dr. Harold "Hal" Abelson is a Professor of Electrical Engineering and Computer Science at MIT, a fellow of the IEEE, and a founding director of both Creative Commons and Public Knowledge. He directed the first implementation of the Logo computing language for the Apple II, which made the language widely available on personal computers beginning in 1981, and published a popular book on Logo in 1982. Abelson co-developed MIT's introductory computer science subject, which included innovative advances in curricula designed for students pursuing different kinds of computing expertise. These curricula had a worldwide impact on university computer science education. Notable awards include the Bose Award (MIT School of Engineering, 1992), the Taylor L. Booth Education Award (IEEE-CS, 1995), and the SIGCSE 2012 Outstanding Contribution to Computer Science Education (ACM, 2012). Abelson

holds an A.B. from Princeton University and a Ph.D. in mathematics from MIT.

2. **Alfred V. Aho.** Alfred V. Aho is a Professor of Computer Science at Columbia University, where he served as department chair. His research interests include programming languages, compilers, algorithms, and quantum computing. He was Vice President of Computing Sciences Research at Lucent Technologies, and a researcher at Bellcore and Bell Labs, where he devised efficient pattern matching algorithms which he implemented when writing the Unix tools egrep and fgrep. He developed the algorithms at the heart of the compiler tools yacc and lex. With Peter Weinberger and Brian Kernighan, he designed and implemented the AWK programming language. He is the coauthor of many influential textbooks. He was awarded the IEEE's John von Neumann Medal and two honorary doctorates. He is a member of the National Academy of Engineering and a fellow of the American Academy of Arts and Sciences, the American Association for the Advancement of Science, Bell Labs, ACM, and IEEE. He has twice served as chair of the Advisory Committee for the Computer and Information Science and Engineering Directorate of the National Science Foundation. He holds a B.A.Sc. from the University of Toronto and a Ph.D. from Princeton.

3. **Tom Ball.*** Tom Ball is a Senior Software Engineer at Google, working on Java-based developer tools. He was previously a Distinguished Engineer at

Sun Microsystems, and a member of the JDK team that first released Java publicly. He wrote the first Java debugger (jdb), was a member of the AWT and Swing teams, and developed the Jackpot automated refactoring tool designed by James Gosling. His current project is J2ObjC (http://j2objc.org), an open-source tool that converts Java source to Objective-C for use by iOS applications (which cannot run Java).

4. **John Perry Barlow.** John Perry Barlow is a former Wyoming rancher and Grateful Dead lyricist. A co-founder of Electronic Frontier Foundation (EFF), he was the first to apply the term cyberspace to the "place" it presently describes. He has written for a diversity of publications, including Communications of the ACM, Mondo 2000, The New York Times, and Time. He has been on the masthead of Wired magazine since it was founded. His piece on the future of copyright, "The Economy of Ideas," is taught in many law schools, and his "Declaration of the Independence of Cyberspace" is posted on thousands of websites. In 1997, he was a Fellow at Harvard's Institute of Politics and has been, since 1998, a Berkman Fellow at the Harvard Law School. John works actively with several consulting groups, including Diamond Technology Partners, Vanguard, and Global Business Network. In 1999, FutureBanker Magazine named him "One of the 25 Most Influential People in Financial Services." He writes, speaks, and consults on a broad variety of subjects, particularly the digital economy.

5. **Brian Behlendorf.** Brian Behlendorf is a Managing Director at Mithril Capital Management. He is also on the Boards of Directors at the Electronic Frontier Foundation, the Mozilla Foundation, and Benetech, three non-profits in the technology field. He has served as an advisor to the Office of Science and Technology Policy at the White House, as well as the Department of Health and Human Services; and as Chief Technology Office at the World Economic Forum. He was also a founding developer of the Apache Web Server, and served as the first President of the Apache Software Foundation.

6. **Richard A. Belgard**. Rich Belgard has been active in the computer industry for over 40 years. Over these years, he designed and managed the development of computer architectures at Burroughs, Data General, Tandem Computers and Rational Software, including hardware, software and microarchitecture. He is co-inventor on 18 patents and sole inventor on 7 additional patents. Rich is the past Chairman and Vice-Chairman of the Association for Computing Machinery (ACM)'s Special Interest Group on Microarchitectures, and former Vice-Chair of the Institute of Electrical and Electronic Engineers (IEEE) Technical Committee on Microprogramming and Microarchitectures. Rich is currently Awards Chairman for the IEEE Computer Society. Rich is an IEEE Fellow.

7. **Jon Bentley.** Jon Bentley is a Distinguished Member of Technical Staff at Avaya Labs Research. His research interests include programming techniques,

algorithm design, and the design of software tools and interfaces. He has written three books on programming and over a hundred articles on a variety of topics, ranging from the theory of algorithms to software engineering. He received a B.S. from Stanford in 1974 and an M.S. and Ph.D. from the University of North Carolina in 1976, then taught Computer Science at Carnegie Mellon for six years. He joined Bell Labs Research in 1982, and retired in 2001 to join Avaya. He has been a visiting faculty member at West Point and Princeton, and has been a member of teams that have shipped software tools, telephone switches, telephones and web services. In March 2000 he received the Dr. Dobb's Excellence in Programming Award for advancing the craft of computer programming.

8. **Matthew Bishop**. Matthew Bishop received his Ph.D. in computer science from Purdue University, where he specialized in computer security, in 1984. He is on the faculty at the Department of Computer Science at the University of California at Davis. His main research area is the analysis of vulnerabilities in computer systems, including modeling them, building tools to detect vulnerabilities, and ameliorating or eliminating them. Currently, he has research projects involving data sanitization, modeling election processes, and examining metrics for evaluating network attack detection mechanisms; he is also looking at the "insider" problem. He has been active in the area of UNIX security since 1979, and has presented tutorials at SANS, USENIX, and other conferences. He also

has done work on electronic voting, and was one of the two principle investigators of the California Top-to-Bottom Review, which performed a technical review of all electronic voting systems certified for use in the State of California. His textbook, Computer Security: Art and Science, was published in December 2002 by Addison-Wesley Professional.

9. **Joshua Bloch**. Joshua Bloch is an expert on API design, with over a quarter century of experience. He was Chief Java Architect at Google, a Distinguished Engineer at Sun Microsystems, and a Senior Systems Designer at Transarc Corporation. He led the design and implementation of numerous Java APIs and language features, including the award-winning Java Collections Framework. He is the author of several books, including the bestselling, Jolt Award-winning *Effective Java* (Addison-Wesley, 2001; Second Edition, 2008), the de facto standard guide to Java best practices. He served on the National Academies CSTB Certifiably Dependable Software Committee. He holds a B.S. from Columbia and a Ph.D. in Computer Science from Carnegie Mellon University.

10. **Dan Boneh**. Dan Boneh is a Professor of Computer Science at Stanford University, where he heads the applied cryptography group. Dr. Boneh's research focuses on applications of cryptography to computer security. His work includes cryptosystems with novel properties, security for mobile devices, web security, and cryptanalysis. He is the author of over a hundred publications in the field and is a recipient of the 2013

Gödel prize, the Packard Award, the Alfred P. Sloan Award, the RSA award in mathematics and five best paper awards. In 2011 Dr. Boneh received the Ishii award for industry education innovation.

