Source Code: Receiving, Reviewing and Producing

LLM LIQUID LITIGATION MANAGEMENT, INC. | Logical Solution. Legendary Service.

In the past several years, the court system has seen an influx of legal matters associated with the production and review of source code. These cases can be civil or criminal in nature, but often involve litigation related to patent and copyright infringement.

In the *Apple v. Samsung Electronics Co., Ltd. (*No. 05:11-CV-01846 LHK PSG, 2012 WL 1595784 N.D. Cal. May 4, 2012) patent infringement case, Judge Paul Grewal summarized the complexities of source code production when he stated, "Put simply, source code production is disruptive, expensive, and fraught with monumental opportunities to screw up." One of the reasons that source code production can be so disruptive is simply the magnitude of source code that can exist for a given program or device. For example, the Windows Vista operating system contains approximately 50 million lines of source code — imagine how that has increased.[1] Because source code can be so immense, the process of reviewing and producing source code and hiring the necessary experts can quickly become very cost-and-labor intensive.

As both patent applications and patent case filings drastically increase, it is important that counsel realize the potential exposure to patent litigation and related source code production. In 2012, the U.S. Patent and Trademark Office reported an 83 percent increase in patent applications from the year 2000, not including reissues. The number of utility patent grants[2] made in 2012 reached 253,155, which accounted for 91.46 percent of the total grants made that year.[3] Furthermore, in 2011 the number of court filings related to patent litigation reached a record number of 4,015.[4] The stakes are high when it comes to patent litigation, and the need to produce source code often closely follows.

Once upon a time, discovery processes involved paper. Now, they are native and involve filtering and searching. Risk management and analysis includes litigation hold and custodian interviews, visualization and analytics as well as metrics to measure progress and risk. Cost managment includes cost sharing. The role of communication and cooperation has also changed: internally between departments and employees, externally with outside counsel and vendors and in the courts.

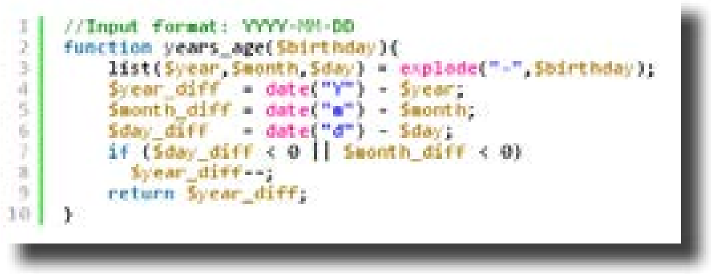
This white paper is intended to provide counsel with a better understanding of the complexities of source code production, why and how it is produced, examples of relevant court cases and some best practices to ensure that these legal matters are as time- and cost-effective as possible.

WHAT IS SOURCE CODE?

A computer program is really just a set of instructions that tells the computer what to do, and source code is the human-readable representation of those instructions. Source code is also used beyond computers in hardware devices, such as cars, mobile phones, printers and even televisions. It can be written in hundreds of different scripting and programming languages, such as Java, C, and Python. The type of programming language used varies based on the functionality of the program or device, and what program qualities are emphasized. For example, some languages are better suited for programs that emphasize transactional speed while others are better for programs that use graphical interfaces. However, the same program can be written in many different ways using many different languages.

This example of source code would allow the user to enter a date to determine someone or something's age.

Depending on the programming/scripting language, the source code can be translated in a variety of ways into operations the computer understands. One of these ways is a compilation of binary code called object code. Object code is not easily readable by humans. For this reason, when court matters arise requiring the review of a program or product feature, source code is isolated, collected, and produced. The following example of source code, written in the PHP language, would allow the user to enter a date to determine someone or something's age.



```
//Input format: YYYY-MM-DD
function years_age($birthday){
    list($year,$month,$day) = explode("-",$birthday);
    $year_diff  = date("Y") - $year;
    $month_diff = date("m") - $month;
    $day_diff   = date("d") - $day;
    if ($day_diff < 0 || $month_diff < 0)
      $year_diff--;
    return $year_diff;
}
```

HOW IS SOURCE CODE RELEVANT TO LITIGATION?

