

Reproduced with permission from The United States Law Week, 82 U.S.L.W. 856, 12/10/13. Copyright © 2013 by The Bureau of National Affairs, Inc. (800-372-1033) <http://www.bna.com>

Evidence

Source Code

The ability to persuasively discuss source code is becoming a necessity in many cases, and the authors suggest that attorneys should consider the significant benefits of its effective presentation at trial. While either a native version or printout of the code, or a combination of the two, should be presented, the authors suggest that a hands-on, engaging expert, who can teach the jury about the code, is essential to reaping the benefits of its introduction at trial.

From the Computer to the Courtroom: The Evolving Role of Source Code in Complex Trials



By SETH A. NORTHROP AND LI ZHU

The modern computer is one of the world's most influential inventions. Today, computer microprocessors invade almost all forms of technology, including personal electronics, automobiles, laboratory equipment, and medical devices. The natural consequence

Seth Northrop and Li Zhu are trial attorneys at Robins, Kaplan, Miller & Ciresi LLP. They focus their intellectual property litigation and complex business litigation practices primarily on large-scale disputes involving a variety of technologies. sanorthrop@rkmc.com and lzhu@rkmc.com.

for trial attorneys is that computer software and firmware has also been invading an ever-expanding number of practice areas. While it may come as no surprise that patent infringement and trade secret cases now routinely involve software disputes, we are also seeing software disputes crawl into areas such as products liability, business torts, and even criminal cases.

While software disputes permeate many different areas of law, they have one thing in common: the use of "source code." Source code is a collection of programming instructions that can be read and understood by a human programmer. This source code is later translated (or "compiled") to "machine language" that the computer reads and uses to accomplish (or "execute") specified tasks. To most lawyers, dealing with source code can be a daunting task. There is, however, untapped potential for attorneys who can effectively navigate source code and present such evidence in a compelling fashion in the courtroom. This article examines key points a trial attorney should consider before presenting source code to a jury.

I. Should I Use Source Code at Trial?

Obviously, you first need to decide if source code is helpful to your case. Are the specific instructions executed by the software a necessary or helpful element to your cause? In patent and trade secret cases, the an-

swer to this question is often fairly obvious. In many other cases, the observable use of the software may be sufficient to prove your point. You may be concerned mainly with the *results* of the operation of the software. This could be a situation in which you wanted, for example, to show the jury that the execution of the software causes a repeatable error. But even in these cases, there may be a distinct advantage to “looking under the hood” of the software by analyzing the underlying source code.

A critical element of persuasion is revealing *why* something happened. People, jurors in particular, typically search for the “why.” Source code can often supply the answer. Careful consideration should be given to whether the underlying computer code can persuasively answer the question of *why* the software operated in the manner it did, even if it is not strictly necessary to prove the case. Given the high cost and complexity of source code discovery, you will have to weigh those benefits early in the case. As the Court of Appeals for the Federal Circuit has explained, a single version of source code may comprise millions of lines of text on paper.¹ The relevant information can be the proverbial needle in a haystack.

In weighing the potential benefits of introducing code, it is important to remember that source code is usually something that jurors do not understand. Too often, however, trial attorneys rely on this as the sole reason to exclude source code from the trial arsenal. While it is certainly true that jurors see source code as an alien language, it would be shortsighted to miss the potential benefits related to that fact. As discussed in more detail below, source code presents a chance to “teach” the jury. While teaching a jury about source code walks the fine line between interesting and boring, a charismatic expert may engage the jury in a surprising fashion through a skilled presentation.

II. How Do I Introduce Source Code Into Evidence?

Once you decide to rely on source code at trial, you will have to grapple with the mechanics of actually introducing the code into evidence. There are generally two ways that source code can be introduced into evidence. First, an expert can indirectly introduce the source code by providing an explanation of his or her review of the code. Even if the source code itself is not admitted into evidence, the expert, subject to cross-examination, will likely be able to discuss his or her opinions regarding the operation of the source code under Federal Rule of Evidence 703. Second, the source code itself can be introduced into evidence.