11. **Gilad Bracha.*** Gilad Bracha is the creator of the Newspeak programming language and a software engineer at Google where he works on Dart. Previously, he was a VP at SAP Labs, a Distinguished Engineer at Cadence, and a Computational Theologist and Distinguished Engineer at Sun. He is co-author of the Java Language Specification, and a researcher in the area of object-oriented programming languages. Prior to joining Sun, he worked on Strongtalk, the Animorphic Smalltalk System. He received a B.Sc. in Mathematics and Computer Science from Ben Gurion University in Israel and a Ph.D. in Computer Science from the University of Utah.

12. **Eric Brewer.*** Eric Brewer pioneered early "cloud" computing starting in the 1990s with research on large-scale services implemented on clusters of commodity servers, for which he was elected to the National Academy of Engineering. In 1996, Brewer co-founded Inktomi Corporation, an early search engine that also influenced the modern Internet architecture. He formulated the CAP theorem, one the tenets of modern distributed systems. In 2000, working with President Clinton, he led the development of usa.gov, the primary federal portal. He is a tenured professor in the Computer Science department at UC Berkeley, but is currently on leave

at Google as VP, Infrastructure. Brewer received a BS in EECS from UC Berkeley, and an M.S. and Ph.D. from MIT.

13. **Frederick Brooks.** Brooks is Kenan Professor of Computer Science at UNC-Chapel Hill. As Corporate Project Manager for IBM's System/360 (mainframe) computer family hardware and the Operating System/360 software, he in 1964 switched the standard computer byte size from 6 to 8 bits. He was an architect of the Stretch and Harvest supercomputers. He founded UNC's Computer Science Department. He's researched computer architecture, software engineering, the design process, and graphics virtual environments. He wrote *The Mythical Man-Month*, *The Design of Design,* and with G.A. Blaauw, *Computer Architecture*. Honors include the National Medal of Technology, the ACM Turing award, the National Academies of Engineering and Science, and British and Dutch academies.

14. **Rick Cattell.** R. G. G. "Rick" Cattell is an independent consultant in database systems. He previously worked as a Distinguished Engineer at Sun Microsystems. Dr. Cattell served for 20 years at Sun Microsystems in management and senior technical roles, and for 10 years in research at Xerox PARC and Carnegie Mellon University. He is best known for his contributions in database systems and middleware, including database scalability, Enterprise Java, object/relational mapping, object-oriented databases, and database interfaces. At Sun he instigated Enterprise Java, JDBC, Java

DB, and Java Blend, and contributed to many Java APIs and products. He previously developed Xerox PARC's Cedar DBMS, Sun's Simplify database GUI, and SunSoft's CORBA-database integration. He is a co-founder of SQL Access (predecessor to ODBC), founder and chair of the Object Data Management Group (ODMG), author of the world's first monograph on object/relational and object databases, recipient of the ACM Outstanding Ph.D. Dissertation Award, and an ACM Fellow.

15. **Vinton G. Cerf.*** Vinton G. "Vint" Cerf is vice president and Chief Internet Evangelist for Google, where he contributes to global policy development and the continued spread of the Internet. Widely known as one of the "Fathers of the Internet," Cerf is the co-designer of the TCP/IP protocols and the architecture of the Internet. He has served in executive positions at MCI, the Corporation for National Research Initiatives, the Defense Advanced Research Projects Agency, and on the faculty of Stanford University. Cerf served as chairman of the board of the Internet Corporation for Assigned Names and Numbers (ICANN) from 2000-2007. Cerf is a Fellow of the IEEE, ACM, and AAAS, the American Academy of Arts and Sciences, the International Engineering Consortium, the Computer History Museum, and is a member of the National Academy of Engineering. He is a former President of the ACM and Founding President of the Internet Society. President Obama appointed him to the National Science Board in 2012. Cerf is a recipient of numerous awards and

commendations in connection with his work on the Internet, including the US Presidential Medal of Freedom, US National Medal of Technology, the Queen Elizabeth Prize for Engineering, the ACM Turing Award, Officer of the Legion d'Honneur and 21 honorary degrees. In December 1994, People magazine identified Cerf as one of that year's "25 Most Intriguing People." Cerf holds a B.S. from Stanford, and an M.S. and Ph.D. from UCLA.

16. **William Cook.** William Cook is an Associate Professor in the Department of Computer Sciences at the University of Texas at Austin. His research is focused on object-oriented programming, programming languages, modeling languages, and the interface between programming languages and databases. Prior to joining UT in 2003, Dr. Cook was Chief Technology Officer and co-founder of Allegis Corporation. He was chief architect for several award-winning products, including the eBusiness Suite at Allegis, the Writer's Solution for Prentice Hall, and the AppleScript language at Apple Computer. At HP Labs his research focused on the foundations of object-oriented languages, including formal models of mixins, inheritance, and typed models of object-oriented languages. He completed his Ph.D. in Computer Science at Brown University in 1989.

17. **Ward Cunningham.** Ward Cunningham has worked for and consulted to daring startups and huge corporations. He has served as CTO, Director, Fellow, Principal Engineer and Inventor. He is best

known for creating wiki. He leads an open-source project rebuilding wiki to solve more complex sharing situations addressing some of society's toughest problems. Ward founded movements in object-oriented, agile software, extreme programming and pattern languages. Ward lives in Portland, Oregon and works for New Relic, Inc.

18. **Mark Davis.*** Dr. Mark Davis has been the Chief Internationalization Architect at Google since 2006, focusing on effective and secure use of Unicode, software internationalization libraries, and related areas. Dr. Davis is also the co-founder and has been president of the Unicode Consortium since its inception in 1991, and is a key technical contributor to the Unicode specifications. In 2003, he founded the Unicode Common Locale Data Repository (CLDR) project, the standard repository for locale data worldwide. He is co-author of BCP 47 ("Tags for Identifying Languages"), used to identify human languages in all XML and HTML documents, and in all modern programming libraries. Mark provided the original architecture of ICU, the premier Unicode software internationalization library, and the Java internationalization libraries. At IBM, he was Chief Software Globalization Architect. At Taligent, he was manager and architect for the international frameworks. At Apple, he co-authored the first Macintosh system to support Japanese (KanjiTalk), and authored the first Macintosh Arabic and Hebrew systems. Mark holds a Ph.D. from Stanford University and a B.A. from the University of California, Irvine.

19. **Jeffrey Dean.*** Jeff Dean joined Google in 1999 and is currently a Senior Fellow in Google's Knowledge Group. He has co-designed/implemented five generations of Google's crawling, indexing, and query serving systems, and co-designed/implemented major pieces of Google's initial advertising and AdSense for Content systems. He is also a co-designer and co-implementor of Google's distributed computing infrastructure, including the MapReduce, BigTable and Spanner systems, protocol buffers, LevelDB, systems infrastructure for statistical machine translation, and a variety of internal and external libraries and developer tools. Prior to joining Google, Jeff did computer systems research at Digital Equipment Corporation's Western Research Lab. Jeff has also worked for both the Centers for Disease Control and the World Health Organization, designing computer software for epidemiology and for statistical analysis of the HIV/AIDS pandemic. He is a Fellow of the ACM and the AAAS, a member of the U.S. National Academy of Engineering, and a recipient of the Mark Weiser Award and the ACM-Infosys Foundation Award in the Computing Sciences. Jeff holds a B.S., *summa cum laude*, in computer science and economics from the University of Minnesota, and a M.S. and Ph.D. in computer science from the University of Washington.