The most common types of cases where source code production is necessary are patent infringement, copyright infringement, and trade secret misappropriation. Parties are required to produce source code to opposing counsel in order to determine whether alleged infringement or misappropriation claims are accurate. As these cases become more prevalent, so do the instances where source code production is court-mandated. While the production of source code occurs often in patent and copyright infringement cases, it can arise in a variety of legal matters. Consider the following two cases.

SOURCE CODE BEYOND PATENT AND COPYRIGHT MATTERS

In 2012, a Minnesota Supreme Court criminal case disputed the accuracy of the Intoxilyzer 5000EN, a device which calculates the blood alcohol content in a person's breath. The source code for this device operates every function of the device, from ensuring "fail safes" are performed to determining alcohol concentration in the sample. A group of private criminal attorneys argued that inaccuracies in the machine's source code made its output unreliable and therefore inadmissible in court. The court required source code be produced and reviewed to determine whether or not this claim was true. After producing and having forensic experts analyze the source code, the judge was able to rule that although the device was "severely challenged" and the source code contained several errors, that overall it was sufficiently accurate.

The case of *Metavante Corp v. Emigrant Savings Bank, (*No. 05-CV-1221, 2008 WL 4722336 E.D. Wis. Oct. 24, 2008*)* was a breach of contract matter in which defendant Emigrant Bank outsourced Metavante Corp to help the bank launch an online, direct banking system. Metavante sued Emigrant for nonpayment of fees. Emigrant argued that the system put in place by Metavante was, "poorly integrated, poorly tested, poorly planned, not scalable and experienced degraded service." Emigrant was successful in arguing that it needed to review source code related to the online banking system in order to accurately defend itself against the breach of contract claims. Metavante argued that it would be unduly burdened by having to produce source code and estimated that isolating its source code would cost over $300,000, take over

5,000 hours and that the source code would not provide Emigrant with sufficient relevant information. The court balanced the value of the source code against the burden of producing it and determined that the value outweighed the burden. In this case, the judge ruled in favor of the plaintiff and ordered Emigrant pay Metavante damages and prejudgment interest in the amount of over $2 million.

WHY DOES SOURCE CODE CAUSE CONCERN?

The mere mention of source code and its production can cause some counsel concern. The reasons can include a lack of familiarity ("How do I use this? Report on this? Retrieve information from this new form of data?"); complexity and structure (new data is often more complex and highly structured); impermanence (data changes quickly over time); authentication and identification (new technologies often work to provide anonymous access); and confidentiality, security and privacy.

WHY IS SOURCE CODE SUCH A HASSLE TO PRODUCE?

Source code can be costly and time consuming to produce for a variety of reasons that depend on the program or device at hand and the nature of the case. Some of the most common issues that cause source code production to become problematic are:

- The magnitude of source code that can exist — possibly millions of lines of code and thousands of files.
- The review of source code more often than not requires the work of forensic experts or highly skilled computer programmers that require high compensation.
- The collection of source code requires that structure and format of the code is maintained so that forensic experts can comprehend and recreate the actions taken by the code when necessary.
- The isolation of source code can take many hours due to the necessity of ensuring completeness of the code and abiding by all elements required by the court order.

SOURCE CODE PRODUCTION HAS BEEN MANDATED: WHAT NEXT?

If the situation arises in which a party is required to produce source code, there are some vital initial steps to take. Have a timely and detailed discussion with the IT department or the programmers who create and manage the source code related to the product or program in question. These individuals would most likely be able to provide counsel with a good idea of the magnitude and man hours that would be required to review, isolate, and collect source code. An important next step would be to discuss whether an expert would be well utilized in the process. Once counsel has a good idea of the task in front of them, it is important that counsel has open communication with both the court and opposing counsel to make sure any plans going forward are exactly what is being asked for, so as not to waste valuable resources.

