Case No. 2017-2145

In the

United States Court of Appeals

for the

Federal Circuit

CISCO SYSTEMS, INC.,
Plaintiff-Appellant

v.

ARISTA NETWORKS, INC.,
Defendant-Appellee

Appeal from a Decision of the United States District Court for the Northern District of California, Case No. 5:14-cv-05344-BLF · Honorable Beth Labson Freeman, U.S. District Court Judge

BRIEF OF *AMICI CURIA*E GITHUB, MOZILLA, ENGINE ADVOCACY, AND SOFTWARE FREEDOM CONSERVANCY, URGING AFFIRMANCE OF THE JUDGMENT

MARCIA HOFFMAN
ZEITGEIST LAW PC
25 Taylor Street
San Francisco, California 94102
(415) 830-6664 Telephone
marcia@zeitgeist.law

Counsel for Amici Curiae

COUNSEL PRESS · (213) 680-2300 PRINTED ON RECYCLED PAPER

CERTIFICATE OF INTEREST

Pursuant to Federal Circuit Rules 29(a) and 47.4, counsel for *amici curiae* certifies the following:

1. The full names of the *amici curiae* represented by me are GitHub, Inc., Mozilla Corporation, Engine Advocacy, and the Software Freedom Conservancy.

2. The name of the real party in interest (if the party named in the caption is not the real party in interest) represented by me is: none.

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

   Mozilla Corporation is wholly owned by the Mozilla Foundation, a non-profit organization. No other *amicus curiae* represented by me has a parent corporation.

   No publicly held corporation owns 10% or more of any of the *amici curiae*'s stock.

4. The names of all law firms and the partners and associates that appeared for the *amici curiae* now represented by me in the district court or are expected to appear in this court are:

   The *amici curiae* did not appear in the district court.

   They are represented before this Court by Marcia Hofmann of Zeitgeist Law PC.

5. The title and number of any case known to counsel to be pending in this or any other court or agency that will directly affect or be directly affected by this court's decision in the pending appeal: none.

/s/ Marcia Hofmann
Marcia Hofmann

TABLE OF CONTENTS

TABLE OF AUTHORITIES

CASES

CONSTITUTIONS

STATUTES AND RULES

OTHER AUTHORITIES

STATEMENT OF INTEREST OF *AMICI CURIAE*[1]

Amici are for-profit technology companies and innovation-focused non-profit organizations. They understand that interoperation of technology best serves the public interest, and that a balanced copyright regime best secures that goal.

GitHub is a web-based software development platform that enables users and businesses to collaboratively develop open-source and proprietary software projects. GitHub.com hosts over 73 million projects and welcomes more than 26 million users, and a majority of the Fortune 50 uses GitHub Enterprise. GitHub-hosted software projects include applications designed for web and mobile devices, as well as the source code that powers entire businesses. Developers on GitHub work together, sharing code and knowledge. As such, GitHub has an interest in reducing barriers to collaboration and promoting innovation in software development.

Mozilla is a global, mission-driven technology organization that works with a community of software developers around the globe to create open-source software such as the Firefox browser. Firefox is among the most popular browsers in the world. Several hundred million users rely on it to discover, experience, and connect to the Internet on computers, tablets, and mobile phones. Mozilla's mission is guided by a

[1] Pursuant to Federal Rule of Appellate Procedure 29(a), *amici* state that all parties have consented to the filing of this brief. It was not written in whole or part by counsel for any party. No person or entity other than undersigned counsel or *amici* has made a monetary contribution to the preparation or submission of this brief.

set of principles recognizing that, among other things, free and open software promotes the development of the Internet as a global public resource, and that the effectiveness of that resource depends on interoperability.

Engine Advocacy is a non-profit technology policy, research, and advocacy organization that bridges the gap between policymakers and startups, working with government and a community of high-technology, growth-oriented startups across the nation to support the development of technology entrepreneurship. Engine creates an environment where technological innovation and entrepreneurship thrive by providing knowledge about the start-up economy and constructing smarter public policy. To that end, Engine conducts research, organizes events, and spearheads campaigns to educate elected officials, the entrepreneur community, and the general public on issues vital to fostering technological innovation.