20. **L Peter Deutsch.** Dr. L Peter Deutsch received a Ph.D. in Computer Science from U.C. Berkeley in 1973. Subsequently at Xerox PARC, he helped develop the Interlisp-D, Cedar Mesa, and Smalltalk-80

programming systems. Deutsch's work on Smalltalk implementation, among other innovations, was an important contributor to the just-in-time compilation technology now used widely to dramatically improve the performance of Java and JavaScript implementations. He is also the author of a number of RFCs and of the The Eight Fallacies of Distributed Computing, and originated the Deutsch limit adage about visual programming languages. From 1986 to 1991, as Chief Scientist at ParcPlace Systems, he developed cross-platform JIT technology. From 1986 to 2003, dba Aladdin Enterprises, he was the creator of Ghostscript, an Open Source implementation of the PostScript language. In 1993, he was a co-recipient of the ACM Software System Award, and was also named a Distinguished Alumnus of the U.C. Berkeley Computer Science program; he was named an ACM Fellow in 1994. In 1994, he founded Artifex Software to license Ghostscript commercially while continuing its development and its release as Open Source; Artifex today is a multi-million-dollar business. In 1999-2000, he served on the board of the Open Source Initiative. He is a co-inventor on two patents.

21. **David L. Dill.** David Dill is a Professor of Computer Science and, by courtesy, Electrical Engineering at Stanford University. Prof. Dill's Ph.D. thesis, "Trace Theory for Automatic Hierarchical Verification of Speed Independent Circuits" was named as a Distinguished Dissertation by the Association for Computing Machinery (ACM), and published as such by M.I.T. Press in 1988. He was named a

Fellow of the Institute of Electrical and Electronics Engineers (IEEE) in 2001 for his contributions to verification of circuits and systems, and a Fellow of the ACM in 2005 for contributions to system verification and for leadership in the development of verifiable voting systems. In 2008, he received the first "Computer-Aided Verification" award for fundamental contributions to the theory of real-time systems verification. In 2013, he was elected to the National Academy of Engineering and the American Academy of Arts and Sciences.

22. **Les Earnest.** Les Earnest is a widely-recognized computer scientist, best known for his deep involvement with the Advanced Research Project Agency Network (ARPAnet) startup committee, which led to his invention of the Finger protocol. He served as a US Navy Aviation Electronics Officer and Digital Computer Project Officer at the Naval Air Development Center, and later joined MIT to help design the Semi-Automatic Ground Environment air defense system. Later, he innovated numerous early features in the nascent field of word processing, including the first spell-checker. He was responsible for numerous significant contributions to the field of robotics, creating systems that coupled computer vision with prosthetic and vehicular applications.

23. **Brendan Eich.** Brendan is the former CTO of Mozilla, and is widely recognized for his enduring contributions to the Internet revolution. In 1995, Eich invented JavaScript (ECMAScript), the Internet's

most widely used programming language. He co-founded the mozilla.org project in 1998, serving as chief architect, and was a board member of the Mozilla Foundation since its inception in 2003 through 2014. Brendan helped launch the award-winning Firefox Web browser in November 2004 and Thunderbird e-mail client in December 2004.

24. **Dave Farber.** David Farber is Adjunct Professor of Internet Studies after his retirement as Distinguished Career Professor of Computer Science and Public Policy in the School of Computer Science at Carnegie Mellon University, holding a secondary appointment the Engineering Public Policy Group. In 2003, he retired as the Alfred Fitler Moore Professor of Telecommunication Systems at the University of Pennsylvania where he held appointments as Professor of Business and Public Policy at the Wharton School of Business and as a Faculty Associate of the Annenberg School of Communications. In 2000, he was appointed to be Chief Technologist at the US Federal Communications Commission while on leave from UPenn for one year ending in early June 2001. While at UPenn, he co-directed The Penn Initiative on Markets, Technology and Policy. He was also Director of the Distributed Systems Laboratory - DSL where he managed leading edge research in Ultra High Speed Networking. He is a Member of the Markle Foundation Taskforce on National Security, and a Member of the Board of Trustees of both the Internet Society and the Electronic Privacy Information Center (EPIC). He is a Fellow of both the ACM and the IEEE and was

the recipient of the 1995 ACM Sigcomm Award for lifelong contributions to the computer communications field. He was awarded in 1997 the prestigious John Scott Award for Contributions to Humanity.

25. **Stuart Feldman.*** Stuart Feldman is responsible for the health and productivity of Google's engineering offices in the eastern part of the Americas, Asia, and Australia. Feldman did his academic work in astrophysics and mathematics and earned his A.B. at Princeton and his Ph.D. at MIT. He is former President of the ACM, and former member of the board of directors of the AACSB (Association to Advance Collegiate Schools of Business). He received the 2003 ACM Software System Award. He is a Fellow of the IEEE, ACM, and AAAS, and serves on a number of government advisory committees. Before joining Google, he worked at IBM for eleven years. Most recently, he was Vice President for Computer Science in IBM Research, where he drove the long-term and exploratory worldwide science strategy in computer science and related fields, led programs for open collaborative research with universities, and influenced national and global computer science policy. Before joining IBM in mid-1995, he was a computer science researcher at Bell Labs and a research manager at Bellcore (now Telcordia). In addition he was the creator of the Unix make utility as well as the architect for a large new line of software products at Bellcore.

26. **Edward W. Felten.** Edward W. Felten is the Robert E. Kahn Professor of Computer Science and Public Affairs at Princeton University, and the founding Director of Princeton's Center for Information Technology Policy. In 2011-12, he served as the first Chief Technologist at the U.S. Federal Trade Commission. His research interests include computer security and privacy, especially relating to media and consumer products; and technology law and policy. He has published about eighty papers in the research literature, and two books. His research on topics such as web security, copyright and copy protection, and electronic voting has been covered extensively in the popular press. His weblog, at freedom-to-tinker.com, is widely read for its commentary on technology, law, and policy. He is a member of the National Academy of Engineering and the American Academy of Arts and Sciences, and is a Fellow of the ACM. He has testified before the House and Senate committee hearings on privacy, electronic voting, and digital television. In 2004, Scientific American magazine named him to its list of fifty worldwide science and technology leaders.

27. **Jeremiah Flerchinger.** Jeremiah Flerchinger is an acquisitions and operations officer in the United States Air Force. He has spent over ten years in developmental engineering and space operations with the Department of Defense. Flerchinger holds an M.S. in Electrical Engineering and has been a registered member of the Institute of Electrical and Electronic Engineers since 1999.

28. **Martin Fowler**. Martin Fowler is an author and educator on software development. He is currently chief scientist at ThoughtWorks, a global system delivery and consulting firm. Mr. Fowler concentrates on the design of enterprise software: what makes a good design and what practices are needed to enhance it. He is the author of seven books on software development, which have over a million copies in print in over a dozen languages. He is the editor of a book series with Addison-Wesley on software design. His website, http://martinfowler.com, is a wide-ranging resource of software development techniques attracting around 150,000 visitors per month.