WHAT IS THE GOAL BEHIND PRODUCING SOURCE CODE? WHO TYPICALLY REQUESTS IT? WHO REVIEWS IT?

The goal behind source code production is to be able to examine the way code is constructed and what it does when executed. Once this is determined, the court can attempt to establish how the source code is relevant to the case at hand. Oftentimes, third-party forensic experts are hired to review source code to help less tech-savvy officers of the court understand exactly how the source code is used within the program or device. The purpose and method behind reviewing source code is largely dependent on the nature of the case and the allegations.

The following are examples of source code review techniques that are often associated with a variety of cases and claims:[5]

Copyright Infringement
Reviewing source code related to copyright infringement claims requires the analysis and comparison of two bodies of source code to determine whether there are any elements of protected (patented) source code that are the same or substantially similar. Copyright-protected elements include literal lines of source code as well as structure, sequence, and organization of source code. This is not necessarily a simple literal comparison, however, since there still may be copyright infringement if the alleged infringing work is a derivative work written in a different programming language.[6] This process often requires the use of specialized computer software such as CodeMatch, JPlag, or Moss because of the vast quantity of source code that can exist. The specific software used will depend on the type of code and language used, however, an integrated development environment is useful to import and review source code while maintaining the structure. This development environment allows the user to access multiple versions of the code as different projects or repositories. A version control software system, as discussed later, would further allow the analyst to merge or compare the source code.

Patent Infringement
The goal of source code review related to patent infringement claims is to determine whether the patent claims are infringed by the executed program's source code.

This requires a detailed understanding of functionality of patented inventions or methods and requires probing and thorough analysis of accused source code. In these cases, the parties must often bring in forensic experts who create an understanding of what each element of source code does when executed by the computer or device. This type of review can be very time-consuming as significant portions of reviewed source code are often irrelevant in hindsight.

Trade Secrets
Source code review related to trade secret misappropriation allegations usually results in a more targeted review of source code because only a specific portion of source code is being questioned. The goal is to determine whether the alleged trade secrets are used within source code. For these cases, the forensic expert should have a clear/detailed statement of what the trade secrets are and review the source code to determine if those trade secrets are used.

In this process, it is best for the defendant if the exact trade secret is identified before access to the source code is granted so that source code is not oversupplied and mining of company trade secrets cannot occur.

As discussed in the case examples, source code production is not limited to a certain type of litigation. Therefore, it is important that counsel and analysts have a clear understanding of the courts' intention behind requiring source code review and that they act accordingly.

HOW TO COLLECT AND PRODUCE SOURCE CODE?

Requests for source code can be made by the plaintiff, defendant or judge. The judge, however, is the only individual who may order the production of source code. Court-issued protective orders are usually the method by which source code production is compelled. The court often mandates how source code should be isolated and produced to the other side. It is important that these orders contain detailed requirements as to what types of files and code will need to be reviewed. In cases involving complex computer software or devices such as smartphones, source code review can mean the search for and review of thousands of interrelated files and lines of code. For this reason, the first step a forensic expert or analyst often takes is to locate the exact portions and files of source code that the court is requiring review of and isolate these files. It is vital that during this step, counsel confirms with the analyst or forensic expert that the isolated source code set is complete and fulfills the court order.[7] Usually source code is produced in the language in which it was originally written to help maintain readability and avoid any translation errors. Either way, the process requires someone who is knowledgeable about the program and its source code to conduct a comprehensive search for relevant material. Protective orders should also include provisions to discuss the means by which to reduce the risk of having source code leaked.

ARE THERE RISKS ASSOCIATED WITH PRODUCING SOURCE CODE? WHAT HAPPENS IF YOU DON'T PRODUCE SOURCE CODE?