Software Freedom Conservancy is a charity that promotes, improves, develops, and defends free and open-source software developed by volunteer communities and licensed for the benefit of everyone. Conservancy is the nonprofit home for 46 free and open-source projects and initiatives such as Git, Busybox, Homebrew, Samba, QEMU and Selenium, which include thousands of volunteer contributors. Conservancy's communities maintain some of the most fundamental utilities in computing today, and introduce innovations that shapes software for the future.

INTRODUCTION

Copyright law is designed to ensure that creators are rewarded for their original efforts, but do not hold a monopoly over the functional concepts embodied in their works. *Amici* urge this Court to ensure that ideas and functionality in software remain free for others to use, as Congress intended.

In this case, the Court is narrowly focused on Arista Network's use of a portion of Cisco's compilation of commands, which network engineers use in a command line interface (CLI) to communicate with switches and routers. But the Court's decision will affect developers and engineers around the globe working to create new and innovative technologies to solve problems. Developers may use existing CLI commands to enable their software programs to interact with other software. This re-use creates new software that is easy for consumers to learn to use and in turn is more efficient for other developers to make interoperable with their own software.

Cisco argues that Arista's use of the compilation cannot be scenes a faire as a matter of law. This Court should not accept Cisco's position because the compilation was a necessary incident to the expression of functional concepts and was dictated by external factors. Cisco's proposed rule would have far-reaching implications for competition, the speed at which new technologies can be created, and consumer convenience. Copyright law must include latitude for newcomers to use existing CLI

commands and other functional elements in new technology so that the software industry can continue to flourish.

ARGUMENT

I. Copyright Law Must Leave Breathing Room for Software Interoperability and Innovation.

Limitations on copyright protection such as scenes a faire, merger, and fair use serve a critical purpose: they ensure that authors do not hold rights over the ideas and functionality embodied in their creative works. *Mitel, Inc. v. Iqtel, Inc.*, 124 F.3d 1366, 1375 (10th Cir. 1997) (citing *Gates Rubber Co. v. Bando Chem. Indus., Ltd.*, 9 F.3d 823, 838 (10th Cir. 1993); *Computer Assocs. Int'l, Inc. v. Altai, Inc.*, 982 F2d 693, 711 (2d Cir. 1997). Courts must ensure these doctrines remain robust to serve the fundamental purpose of copyright law: to "promote the progress of Science and useful Arts." U.S. CONST. art. I, § 8, cl. 8. The decision below is consistent this purpose. Copyright law is intended not only to protect original expression, but also to give innovators latitude to build upon earlier works and create new ones.

A. The Courts Have Declined to Extend Copyright Protection to Functional Aspects of Software Under a Variety of Theories, Which Serves the Underlying Goals of the Copyright Act.

The courts have long found that functional aspects of software are not protected by copyright, and that affirmative defenses apply to functional elements where expressive elements are copyrightable. This is because copyright law is designed to strike a balance. On one hand, the Copyright Act rewards creators for

innovation by granting limited exclusive rights over their creative expression. 17 U.S.C. §§ 102(a) & 106. On the other hand, copyright law ensures that functional concepts or elements embodied in a work are free for all to use so that others can innovate, too. *Id*. at § 102(b).

It is particularly critical in the software space to strike the right balance between these two extremes. Several software companies serving as *amici* argue that "[w]ithout adequate copyright protection, the industry would collapse." MathWorks *Amici* Br. at 12. But developers have long created interoperable software without permission of copyright holders, and the industry has not collapsed—it has thrived. Perhaps more than in any other field, innovation in the software industry depends on the freedom to create code that communicates and works with other technologies. Copyright protection should extend no further than absolutely necessary, or it will chill this innovation. Software developers rely on copyright exceptions to access ideas, extend functionality, and advance the state of the art without having to obtain permission from their competitors.