29. **Neal Gafter.** Neal Gafter is a Partner Architect at Microsoft, where he is the technical lead for the design and implementation of new compilers for the Roslyn Project (the core of Microsoft's future C# and Visual Basic products). Previously he was a software engineer and Java Evangelist at Google, where he designed and implemented the distributed storage architecture for Google Calendar, and a Senior Staff Engineer at Sun Microsystems, where he led the development of the Java compiler and implemented the Java language features in releases 1.4 through 5.0. Neal was a member of the C++ Standards Committee and led the development of C and C++ compilers at Sun Microsystems, Microtec Research, and Texas Instruments. He holds a B.S. in computer engineering from Case Western Reserve University and a Ph.D. in computer science from the University of Rochester.

30. **John Gage.** John Gage was the 21st employee of Sun Microsystems, where he served as Chief Researcher and Vice President of the Science Office. He coined the phrase "The network is the computer" and co-founded JavaOne, the preeminent Java technology conference for over a decade. In 2008, he left Sun to join Kleiner Perkins Caufield & Byers as a partner to work on green technologies for global warming. He departed KPCB in 2010 to apply what he had learned to develop water, power and sanitation projects in world slums, beginning with Kibera, the largest slum in East Africa. Gage has served on scientific advisory panels for the US National Research Council and the US National Academy of Sciences. Most recently, he served on the National Academy Committee on Scientific Communication and National Security and the Markle Task Force on National Security. He co-founded NetDay in 1995 and also served on the boards of the US National Library of Medicine, FermiLabs; the Mathematical Sciences Research Institute; and other scientific and educational groups. Currently he serves on the United Nations Task Force on Digital Health, the Malaysian Multimedia Corridor International Advisory Panel, and on the boards of the University of Oxford Martin School, the Tegla Loroupe Peace Foundation, and the Human Needs Project. Awards include the ACM Presidential Award and the S.A.P. Information Technology Leadership Award for Education. Gage attended the University of California, Berkeley; Harvard Business School; and the Harvard Kennedy School of Government. He holds a B.S. from UC Berkeley.

31. **Allan Gottlieb.** Allan Gottlieb has been a professor of Computer Science within the Courant Institute at New York University for 34 years, teaching both undergraduate and graduate level courses covering programming languages, compilers, operating systems and system architecture. He is an elected fellow of the Association for Computing Machinery (ACM), and has previously served as Director of the NYU Ultracomputer Project. Allan is a member of the Computer Society of the Institute of Electrical and Electronic Engineers (IEEE), and has edited 'Puzzle Corner' for MIT's "Technology Review" for over 48 years. Allan holds a BS from MIT, and an MA and PhD from Brandeis University, all in mathematics.

32. **Miguel de Icaza.** Miguel de Icaza was an early contributor to Linux projects. In 1997, he co-founded the GNOME project, with the goal to create a completely free desktop environment. In 2001, he co-founded and directed the Mono Project, with the goal to implement Microsoft's .NET development platform on Linux. He has started two companies: Ximian in 1999, which focused on the Linux desktop and was sold to Novell in 2003; and Xamarin which was founded in 2011. Xamarin builds development tools for mobile developers and currently employs over 200 people. He has received numerous awards and recognitions including: the Free Software Foundation Free Software Award, the MIT Technology Review Innovator of the Year Award, and was named one of Time Magazine's 100 innovators for the new century.

33. **John Hennessy.*** John Hennessy is a Professor of Electrical Engineering and Computer Science and is the President of Stanford University. He serves on the boards of Google, Cisco Systems, and the Gordon and Betty Moore Foundation. Professor Hennessy is an IEEE Fellow, a member of the National Academy of Sciences and the National Academy of Engineering, and a Fellow of the American Academy of Arts and Sciences and the Association for Computing Machinery. He is the co-author of two internationally used undergraduate and graduate textbooks on computer architecture design.

34. **Tom Jennings.** Tom Jennings has specialized in computers, software, and electronics since 1977; computer networking since 1984; and the Internet since 1992. Jennings was on the team that wrote the interface specification (API in today's parlance) for Phoenix Software's IBM compatible ROM BIOS. Jennings is the creator of FidoNet, the first and most influential message and file networking system protocol for networking computer bulletin boards. Jennings built Wired magazine's first internet presence as its first webmaster and ran an early regional internet service provider, TLGnet. Currently, Jennings is on the faculty at Calarts Art+Technology program.

35. **Mitchell Kapor.** Mitchell Kapor founded Lotus Development Corp. in 1982 and co-created Lotus 1-2-3. He served as the President (later Chairman) and Chief Executive Officer of Lotus from 1982 to

1986, and as a Director until 1987. In 1990, Mr. Kapor co-founded the Electronic Frontier Foundation, and served as its chairman until 1994. From 1999 to 2001, he was a partner at Accel Partners, a venture capital firm based in Palo Alto, California. In 2003, Mr. Kapor became the founding Chair of the Mozilla Foundation, which is dedicated to the development and promulgation of standards-compliant, open source web browser software. From 1994-1996, he served as Adjunct Professor at the Massachusetts Institute of Technology's Media Lab, where he taught courses on software design, democracy and the Internet, and digital community. From 2005 to 2009, Mr. Kapor was on the faculty of the Information School at the University of California, Berkeley, as Lecturer (2005-2006) and Adjunct Professor (2006-2009), where he co-taught "Open Source Development and Distribution of Information." He is currently a Partner at Kapor Capital, which invests in seed-stage, social-impact tech startups; and Co-Chair of the Kapor Center for Social Impact, which pursues creative strategies to leverage tech for positive, progressive change. Mr. Kapor received a B.A. from Yale College in 1971 and studied psychology, linguistics, and computer science as part of an interdisciplinary major in Cybernetics.

36. **Alan Kay.** Alan Kay is one of the pioneers of object-oriented programming, personal computing, and graphical user interfaces. For this work, Dr. Kay has received the Draper Prize from the National Academy of Engineering, the ACM Turing Award, and the Kyoto Prize from the Inamori Foundation. Alan has been

elected a fellow of the American Academy of Arts and Sciences, the National Academy of Engineering, the Royal Society of Arts, the AAAS, and the Computer History Museum. Alan has held fellow positions at HP, Disney, Apple, and Xerox, and has served as the chief scientist at Atari. While at Xerox PARC, he was one of the key members there to develop prototypes of networked workstations using the programming language Smalltalk. He is an adjunct professor of computer science at UCLA and an advisor to One Laptop per Child. At Viewpoints Research, Alan also continues his work with "powerful ideas education" for the world's children, as well as the development of advanced personal computers and networking systems.

37. **Brian Kernighan.*** Brian Kernighan is a professor in the Computer Science Department of Princeton University. He worked at Bell Labs alongside Unix creators Ken Thompson and Dennis Ritchie and contributed to the development of Unix. He co-authored a number of Unix programs, including widely used document preparation tools. He is also the author or co-author of 11 books on computing, including the first book on the C programming language with Dennis Ritchie; these books have been translated into more than two dozen languages. He is also a co-creator of the AWK and AMPL programming languages. In collaboration with Shen Lin he devised well-known heuristics for two fundamental NP-complete optimization problems: graph partitioning and the traveling salesman problem. Kernighan

received a Bachelor's degree in engineering physics from the University of Toronto, and his Ph.D. in electrical engineering from Princeton University. He is a member of the National Academy of Engineering.