Practically speaking, the introduction of source code through expert testimony is often the most common and most important evidence relating to source code, because lay jurors will not likely be able to read and understand source code by itself. The actual *presentation* of this code is more challenging and is addressed in the next section. The procedure for introducing expert testimony regarding code is fairly straightforward and likely understood by most trial lawyers. The process gets tricky, however, when introducing the source code itself into evidence.

¹ *Lucent Techs., Inc. v. Gateway, Inc.*, 580 F.3d 1301, 1332 (Fed. Cir. 2009).

If you decide to admit the underlying source code itself into evidence, it is important to first consider the format in which you will submit the code to the court: printouts or native format. Printouts and native versions of source code each have advantages and drawbacks. Many trial lawyers and judges without programming experience will assume that simply printing out source code will suffice. After all, source code files are simply text files that can be printed out with corresponding line numbers, just like a regular word processing document. Sometimes, this is true. In these cases, a printout can be reviewed by a deliberating jury without need for a computer or an understanding of how to navigate electronic versions of code. This could be particularly useful if the jury is comparing specific lines of source code for similarities, as in a copyright or trade secret case, or when a jury is looking at certain comments made in the source code itself.

The problem with relying on printouts of source code is that the vast majority of software programs rely on many, sometimes thousands, of separate source code files. Printing out one source code file may not tell an expert even a fraction of how the software would perform when it is executed. Even very discrete software tasks may require walking through several different source code files. While well-organized printouts of the source code may be manageable on direct examination, cross-examination could be incredibly hard to follow and laborious without using the source code in its native format. Consequently, best practice dictates that a native version of the source code should at least be available at trial, if not introduced into evidence.

Best practice dictates that a native version of the source code should at least be available at trial, if not introduced into evidence.

The safest course of action would be to introduce a combination of both native files and printouts. Native format documents can be encrypted and placed on an electronic media such as an external hard drive, thumb drive, or optical disc for submission to the court. This solution allows the witness to testify from a complete and navigable version of source code in admitted form. Then, for those specific files that may be particularly important to send to the jury room or quote in a post-trial or appellate brief, the proponent of the evidence can print out a copy and submit that as evidence.

When introducing printouts of source code, format matters. Many different types of software allow you to print source code in color-coded form so that experts, the jury, and the court can easily identify key components of the software, including function names, variables, and source code comments. Likewise, many types of software ensure that source code is printed with consistent indentation. Although on its face this may seem inconsequential, readability of the software can be the difference between the jury/judge understanding how the source code operates or getting lost in the complexity of the code.

Sometimes, one side may object to the submission of native versions of source code based on confidentiality

concerns. In those instances, it is prudent for the parties to attempt to agree, before trial, on how the information will be admitted. One solution is to allow the use of the native version of source code as a demonstrative at trial, and then allow the proponent to submit as evidence printouts of any of the code that was discussed by the witness on direct or cross examination. Even if the parties agree, it is a very good idea to address these concerns and possible solutions with the court during the pre-trial hearing.²

III. How Should I Present the Source Code at Trial?

Of course, the key for a trial lawyer is determining how to present source code to the fact-finder in a persuasive manner. The most skilled trial lawyers will be able to capitalize on the opportunity to teach a jury about source code without boring them with too many mundane or complex details. The presenter must strike a balance by finding the right mix of show-and-tell. Some trial lawyers believe that trying a case is one part science and two parts art. As such, this section cannot provide a formula for success. It does, however, provide a few useful tools for practitioners to draw from when preparing their presentations.

Obviously, choosing the right expert witness is critical. No matter the practice area, cases are often won or lost based on expert testimony. Thus, counsel must select an expert that will resound with the jury—an expert who can both teach and persuade. Given the complicated nature of source code, it is crucial that the expert not be merely a “talking brain” in the courtroom. To be persuasive, the expert must be “hands on” and should, if possible, manipulate the source code through demonstrations to effectively explain the subject matter.