Developers rely on interoperability to create software that works not just on one computer, but on devices made by multiple manufacturers. Interoperability also spurs competition by encouraging developers to create in new and different ways, which is especially important for smaller companies and newcomers to the software industry to succeed. Interoperability ensures that developers can create software

efficiently and nimbly. They can begin with stable, established building blocks—including formal standards and widely adopted conventions such as protocols and user interface patterns—and focus their creative efforts on developing truly new features and capabilities. Interoperability also supports consumer choice, making it possible for individuals to choose the devices and platforms they prefer to use without sacrificing functionality, and discouraging consumer lock-in by established platforms.

All these market and consumer benefits depend on the freedom to access and use the ideas and functional elements embodied in software. Indeed, courts often point to innovation, compatibility, and interoperability as important factors when concluding that functional aspects of software are not protected by copyright.

For example, computer menu command hierarchies have been found to be unprotectable processes or methods of operation. *MiTek Holdings, Inc. v. Arce Eng'g Co.*, 89 F.3d 1548 (11th Cir. 1996); *Lotus Dev. Corp. v. Borland Int'l Inc.*, 49 F.3d 807 (1st Cir. 1995), *aff'd without opinion*, 516 U.S. 233 (1996). In other words, where a command hierarchy serves as the way one controls or makes use of a computer program's functional capabilities, that hierarchy does not qualify for copyright protection. *Lotus*, 49 F.3d at 815.

In reaching this conclusion, the First Circuit emphasized that newcomers must have access to methods of operation to innovate:

> "[B]uilding" requires the use of the precise method of operation already employed; otherwise "building" would require dismantling, too. Original developers are not the only people entitled to build on the methods of operation they create; anyone can.

Id. at 818. And extending copyright protection to a command hierarchy would make computer programs needlessly inefficient for consumers to use:

> Under Lotus's theory, if a user uses several different programs, he or she must learn how to perform the same operation in a different way for each program used. For example, if the user wanted the computer to print material, then the user would have to learn not just one method of operating the computer such that it prints, but many different methods. We find this absurd.

Id. at 817-18.

Even where courts find that elements of software are copyrightable expression, affirmative defenses may justify infringement necessary to achieve compatibility. As the Ninth Circuit has found, fair use permits the unauthorized copying of a competitor's software for the purpose of learning the functional requirements for compatibility. *Sega Enterprises Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1522 (9th Cir. 1992), as amended (Jan. 6. 1993); *Sony Comput. Entm't, Inc. v. Connectix Corp.*, 203 F.3d 596 (9th Cir. 2000) (both applying fair use). The Ninth Circuit has specifically noted that the ability to access functional elements in creative works serves the public interest by encouraging others in the same market to innovate: "It

is precisely this growth in creative expression, based on the dissemination of other creative works and the unprotected ideas contained in those works, that the Copyright Act was intended to promote." *Sega*, 977 F.2d at 1523.

The Ninth Circuit has also declined to constrain developers to pursue "the least efficient solution," noting that "wasted effort" is a harm that "the proscription against the copyright of ideas and facts . . . [is] designed to prevent." *Sony*, 203 F.3d at 605, quoting *Feist Pubs., Inc. v. Rural Tel. Serv. Co.*, 499 U.S. 340, 354 (1990). Forcing developers to perform their work inefficiently would "erect an artificial hurdle" that impedes access to the ideas embodied in software programs. *Sony*, 203 F.3d at 605.

These policy foundations remain sound and relevant in this case. Copyright law should not give the first to incorporate a functional concept in software a monopoly over that idea. Developers and interface designers should be free to re-use existing functional aspects of CLIs so they are not forced to reinvent the basic terms of communication between users and computers each time they create something new. When developers can rely on fundamental commands that have been used before, they are able to work efficiently, focusing their attention on elements of their work that truly require originality. Users also benefit because they do not have to learn how to perform the same the same operation a different way each time they use a different program, but instead can draw on their existing knowledge.

B. To the Extent They Are Copyrightable, CLI Elements or Compilations Are Scenes a Faire When Dictated by External Factors.

At issue in this case is Cisco's copyright in the selection, arrangement, organization, and design of CLI commands that people use to communicate with switches and routers—not the individual expressions themselves. Cisco Br. at 1. The selection and arrangement of otherwise unprotected elements may be sufficiently original itself to qualify for copyright protection. *Feist*, 499 U.S. at 348. But when it is, protection in such compilations is "thin." *Id*. at 349.