38. **David Klausner**. David Klausner has over 45 years of software/hardware development and consulting experience in the computer and software industry. He has written software for commercial products as an engineer, developer, supervisor, project manager, department manager, middle manager and company executive. He has worked in forensic investigation and in reverse engineering. He has been employed in various capacities for various companies, such as Microsoft, AT&T, Cisco, IBM, Hewlett Packard, and Intel Corporation. David Klausner holds a Bachelors of Arts degree in Mathematics, and a Master of Science degree in Electrical Engineering. He has taught programming, public speaking, has guest lectured at Stanford University, and been an invited speaker by IBM, AT&T, and others.

39. **Ray Kurzweil.*** Ray Kurzweil is an inventor, author and futurist. He was the principal inventor of the first CCD flat-bed scanner, the first omni-font optical character recognition, the first print-to-speech reading machine for the blind, the first text-to-speech synthesizer, the first music synthesizer capable of recreating the grand piano and other orchestral instruments, and the first commercially marketed large-vocabulary speech recognition. Kurzweil is the recipient of the National Medal of Technology, was

inducted into the National Inventors Hall of Fame, holds twenty honorary Doctorates, and has received honors from three U.S. presidents. He is presently a Director of Engineering at Google heading up a team developing machine intelligence and natural language understanding.

40. **Kin Lane.** Kin is a computer scientist and API Evangelist working to understand the technology, business and politics of APIs and help share this insight with the world. He is the author of the book, Business of APIs, and is behind the popular API Evangelist blog. He has over 20 years of experience as a programmer, database administrator, architect, product developer, manager, and executive in the API space.

41. **Ed Lazowska.** Ed Lazowska is the Bill & Melinda Gates Chair in Computer Science & Engineering at the University of Washington. His research concerns the design, implementation, and analysis of high performance computing and communication systems, and, more recently, the techniques and technologies of data-intensive discovery. He co-chaired (with Marc Benioff) the President's Information Technology Advisory Committee from 2003-05, and (with David E. Shaw) the Working Group of the President's Council of Advisors on Science and Technology to review the Federal Networking and Information Technology Research and Development Program in 2010. He is a Member of the National Academy of Engineering and a Fellow of the American Academy of Arts and Sciences.

42. **Doug Lea.** Doug Lea is a professor of Computer Science at the State University of New York at Oswego. He is an author of books, articles, reports, and standardization efforts on object oriented software development including those on specification, design and implementation techniques, distributed, concurrent, and parallel object systems, and software reusability; he has served as chair, organizer, or program committee member for many conferences and workshops in these areas. He is the primary author of several widely used software packages and components.

43. **Sheng Liang.** Sheng Liang is a software entrepreneur. He was CTO of the Cloud Platform group at Citrix Systems after their acquisition of Cloud.com, where he was co-founder and CEO. Sheng was co-founder and CTO of Teros, a provider of perimeter and network security solutions for enterprises and service providers, acquired by Citrix Systems in 2005. He also served as VP of Engineering at SEVEN Networks, and Director of Software Engineering at Openwave Systems. He was a Staff Engineer in Java Software at Sun Microsystems, where he designed the Java Native Interface (JNI) and led the Java Virtual Machine (JVM) development for the Java 2 platform. He has a B.S. from the University of Science and Technology of China and a Ph.D. from Yale University.

44. **Barbara Liskov.** Barbara Liskov is one of the world's leading authorities on computer language

and system design. Liskov joined MIT in 1972 as a member of the Department of Electrical Engineering and computer Science. She is also a member of the MIT laboratory for Computer Science and Artificial Intelligence and heads the programming methodology group. Her research interests lie in programming methodology, programming languages and systems, and distributed computing. Major projects include: the design and implementation of CLU, the first programming language to support data abstraction; the design and implementation of Argus, the first high-level language to support implementation of distributed programs; and the Thor object-oriented database system, which provides transactional access to persistent, highly-available objects in wide-scale distributed environments. Liskov is a fellow of the American Academy of Arts and Sciences, the National Academy of Inventors, the Association for Computing Machinery, and the Massachusetts Academy of Science. She is a member of the National Academy of Science and the National Academy of Engineering. In 2009, she received the A.M. Turing Award from the ACM. Other honors include the Society of Women Engineers' Achievement Award, the IEEE von Neumann medal, the ACM SIGPLAN Programming Languages Achievement Award, the University of Pennsylvania Harold Pender Award, the ACM SIGOPS Hall of Fame Award, the CMU and Tokyo University of Technology Katayanagi Award for Research Excellence, the ACM SIGOPS Lifetime Achievement Award, and five honorary doctorates. She holds a B.A. from UC Berkeley and a Ph.D. from Stanford.

45. **Paul Menchini**. Paul Menchini is the Chief Information Security Officer at the North Carolina School of Science and Mathematics. Previously, he held technical positions at HP, Intel, GE Microelectronics, CLSI and OrCAD. As a member of the "Woods Hole Summer Study on Hardware Description Languages," he contributed to the specifications for VHDL; subsequently, he edited two revisions of IEEE Std 1076 VHDL and developed the first commercially successful VHDL compiler. As part of the compiler project, he developed an API for a VHDL intermediate form, which was subsequently standardized by the IEEE. He holds a Masters in Computer Engineering from Stanford University and is the recipient of numerous technical awards, including charter membership in the "IEEE Golden Core."

46. **James H. Morris.** Dr. James H. Morris is a Professor of Computer Science at Carnegie Mellon University, where he served as Dean of the Silicon Valley Campus, Dean of the School of Computer Science, Head of the Computer Science Department, and Director of the Information Technology Center, a joint project with IBM that developed a prototype university computing system. He founded Carnegie Mellon's Human Computer Interaction Institute, Robot Hall of Fame, and Silicon Valley Campus. He was an Associate Professor at UC Berkeley, where he developed two fundamental principles of programming languages: inter-module protection and lazy evaluation. He was co-discoverer of the Knuth-Morris-Pratt string-

searching algorithm. He was Principal Scientist and Research Fellow at Xerox PARC, where he was part of the team that developed the Alto, a precursor to today's personal computers. He is a founder of MAYA Design Group and an ACM Fellow. He holds a B.S. from CMU and an M.S. and Ph.D. from MIT.

47. **Peter Norvig.*** Peter Norvig is a Director of Research at Google; previously he directed Google's core search algorithms group. He is co-author of *Artificial Intelligence: A Modern Approach*, the leading textbook in the field, and co-teacher of an Artificial Intelligence class that signed up 160,000 students, helping to kick off the current round of massive open online classes (MOOCs). He is a fellow of the AAAI, ACM, California Academy of Science and American Academy of Arts and Sciences.

48. **Martin Odersky.** Martin is a professor at EPFL in Lausanne, Switzerland. He is best known as the creator and principal designer of the Scala programming language. Prior to that, he made several contributions to the development of Java. He created the Pizza and GJ languages, designed the original version of generics for Java, and wrote the javac reference compiler for Java. He is a fellow of the ACM.

49. **Tim Paterson.** Tim began his career designing one of the first 16-bit microcomputer systems at Seattle Computer Products. He then wrote an operating system for that computer, which was later sold to

Microsoft and became widely used as MS-DOS. He went on to found his own company, Falcon Technology, whose primary products were hard disk systems for personal computers. He moved on to Microsoft where he was a software engineer for many years, working on such products as QuickBASIC, Visual Basic, VBScript, and Visual J++ (Java). After his retirement in the late '90s he has continued developing software and microcontroller-based hardware projects as a hobby and part-time small business. He has been granted three US patents on software methods.