There are various techniques that can be used for presenting code. First and foremost, it is important to give the jury a concrete understanding of what the code actually does. This should usually involve, if at all possible, a demonstration of the result(s) of executing the software created by the source code. This likely requires a demonstration, in the courtroom, of how a product works when its software receives certain inputs or commands. This allows the jury to conceptualize the observable results of executing the code.

One technique is to supplement, or even replace, the in-court demonstration with a previously recorded demonstration that can be admitted into evidence. For example, screen capturing software can be used to capture a video recording of everything that a user does on a computer screen. This allows an expert or deponent to input various commands into a software program and show the result of those commands on the screen. Similarly, a witness could video-record the operation of a product or mechanical device to show how the software operates in practice. Rules 901(b)(3) and (b)(9) can allow the introduction of such evidence, which provides a significant advantage over a demonstrative.

The same screen capture technology can be used during adverse depositions and presented at trial so the

jury can see specific code being talked about by the witnesses. If a trial attorney can send video evidence of the operation of software back with the jury during deliberations, this will likely result in a significant persuasive advantage as compared to the mere demonstration of the product in the courtroom.

Once a demonstration of the observable execution of the software has been presented, it is time for the expert to focus on the particular source code that may help your case. The most important point in this process is to keep the jury engaged with visuals. If the witness takes more than five minutes discussing the code alone, without visual or physical demonstrations to supplement the discussion, then you are likely boring the jury.

One suggestion, in a case involving a significant amount of code, is for the expert to take a brief opportunity to introduce the very basic concept and operation of source code to the jury. For example, the expert can explain the purpose of source code and prepare a very simple and short example of how source code works. In computer programming tutorials, this demonstration typically involves writing a less than five-line source code file that, when executed, writes “Hello World” to the computer screen. In a matter of minutes, an expert could teach the jury the very simple steps that a programmer would take to write and execute the code in order to produce a tangible result. A basic presentation, as described, helps seize an opportunity to teach jurors how source code works in an easy to understand way. Most importantly, your expert witness has an opportunity to teach, which can pay huge dividends in terms of gaining credibility with the jury.

**First and foremost, it is important to give the jury
a concrete understanding of what the code
actually does.**

After a basic presentation, it will be necessary to direct the expert to a discussion of much more complicated lines of code. Practitioners should consider how they will incorporate visuals into each step of the source code testimony. The graphic could simply be a demonstration of a certain task performed by the product. Alternatively, the graphic could be an animation that allows the jury to visually understand what is happening when the software executes a particular function from the source code. This can involve a depiction of, for example, information flowing from one component within a product to another.

Even a simple PowerPoint slide can assist in reducing the complexity of the code. If you can integrate enough visuals with step-by-step expert testimony regarding how the source code works, then you will be able to maximize the perception that the expert’s analysis was thorough, credible, and impressive without lulling the jury to sleep.

If the trial involves many different products that contain different, albeit similar, software, it may make sense to use a Federal Rule of Evidence 1006 summary. The expert can detail how he or she reached his or her opinions using a single device. Then, the expert can in-

² *Kelora Sys., LLC v. Target Corp.*, No. C 11-01548 CW (LB), 2011 U.S. Dist. LEXIS 96724, at *19 (N.D. Cal. Aug. 29, 2011) (stating that source code is “highly confidential, technical information”).

roduce a Rule 1006 summary³ that identifies the corresponding code that performs a similar or equivalent function in other devices. This allows the expert to demonstrate to the jury the precise basis for the opinion while limiting the amount of repetitive and highly-detailed source code testimony that the jury needs to consider.

³ See Freidman and Boone, COMPUTER FORENSICS ONLINE, Vol. 1, No. 1, *Old Rules for New Stuff: Pretrial Discovery and Seizure of Computer Based Evidence* (Dec. 1997), available at http://www.shk-dplc.com/cfo/issue_01/rules.html.

IV. Conclusion

As much as lawyers may want to hide from presenting source code, the ability to persuasively discuss source code is becoming a necessity in many cases. Trial attorneys would be well served to consider the significant persuasive benefits to the effective presentation of source code at trial. To that end, more trial attorneys should consider, at the beginning of their case, how they can take advantage of presenting source code at trial and how they can execute that strategy.