Assuming that a compilation of CLI commands is original enough to qualify for copyright protection at all (as the jury did find in this case), it is important to ensure that affirmative defenses remain robust to preserve interoperability and the capability to create new technology.

The purpose of the scenes a faire doctrine is to ensure the public can use the "necessary incidents" of ideas to strike the "balance between competition and protection." *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1253 (3d Cir. 1983); *Computer Assocs*., 982 F.2d at 711. The doctrine denies protection to the elements of a computer program that were dictated by external factors at the time the work was created. *Oracle Am., Inc. v. Google, Inc.*, 750 F.3d 1339, 1363-64 (Fed. Cir. 2014); *Mitel,* 124 F.3d at 1375.

Cisco contends that the scenes a faire doctrine denies copyright protection to elements of a program dictated by external constraints such as "the mechanical specifications of the computer on which a particular program is intended to run" or "widely accepted programming practices within the computer industry." Cisco Br. at 26 (citing *Oracle,* 750 F.3d at 1363). But courts have defined the doctrine more broadly than that. Scenes a faire includes elements of a computer program dictated by hardware standards,[2] software standards,[3] compatibility requirements,[4] computer manufacturers' design standards,[5] industry demands,[6] target industry practices,[7] customer demand,[8] market factors,[9] and functionality.[10] Indeed, according to this Court, scenes a faire excludes from copyright protection any expression that "flowed naturally from considerations external to the author's creativity." *Oracle,* 750 F.3d

[2] *Mitel,* 124 F.3d at 1375; *Gates Rubber*, 9 F.3d at 838.

[3] *Mitel,* 124 F.3d at 1375; *Gates Rubber.*, 9 F.3d at 838.

[4] *Computer Assocs.*, 982 F.2d at 710; *Mitel,* 124 F.3d at 1375; *Gates Rubber*, 9 F.3d at 838.

[5] *Computer Assocs.,* 982 F.2d at 710; *Mitel,* 124 F.3d at 1375; *Gates Rubber*, 9 F.3d at 838.

[6] *Computer Assocs.,* 982 F.2d at 710; *Mitel,* 124 F.3d at 1375.

[7] *Gates Rubber*, 9 F.3d at 838.

[8] *Mitel,* 124 F.3d at 1375.

[9] *Plains Cotton Coop. Assoc. v. Goodpasture Serv., Inc*., 807 F.2d 1256, 1262 (5th Cir.1987).

[10] *Mitel,* 124 F.3d at 1376.

at 1364 (quoting Melville B. Nimmer & David Nimmer, NIMMER ON COPYRIGHT § 13.03[F][3] at 13-131 (1997)).

Thus, a wide range of externalities are relevant to a software scenes a faire analysis, which makes sense for an industry in which creative choices are tempered by a host of real-world limitations—which may include the need to make a program or interface interoperable with other technology. Scenes a faire is intended to ensure that "copyright rewards and stimulates artistic creativity in a utilitarian work in a manner that permits the free use and development of non-protectable ideas and processes that make the work useful." *Mitel,* 124 F.3d at 1375 (quoting *Computer Assocs.,* 982 F.2d at 711 (internal quotation marks omitted)). Thus, relevant considerations for a scenes a faire analysis might include a developer's decision to use commands in which target users or other developers are already fluent in order to align with user expectations and industry demands.

II. The Ability to Re-Use Command Line Interface Commands Is Critical for Innovation in the Computer and Software Industry.

If commands to instruct computers to carry out certain functions are eligible for copyright protection at all, *Oracle*, 750 F.3d at 1367, the scenes a faire doctrine should consider the full range of externalities that affected a developer's decision-making to allow for innovation and interoperability.