50. **David Patterson.*** David Patterson joined UC Berkeley in 1977. He has been Director of the Par Lab, Chair of UC Berkeley's CS Division, Chair of the Computing Research Association, and President of the Association for Computing Machinery. His most successful projects have been Reduced Instruction Set Computers (RISC), Redundant Arrays of Inexpensive Disks (RAID), and Network of Workstations. All helped lead to multibillion-dollar industries. This research led to many papers, six books, and about 35 honors, including election to the National Academy of Engineering, the National Academy of Sciences, the Silicon Valley Engineering Hall of Fame, and Fellow of the Computer History Museum. He shared the IEEE von Neumann Medal and the NEC C&C Prize with John Hennessy, President of Stanford University and co-author of two of his books.

51. **Alex Payne.** Alex Payne consults, advises, and invests in early-stage technology startups. As Platform

Lead at Twitter he managed one of the web's most popular APIs. He was subsequently co-founder and Chief Technology Officer of online banking service Simple, acquired by BBVA in 2014. Alex organizes an annual conference showcasing advances in programming languages and has co-authored a book on the Scala programming language (O'Reilly, 2009). He is a regular speaker at technology and business conferences worldwide and has lectured on API design at Stanford.

52. **Tim Peierls.** Since receiving a BS in Computer Science from Yale in 1983 and an MS in CS from Cornell in 1986, Tim has worked in the software industry continuously, first at Bell Labs (airline crew scheduling), then co-founding the Lightstone Group in 1990 (aircraft scheduling, delivery vehicle routing and scheduling, sold to Descartes Systems Group in 1998) and Seat Yourself in 2002 (online ticketing for school performing arts groups). For the last fifteen years, almost all of his programming work has been in Java. He has served on the Expert Groups of several Java Specification Requests (166, 201, 330, 334) and on the SE/EE Executive Committee of the Java Community Process; he co-authored a book, *Java Concurrency in Practice*; and he contributed code, support, and advice to various open source projects, including Restlet, Hazelcast, and JClouds.

53. **Simon Phipps.** Simon is the President of the Open Source Initiative, the global steward of the Open Source Definition. OSI serves to advocate for,

educate about and build bridges within the open source community. A Fellow of the British Computer Society, his career has included early engagement in establishing Java, XML and weblogs as computer industry technologies as well as contributions to open standards in a variety of fields. As chief open source officer at Sun Microsystems he supervised the open source relicensing of Solaris Unix, Java and many other software systems. He is currently founder and CEO of Meshed Insights Ltd, a UK firm offering management services related to open source and digital rights.

54. **Bill Pugh.** Bill invented Skip Lists, a randomized data structure that is widely taught in undergraduate data structure courses. He has also made research contributions in techniques for analyzing and transforming scientific codes for execution on supercomputers, and in a number of issues related to the Java programming language, including the development of JSR 133 - Java Memory Model and Thread Specification Revision. Current research projects include FindBugs, a widely used static analysis tool for Java, and Marmoset, an innovative framework for improving the learning and feedback cycle for student programming projects. He is currently a professor emeritus of computer science at the University of Maryland.

55. **Larry Roberts.** Dr. Roberts most recently was Founder, Chairman and CEO of Anagran Inc. Anagran is currently manufacturing flow rate

management network equipment, the first major improvement in packet network technology in the 40 years since Dr. Roberts designed and managed the first packet network, the ARPANET (now the Internet). At that time, in 1967, Dr. Roberts joined ARPA taking on the task of designing, funding, and managing a radically new communications network concept (packet switching) to interconnect computers worldwide. The first four nodes of the ARPANET were installed in 1969 and by 1973 when Dr. Roberts left ARPA to become CEO of Telenet (now part of Sprint), the concept of packet switching had been well proven to the world. Dr. Roberts has B.S., M.S., and Ph.D. Degrees from MIT and has received numerous awards for his work, including the Secretary of Defense Meritorious Service Medal, the L.M. Ericsson prize for research in data communications, in 1992 the W. Wallace McDowell Award, in 1998 the ACM SIGCOMM Award, in 2000 the IEEE Internet Award, in 2001 the National Academy of Engineering Draper Award, in 2002 the Principe de Asturias Award, and in 2005 the NEC Computer and Communication Award.

56. **Guido van Rossum.** Guido van Rossum created the open-source programming language Python, and is its lead developer and thought leader. Python is widely used in industry, and is the most popular introductory teaching language at top US universities. Guido developed the Python language while at CWI in Amsterdam. After moving to the US he worked as a guest researcher at NIST, at CNRI, and at several

start-up companies. He became a Senior Staff Engineer at Google, and currently works for Dropbox. Guido is an ACM Distinguished Engineer and a recipient of several awards including the USENIX STUG Award, the NLUUG Award, the Free Software Foundation Award, and the Dr. Dobb's Journal 1999 Excellence in Programming Award. In 2013, Python was awarded the Dutch National ICT COMMIT/ Award. Guido holds an M.S. in Mathematics and Computer Science from the University of Amsterdam.

57. **Avi Rubin.** Dr. Aviel D. Rubin is Professor of Computer Science and Technical Director of the Information Security Institute at Johns Hopkins University. Prior to joining Johns Hopkins, Rubin was a research scientist at AT&T Labs. Rubin has testified before the U.S. House and Senate on multiple occasions, and he is author of several books including *Brave New Ballot* (Random House, 2006), *Firewalls and Internet Security, second edition* (with Bill Cheswick and Steve Bellovin, Addison-Wesley, 2003), *White-Hat Security Arsenal* (Addison-Wesley, 2001), and *Web Security Sourcebook* (with Dan Geer and Marcus Ranum, John Wiley & Sons, 1997). He has served as Associate Editor of IEEE Transactions on Information Forensics and Security, Associate Editor of Communications of the ACM (CACM), and an Advisory Board member of Springer's Information Security and Cryptography Book Series. Dr. Rubin spent the 2010-2011 academic year as a Fulbright Scholar at Tel Aviv University. Rubin has a B.S. ('89), M.S.E. ('91), and Ph.D. ('94) from the University of Michigan.

58. **Bruce Schneier.** Bruce Schneier is an internationally renowned security technologist, called a "security guru" by The Economist. He is the author of 12 books – including Liars and Outliers: Enabling the Trust Society Needs to Survive – as well as hundreds of articles, essays, and academic papers. His influential newsletter "Crypto-Gram" and blog "Schneier on Security" are read by over 250,000 people. Schneier is a fellow at the Berkman Center for Internet and Society at Harvard Law School, a program fellow at the New America Foundation's Open Technology Institute, a board member of the Electronic Frontier Foundation, and an Advisory Board member of the Electronic Privacy Information Center. He is also the Chief Technology Officer of Co3 Systems, Inc.

59. **Curtis Schroeder.** Curtis is a U.S. citizen currently working for a European company, Antycip Simulation, and based in the UK. He served as the Drafting Group Editor for the Simulation Interoperability Standards Organization (SISO) Common Image Generator Interface (CIGI) 4.0 international standard. The success of SISO international standards depends upon implementation of said copyrighted standards by numerous simulation vendors and end-users, including NATO. Previously, Curtis worked for Lockheed Martin for sixteen years, where he utilized a number of open standards in projects he was involved in. He earned B.S. & M.S. Computer Science degrees at the Missouri University of Science & Technology.