Producing source code is inherently risky as parties are often turning over intellectual property to adverse parties who are frequently competitors. For this reason, many parties will do whatever it takes to circumvent having to produce source code. For example, many times companies will argue that their user manuals and the like are sufficient to show how the product operates for the purpose of patent infringement without disclosing confidential information. Oftentimes, the courts are forced to impose sanctions upon parties for producing incomplete source code or not producing it at all. To quote Judge Grewal once more, "In a typical patent infringement case involving computer software, few tasks excite a defendant less than a requirement that it produce source code. Engineers and management howl at the notion of providing strangers, and especially a fierce competitor, access to the crown jewels." The following are examples of cases that involve sanctions the court put in place because of a lack of properly produced source code.

The *Apple v. Samsung* patent trial — a civil case under the jurisdiction of the Northern District of California — is a good illustration of the types of source code sanctions that can be put in place. Because Samsung refused to comply with court orders to produce source code and kept the court waiting for months, Judge Grewal ruled that Samsung would be precluded from offering any evidence of its design-around efforts for certain patents and could not argue that the design-arounds are in any way distinct from those versions of code produced in accordance with the court's order. This means that Samsung essentially had to assume liability for continued use of old, possibly infringing code because the new, possibly non-infringing code wasn't shown in time. In addition to legal, case-related sanctions, Judge Grewal also imposed monetary sanctions on Samsung for lack of producing source code.

Another civil case under the jurisdiction of the Northern District of California in which source code sanctions were imposed is that of *Keithley v. Home Store.com* (No. 3:03-CV-04447 SI (EDL), 2008 WL 3833384, N.D. Cal., Aug. 12, 2008). In this case, the plaintiffs alleged their patent on a system for acquiring and tracking real estate information had been infringed by defendants through their real estate-related websites. Plaintiffs argued that defendants had destroyed several key items, including source code; early architectural, design and implementation documents; and reports. The plaintiffs contended that the spoliation of these materials had an impact on their ability to meet their burden of proving infringement. The court awarded plaintiffs $148,269.50 in sanctions to reimburse their attorney fees and costs in seeking sanctions for spoliation. Additional sanctions against defendants included awards in amounts to be determined for plaintiffs' fees and costs in re-doing their litigation preparation with the eventual production of documents estimated at $862,125.

These two cases are good examples of the importance of producing source code in a timely and appropriate manner as required by the court. Violating court mandates for source code production can not only lead to monetary sanctions but also litigation disadvantages, as the Samsung case illustrates.

COURT STANDARDS FOR SOURCE CODE PRODUCTION

In discussing source code production in the *Apple v. Samsung* matter, Judge Paul Grewal made the following statements: "Counsel struggles to understand even exactly what code exists and exactly how it can be made available for reasonable inspection. All sorts of questions are immediately posed." Several courts have taken a proactive approach to source code production and implemented standards to help counsel understand what is expected of them. The Northern District of California was one of the first courts to institute source code procedures followed by the Eastern District of Texas and the District of Delaware. These courts have the following similar requirements:

1. Copies of source code will be made available for inspection on standalone computers at a third-party site or mutually agreed upon location.
2. The standalone computer will contain the appropriate software necessary for counsel and experts to view, search, and analyze the source code.
3. The receiving party will make a reasonable effort to restrict its access to the stand-alone computers to regular business hours (8:00am to 6:00pm) unless otherwise agreed upon.
4. No electronic or hard copies of Source Code Material will be made without prior written consent of the producing party.

Each of these courts has also implemented the following requirements that are unique to their districts:

The Northern District of California

- The Producing Party may visually monitor the activities of the Receiving Party's representatives during any source code review, but only to ensure that there is no unauthorized recording, copying, or transmission of the source code.
- The Receiving Party may request paper copies of limited portions of source code that are reasonably necessary for the preparation of court filings, pleadings, expert reports, or other papers, or for deposition or trial, but shall not request paper copies for the purposes of reviewing the source code other than electronically.
- The Receiving Party shall maintain a record of any individual who has inspected any portion of the source code in electronic or paper form.