Consider, for example, GNU Core Utilities, a software package that provides basic command-line tools for GNU/Linux, one of the most widely used operating

systems in the world. GNU Operating System, *GNU Coreutils* at 1.[11] GNU Core Utilities was largely designed to comply with the Portable Operating System Interface (POSIX), a family of standards for UNIX-like operating systems intended to maintain compatibility between different computing environments. *Id*. at 2.13.[12] POSIX reflects a consensus of technology manufacturers, designers, developers and others that core command-line options should follow the same conventions, even though different vendors create different implementations. *See* Institute of Electrical and Electronics Engineers and Open Group, IEEE Standard 1003.1-2008 12.1 (2016).[13]

But POSIX is not the only external factor that affected the development of GNU Core Utilities. The package has also been influenced by user and developer expectations. The tools make it easy for users accustomed to UNIX to quickly and easily adapt to the GNU/Linux operating system without learning new commands. And GNU Core Utilities has been designed to facilitate interoperability. Developers can port GNU/Linux tools to different systems, which opens up new development opportunities.

---

[11] https://www.gnu.org/software/coreutils/manual/coreutils.html#Introduction (last visited Dec. 29, 2017).

[12] https://www.gnu.org/software/coreutils/manual/coreutils.html#Standards-conformance (last visited Dec. 29, 2017).

[13] http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap12.html (last visited Dec. 29, 2017).

For instance, BusyBox and Toybox are each small-scale implementations of GNU Core Utilities in small, single executable files that are particularly suitable for computer systems with limited resources. BusyBox, *BusyBox: The Swiss Army Knife of Embedded Linux*;[14] Toybox Home Page.[15] They make it possible for GNU/Linux tools to function in a different operating system while maintaining the same interface and behavior that has become standard for those tools, creating development possibilities in computing environments where there otherwise would be none. To the extent that elements of GNU Core Utilities or any other command-line tool are designed to comply with standards, align with user and developer expectations, or be compatible with new systems, those elements should be scenes a faire.

Another strong scenes a faire candidate would be the command options (or flags) from the GNU Compiler Collection (GCC), a code compiler for Unix-like systems. *See* GNU Operating System, *GCC, the GNU Compiler Collection*.[16] GCC expanded a pattern of flags set for earlier C compilers for UNIX, which were originally written by engineers at Bell Labs. *Compare* K. Thompson & D. M. Richie, *Unix Programmer's Manual* CC(I) (5th ed. 1974) (discussing c, p, O, S, and P flags),

[14] https://busybox.net/about.html (last visited Dec. 29, 2017).

[15] https://landley.net/toybox (last updated Oct. 12, 2017).

[16] https://gcc.gnu.org (last updated Dec. 18, 2017).

with GNU Operating System, *GCC Option Index*[17] (listing all GCC command line options, including c, p, O, S, and P flags). GCC is designed to be compatible with a variety of standards, and its command line options can be used with multiple languages. GNU Operating System, *Language Standards Supported by GCC*;[18] GNU Operating System, *GCC Command Options*;[19] William von Hagen, THE DEFINITIVE GUIDE TO GCC 5-6 (2d. ed. 2006).

Other compilers are designed to support GCC flags, as well. For example, LLVM is a compiler for the C language family built around a set of libraries, and it has widespread support in the technology industry. Apple, *LLVM Compiler Overview.*[20] LLVM has been designed so that its front end, Clang, supports extensions from a range of programming languages, as well as many extensions from GCC. *Clang Language Extensions*;[21] *Clang 6 Documentation: Clang*

---

[17] https://gcc.gnu.org/onlinedocs/gcc/Option-Index.html#Option-Index (last visited Dec. 29, 2017).

[18] https://gcc.gnu.org/onlinedocs/gcc/Standards.html#Standards (last visited Dec. 29, 2017).

[19] https://gcc.gnu.org/onlinedocs/gcc-2.95.2/gcc_2.html (last visited Dec. 29, 2017).

[20] https://developer.apple.com/library/content/documentation/CompilerTools/Conceptual/LLVMCompilerOverview/index.html (last updated Dec. 13, 2012). LLVM's sponsors include Apple, Google, Intel, *amicus* Mozilla, and Facebook. *LLVM Foundation Sponsors*, http://foundation.llvm.org/sponsors.html (last visited Dec. 29, 2017).