60. **Barbara Simons.** Barbara Simons is a former President of the Association for Computing Machinery (ACM), the nation's oldest and largest educational and scientific society for computing professionals. She is the only woman to have received the Distinguished Engineering Alumni Award from the College of Engineering of U.C. Berkeley, where she earned her Ph.D. in computer science. A fellow of ACM and the American Association for the Advancement of Science, she also received the Computing Research Association Distinguished Service Award and the Electronic Frontier Foundation Pioneer Award. She recently published *Broken Ballots: Will Your Vote Count?,* a book on voting machines co-authored with Douglas Jones. She was appointed to the Board of Advisors of the U.S. Election Assistance Commission in 2008, and she co-authored the report that led to the cancellation of Department of Defense's Internet voting project (SERVE) in 2004 because of security concerns. She was a member of the National Workshop on Internet Voting, convened by President Clinton, which conducted one of the first studies of Internet Voting and produced a report in 2001. She is Board Chair of Verified Voting and is retired from IBM Research.

61. **Dave Snigier.** Dave Snigier is a futurist who uses technology to solve human problems. He has led several successful projects as part of the Emerging Technologies group at UMass including a system-wide paperless initiative. He now spends his days as an architect responsible for designing how different

systems will communicate with each other; often including the use of APIs as a significant component.

62. **Alfred Z. Spector.*** Dr. Alfred Z. Spector is responsible for research at Google and also Google's open source, university relations, internationalization, and many education initiatives. Dr. Spector is the Executive Engineering Lead for Google Social Impact Engineering. Previously, Dr. Spector was Vice President of Strategy and Technology at IBM's software business, and prior to that, he was Vice President of Services and Software Research across IBM. He was also founder and CEO of Transarc Corporation and was a faculty member at Carnegie Mellon University, specializing in highly reliable, highly scalable distributed computing. Dr. Spector received an A.B. in applied mathematics from Harvard and a Ph.D. in computer science from Stanford. He is a member of the National Academy of Engineering and a Fellow of American Academy of Arts and Sciences, the IEEE, and the ACM. Dr. Spector is also the recipient of the 2001 IEEE Computer Society's Tsutomu Kanai Award for work in scalable architectures and distributed systems.

63. **Bjarne Stroustrup.** Bjarne Stroustrup is the inventor of the C++ programming language. He wrote the standard textbook on the language and its implementation, *The C++ Programming Language*, and many other academic and popular books and articles. He has served on the ISO Standards committee since its creation in 1989. He is a fellow

of both the ACM and the IEEE, and an elected as a member of the National Academy of Engineering. He holds a masters degree in mathematics and computer science from Aarhus University, in Denmark, and a Ph.D. in computer science from the University of Cambridge.

64. **Gerald Jay Sussman.** Gerald Jay Sussman is the Panasonic (formerly Matsushita) Professor of Electrical Engineering at the Massachusetts Institute of Technology. He has been involved in artificial intelligence research at M.I.T. since 1964. His research has centered on understanding the problem-solving strategies used by scientists and engineers, with the goals of automating parts of the process and formalizing it to provide more effective methods of science and engineering education. Sussman has also worked in computer languages, in computer architecture, and in VLSI design. Sussman is a coauthor of the introductory computer science textbook that included innovative advances in curricula designed for students pursuing different kinds of computing expertise, which has had a worldwide impact on university computer-science education. Sussman has received numerous awards and recognitions including: the ACM's Karl Karlstrom Outstanding Educator Award, the Amar G. Bose award for teaching, a fellow of the Institute of Electrical and Electronics Engineers, a fellow of the American Academy of Arts and Sciences, a member of the National Academy of Engineering, and a fellow of the American Association for the Advancement of

Science. He received the S.B. and the Ph.D. degrees in mathematics from the Massachusetts Institute of Technology in 1968 and 1973.

65. **Brad Templeton.** Brad Templeton, active in the computer network community since 1979, was founder and publisher at ClariNet Communications Corp., the electronic newspaper that was perhaps the earliest dot-com company. He participated in the building and growth of USENET from its earliest days, and in 1987 founded and edited rec.humor.funny, for many years the world's most widely read electronic publication. He was the first employee of Personal Software/Visicorp, the first major microcomputer applications software company. He later founded Looking Glass Software and over the years was author of a dozen packaged microcomputer software products, including VisiPlot for the IBM-PC, various games, popular tools and utilities for Commodore computers, special Pascal and Basic programming environments designed for education (ALICE), an add-in spreadsheet compiler for Lotus 1-2-3 (3-2-1 Blastoff), and various network related software tools. He currently is track chair for computing and networks at Singularity University, a consultant and speaker on self-driving cars, and is on the board of the Electronic Frontier Foundation and the Foresight Nanotech Institute. He was Chairman of EFF's Board of Directors from 2000 to early 2010.

66. **Ken Thompson.*** Ken Thompson spent much of his career at Bell Laboratories where he co-designed and implemented the original Unix operating system,

invented the B programming language that was a precursor to the C programming language, invented the Bon programming language, co-developed the Plan 9 operating systems, developed the CTSS version of the editor QED, developed ed, which is the standard text editor on Unix, and the definition of the UTF-8 encoding, which is used for more than half of all Web pages. Thompson also co-developed the software and hardware for Belle, which was the first computer built for the sole purpose of chess playing, and it officially became the first master-level machine in 1983. He is currently a Google Advisor and was formerly a Distinguished Engineer at Google, where he invented new programming languages (including the Go programming language as a co-inventor), among other projects. Thompson is a recipient of numerous awards and commendations in connection with his work on Unix, including the IEEE Emanuel R. Piore Award (1982), the Turing Award (1983), the IEEE Richard W. Hamming Medal (1990), and the National Medal of Technology (1999). Thompson holds a B.S. and an M.S., both in Electrical Engineering and Computer Science, from the University of California, Berkeley. He has been awarded two honorary Ph.D.s.

67. **Michael Tiemann.** Michael Tiemann is a true open source software pioneer. He made his first major open source contribution more than two decades ago by writing the GNU C++ compiler, the first native-code C++ compiler and debugger. His early work led to the creation of leading open source technologies and the first open source business model. Tiemann co-founded

Cygnus Solutions, the first company to provide commercial support for open source software. During his ten years at Cygnus, Tiemann contributed in a number of roles from President to hacker, helping lead the company from fledgling start-up to an admired open source leader. When Cygnus was acquired by Red Hat in 2000, Tiemann became Red Hat's Chief Technical Officer (CTO) before becoming its first Vice President of Open Source Affairs. Tiemann graduated from the Moore School at the University of Pennsylvania (Class of 1986) with a BS CSE degree, and later did research at INRIA (1988) and Stanford University (1988-1989). Tiemann retired as President of the Board at the Open Source Initiative in 2012, and was a founding Board member of both the Eclipse Foundation and the Embedded Linux Consortium.

68. **Andrew Tridgell.** Dr. Andrew Tridgell is a computer scientist and free software developer in Canberra, Australia. Best known for his contributions to the development of the award winning Samba suite of networking software that enables interoperability with Microsoft networking services, he has been actively developing in the area of interoperability for more than 20 years.