The Eastern District of Texas

- Access to Protected Material designated "Restricted Confidential - Source Code" shall be limited to outside cousel and up to three outside consultants or experts (i.e., not existing employees or affiliates of a Party of an affiliate of a Party.
- To the extent portions of Source Code Material are quoted in a Source Code Document, either (1) the entire Source Code Document will be stamped and treated as Restricted Confidential Source Code or (2) those pages containing quoted Source Code Material will be separately stamped and treated as Restricted Confidential Source Code.

U.S. District Courts for the District of Delaware

- The source code provider shall provide a manifest of the contents of the standalone computer. This manifest, which will be supplied in both printed and electronic form, will list the name, location, and MD5 checksum of every source and executable file escrowed on the computer.

- If the court determines that the issue of missing files needs to be addressed, the source code provider will include on the standalone computer the build scripts, compilers, assemblers, and other utilities necessary to rebuild the application from source code, along with instructions for their use.

These courts have attempted to strike the delicate balance of not making source code production overly burdensome while also allowing a substantial and efficient review to be completed. As the production of source code becomes more common in courtroom settings, it is likely that other district courts will use these courts' standards as guidance.

BEST PRACTICES FOR SOURCE CODE PRODUCTION

Planning: Meaningful and Timely Meet and Confer
Prior to the meet and confer, attorneys should check with the court to see if there are any standards in place for source code production. The meet and confer should occur early on in the litigation process to avoid any unnecessary delays. Any litigation holds that may be required should be discussed early on to avoid the destruction or loss of any potentially relevant source code. Attorneys should provide valid and logical reasons for needing the production of source code and should have reasonable time and cost expectations. In the meet and confer, proper date ranges and relevant software version numbers should be identified and finalized. In addition, counsel should consider using a third-party vendor that specializes in source code collection, review and expert testimony. Many of these vendors supply neutral offsite "viewing rooms," or even remote-login inspections that provide the necessary security requirements to comply with court orders.

It's necessary to emphasize here that collaboration is crucial. It's important to bring in the experts who understand the code and can review and explain it in a way the court will understand. It's also important to use meet and confers to address areas of concern and negotiate parameters. Courts are sensitive to the amount of complexities, time and costs involved with such cases.

If one can provide a workable alternative and solution to the court, and explain it in easy-to-understand, non-technical terms, the more likely one will be successful in defense or prosecution of the case.

In *Ameranth, Inc. v. Pizza Hut, Inc.*, (11-CV-1810, 2013 WL 636936 S.D. Cal. Feb. 20, 2013), Ameranth provides wireless and internet-based systems and services to third parties. This case had a discovery dispute in which the court anticipated a problem with the large amount of data to be produced and, as a result, had the parties brief the issue.

Ameranth wanted the court to order each of the 30 defendants to produce the entire source code tree for each accused product. The court recognized that this tremendous amount of source code would require a great deal of time and money.

The defendants argued that they should only have to produce the portion of the code that was relevant to the infringement. The defendants also argued that it would be more efficient to have their technical experts identify and provide the portions of the source code that specifically relate to the aspects or functionality identified by the plaintiff in its infringement contentions. They used the expert testimonies to clearly explain that analyzing non-relevant source code would waste time and money and delay the case. The court agreed and rejected the plaintiff's motion.

Keeping Costs to a Minimum
It is important for attorneys to take cost considerations into account early on and discuss any cost concerns with the court prior to starting the review of source code. The court's involvement in this discussion is crucial because in some cases the judge will have both parties contribute to the cost of source code review. In addition, counsel should have a concrete understanding of production requirements so as not to overproduce or spend excess hours reviewing source code that is not relevant to litigation.

Technological Considerations: Collection
Counsel should make sure the IT department or service provider knows where source code is stored and how it can be collected completely and efficiently. In the event the court mandates a third-party review of source code, counsel should be prepared to hire an outside forensic expert.