[21] http://clang.llvm.org/docs/LanguageExtensions.html (last visited Dec. 29, 2017). A compiler's front end analyzes source code so that it can be processed and transformed into object code by the compiler's back end.

Command Line Argument Reference (listing command line arguments supported by GCC-compatible Clang drivers).[22] To the extent GCC flags have been dictated by past industry practice and standards, those elements should be scenes a faire, and others should be able to use them freely.

Finally, the Court should consider efficiency a relevant externality for purposes of a scenes a faire analysis. For example, millions of engineers use a client called npm to access a global registry of shared JavaScript packages. (The npm client itself borrows from existing CLI conventions, and those elements should be scenes a faire.)[23] Several collaborators created a new client, Yarn, to enable engineers to install packages from the registry more quickly than they can with npm, manage dependencies across multiple computers, and install certain packages when offline. Facebook Code, *Yarn: A New Package Manager for JavaScript* (Oct. 11, 2016).[24] The Yarn CLI replaces npm's CLI using matching or similar commands. *Id*. As a result, users who are accustomed to npm can download packages from the registry through Yarn without learning new commands or expressions, making it easy to use

[22] https://clang.llvm.org/docs/ClangCommandLineReference.html (last visited Dec. 29, 2017).

[23] *Compare* npm, *npm commands*, https://docs.npmjs.com/cli/ls (last updated Nov. 3, 2017) ("npm ls" command lists installed packages), *with Unix Programmer's Manual* at vii ("ls" command lists contents of directory).

[24] https://code.facebook.com/posts/1840075619545360.

and adapt to. The elements of Yarn's CLI that are dictated by these efficiency considerations should be scenes a faire.

Another example of a command-line tool dictated by efficiency is a wrapper: a computer program that has a command embedded in it. A wrapper serves as a coding shortcut to orchestrate a complex command by simplifying or consolidating certain complex operations. *Amicus* GitHub, for instance, offers a wrapper that generally aims to simplify certain tasks while remaining compatible with Git, a and a project of *amicus* Software Freedom Conservancy that tracks changes in software files and helps to coordinate efforts by multiple developers. *See* GitHub, *hub*.[25] In other words, the purpose of wrappers is to save time and effort. They should be scenes a faire because they are developed due to efficiency considerations.

A finding that scenes a faire must be narrowly limited as a matter of law would be at odds with the fundamental purpose of copyright law: to promote innovation. *Amici* urge this Court to ensure the law allows flexibility for lawful uses of software, in turn fostering innovation.

[25] https://hub.github.com (last visited Dec. 29, 2017). Even though Git is a charitable project of Software Freedom Conservancy and GitHub is a for-profit company, both organizations have signed onto this brief because interoperation of their technology best serves the public and should continue unfettered.

CONCLUSION

Amici respectfully ask this Court to uphold the jury verdict in Arista's favor and affirm the district court's judgment.

Dated: December 29, 2017

Respectfully submitted,

/s/ Marcia Hofmann
Marcia Hofmann
Zeitgeist Law PC
25 Taylor Street
San Francisco, CA 94102
Telephone: (415) 830-6664
marcia@zeitgeist.law

Counsel for Amici Curiae

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE REQUIREMENTS

I certify that the foregoing brief complies with the type-volume limitation of Federal Rules of Appellate Procedure 29(a)(5) and 32(a)(7), as well as Federal Circuit Rule 28.1. This brief contains 3,624 words, excluding the parts of the brief exempted by Federal Rule of Appellate Procedure 32(f).

This brief's type size and typeface comply with Federal Rule of Appellate Procedure 32(a)(5) and (6) and Federal Circuit Rule 28.1. It was written in Times New Roman proportionally spaced typeface with 14-point font.

/s/ Marcia Hofmann
Marcia Hofmann

CERTIFICATE OF SERVICE

I certify that I filed and served the foregoing *amici curiae* brief on December 29, 2017, via the Court's CM/ECF electronic filing system.

I certify that all participants in the case are registered CM/ECF users and that service will be accomplished by the appellate CM/ECF system.

/s/ Marcia Hofmann
Marcia Hofmann