69. **Josh Triplett.** Josh Triplett is a Free and Open Source Software developer, contributing to projects such as the Linux kernel, Debian, Chrome OS, Git, Sparse, and the X Window System. Josh earned a Ph.D. at Portland State University, constructing concurrent data structures and synchronization techniques for

highly parallel systems. Josh enjoys using software for unconventional purposes, such as running Python in the GRUB2 bootloader to test the BIOS, and using Git as a database. In his "free time," Josh does as much of his hacking as possible in Haskell.

70. **Jeff Ullman.*** Jeff Ullman is the Stanford W. Ascherman Professor of Engineering (Emeritus) in the Department of Computer Science at Stanford and CEO of Gradiance Corp. He received the B.S. degree from Columbia University in 1963 and the Ph.D. from Princeton in 1966. Prior to his appointment at Stanford in 1979, he was a member of the technical staff of Bell Laboratories from 1966-1969, and on the faculty of Princeton University between 1969 and 1979. From 1990-1994, he was chair of the Stanford Computer Science Department. Ullman was elected to the National Academy of Engineering in 1989, the American Academy of Arts and Sciences in 2012, and has held Guggenheim and Einstein Fellowships. He has received the Sigmod Contributions Award (1996), the ACM Karl V. Karlstrom Outstanding Educator Award (1998), the Knuth Prize (2000), the Sigmod E. F. Codd Innovations award (2006), and the IEEE von Neumann medal (2010). He is the author of 16 books, including books on database systems, compilers, automata theory, and algorithms.

71. **John Villasenor.** John Villasenor is a professor of electrical engineering and public policy at UCLA and a nonresident senior fellow at the Brookings Institution in Washington DC. In addition, during fall 2014 he is a

National Fellow at the Hoover Institution at Stanford University. Professor Villasenor's research considers communications and information technologies and their broader ramifications, and has addressed topics including cybersecurity, autonomous vehicles, and digital media policy. Professor Villasenor is a member of the Council on Foreign Relations, a member of the World Economic Forum's Global Agenda Council on Cybersecurity, and a former vice chair of the World Economic Forum's Global Agenda Council on the Intellectual Property System. He holds an M.S. and Ph.D. in electrical engineering from Stanford University, and a B.S. in electrical engineering from the University of Virginia. Professor Villasenor advises students whose work is supported in part by funding from Google; he has not personally received any compensation from that funding.

72. **Jan Vitek.** Jan Vitek is a Professor of Computer Science at Northeastern University. He is the Chair of the ACM Special Interest Group on Programming Languages (SIGPLAN), the vice chair of AITO and of the IFIP WG 2.4, and is Chief Scientist at Fiji Systems. He holds a Ph.D. from the University of Geneva and an MSc from the University of Victoria. He works on various aspects of programming languages including virtual machines, compilers, software engineering, real-time and embedded computing, concurrency and information security. Prof. Vitek led the Ovm project which resulted in the first successful flight test of real-time Java virtual machine. With Noble and Potter, Vitek proposed the notion of ownership

for alias control, which became known as ownership types. He chaired PLDI, ISMM and LCTES and was program chair of ESOP, ECOOP, VEE, Coordination, and TOOLS.

73. **Phillip Wadler.** Philip Wadler is Professor of Theoretical Computer Science at the University of Edinburgh. He is an ACM Fellow and a Fellow of the Royal Society of Edinburgh, past chair of ACM SIGPLAN, and past holder of a Royal Society-Wolfson Research Merit Fellowship. Previously, he worked or studied at Stanford, Xerox Parc, CMU, Oxford, Chalmers, Glasgow, Bell Labs, and Avaya Labs, and visited as a guest professor in Copenhagen, Sydney, and Paris. He has more than 18,000 citations to his work according to Google Scholar. He is a winner of the POPL Most Influential Paper Award, has contributed to the designs of Haskell, Java, and XQuery, and is a co-author of Introduction to Functional Programming (Prentice Hall, 1988), XQuery from the Experts (Addison Wesley, 2004) and Generics and Collections in Java (O'Reilly, 2006). He has delivered invited talks in locations ranging from Aizu to Zurich.

74. **James H. "Jim" Waldo.** Jim Waldo is the Chief Technology Officer for Harvard University, where he is responsible for the architecture and implementation of the technology environment. He is also a Gordon McKay Professor of the Practice of Computer Science in the School of Engineering and Applied Sciences at Harvard. Previously, Jim designed clouds at VMware, and was a Distinguished Engineer at

Sun Microsystems, where he investigated next-generation large-scale distributed systems. He was the lead architect for Jini, a distributed programming system based on Java. Before joining Sun, Jim spent eight years at Apollo Computer and Hewlett Packard, working in the areas of distributed object systems, user interfaces, class libraries, text and internationalization. While at HP, he led the design and development of the first Object Request Broker, and was instrumental in getting that technology incorporated into the first OMG CORBA specification. Jim edited the book *The Evolution of C++: Language Design in the Marketplace of Ideas* (MIT Press), co-edited *Engaging Privacy and Information Technology in a Digital Age* (National Academies Press), and was one of the authors of *The Jini Specification* (Addison Wesley). More recently, he authored *Java: The Good Parts*. He is currently a member of the editorial boards of Queue magazine and Communications of the ACM. He holds over 50 patents. Jim received his Ph.D. from the University of Massachusetts (Amherst). He holds two M.A. degrees from the University of Utah.

75. **Dan Wallach.** Dan Wallach is a professor in the Department of Computer Science and a Rice Scholar at the Baker Institute for Public Policy at Rice University in Houston, Texas. His research considers a variety of different computer security topics, ranging from web browsers and servers through Java security, electronic voting technologies, and smartphones. Wallach is a member of the Air Force

Science Advisory Board and is a former member of the USENIX Association Board of Directors.

76. **Peter Weinberger.*** Peter Weinberger is a computer scientist at Google. Previously he was Chief Technology Officer at Renaissance Technologies, and held many positions at Bell Labs, including Information Sciences Research Vice President, responsible for computer science research, math and statistics, and speech. As a scientist at Bell Labs he worked on Unix, contributing to the design and implementation of the AWK programming language, the IO library for f77, the fast factoring program qfactor, the B-tree library cbt, a code generator for C, and a network file system. He did research on topics including operating systems, compilers, security, and number theory. Before joining Bell Labs, he taught in the Department of Mathematics at the University of Michigan, Ann Arbor. He holds a B.S. from Swarthmore and a Ph.D. from the University of California, Berkeley.

77. **Frank Yellin.*** Frank Yellin has spent over a decade working on runtime systems for interpreted and compiled languages. As a Staff Engineer in Embedded and Consumer at Sun Microsystems, he was an original member of the Java project. Yellin is co-author of *The Java Virtual Machine Specification* (Addison-Wesley, 1999), and co-authored the first version of the Java API specification. Previously he worked at Lucid, where he focused on multitasking, garbage collection, interrupts, and the compilation of Common Lisp. Yellin currently is a Staff Software

Engineer at Google, where he works on automatic scalable security testing. He holds an A.B. in Applied Mathematics from Harvard and an M.S. in Computer Science from Stanford. He is the inventor or co-inventor of sixteen patents.