Creating / Receiving a Complete Source Code Production
Once the initial review process has been completed and relevant source code has been determined, the process of creating a production to provide opposing counsel begins.

Receiving source code in a printed or imaged (TIFF or PDF) format will not be useful for analysis. Any format other than the original source code tree in digital form will increase the cost of analysis. It is usually most cost-effective, as well as easiest for the producing party, to produce the entire source code tree as maintained in the version control system. This is often referred to as revision control or source control system.[8] Additionally, pursuant to Fed. R. Civ. P. 34(b)(2)(E)(ii), the source code tree best represents the electronic structure "in which it is ordinarily maintained or in a reasonably usable form or forms." Another option for production would be to have the producing party provide a laptop that would contain all supporting material required to review the production, such as development tools, database systems, and source code control systems. The party receiving the source code should request the exact software package supplier, name and version of the control system to confirm that upon receipt of the production, it will have the technological capability to review the files.[9] In some cases, this can save both sides time as the requesting party does not have to bombard the producing party with questions about setup and configuration.

It is important that the producing party determines completeness of source code before handing it over to opposing counsel. This will be specific to the scope of the dispute. If relevancy and scope encompasses a full and complete program, then one way to check completeness would be to load the production onto a configured computer and rebuild an executable version of the program. If the scope is more narrow and requires only sections of the source code to be produced, then counsel needs to ensure that the agreed-upon portion is complete and covers all the pieces under dispute. In some cases, several versions of the source code may need to be examined to determine what versions, if any, contained infringing elements. Dr. Lee Hollaar, computer law professor at the University of Utah in Salt Lake City, recommends that utility programs such as Microsoft's WinDiff be used to compare source code files in a directory structure in order to determine which files may have changed between versions as well as specific lines in the file that may have changed.

In addition to the code itself, other files that might be required or useful to produce include program documentation such as supporting design documents, user manuals, project plans and bug reports. This will help those analyzing the source code to have a clearer understanding of the intentions of the original programmer as well as any pitfalls or issues that arose during the creation of source code. Makefiles and header files are two additional file types that can be beneficial to produce. Makefiles are text files that are used to translate source code files into an executable object code file. The person who analyzes the source code should utilize the makefile to better comprehend how the finished program is created. Header files are certain pieces of source code that programmers can use to separate code into reusable files.

In *Metavante Corp. v. Emigrant Sav. Bank,* Emigrant motioned to compel Metavante to produce the source code. Emigrant claimed it needed the source code not only to defend itself against the breach of contract claim, but also to prosecute the fraud and breach of contract counterclaim. Emigrant believed the source code would reveal the quality of the product delivered.

The court granted Emigrant's motion to compel despite Metavante's concerns about the high cost of production. Emigrant offered to control the cost by using its outside consultants to review the code and parse relevant information. The court found this to be a very focused, attractive solution and agreed that the code should be produced.

Transferring Data to Opposing Counsel
When transferring data to opposing parties, the source code should be produced on encrypted key-based drives for added security. File-by-file encryption would be ineffective and tedious for both parties due to the magnitude of files. TrueCrypt and PGP (Pretty Good Privacy) are high-level encryption programs for hard drives. It is important for counsel to keep in mind that source code can take up several terabytes of storage which can take days to make a single copy of. So, if this is required, counsel should plan the time accordingly.

Security and Risk Mitigation
There are several best practices that counsel can follow to help ensure source code is only reviewed by the appropriate people and that no source code is leaked. Counsel can reduce the risk of having source code leaked by explicitly communicating concerns with the court so the protective order can contain provisions — such as privilege logs — and sanctions regarding leaks. In addition, counsel should ensure that any proprietary or sensitive information being turned over to opposing counsel is required by the court. It is important not to be overly inclusive. During the review process, computers used to access or review the source code should not be stored in a place where they can be accessed by unauthorized persons. These computers should also not have any Internet access. Refer to the section regarding "Court Standards for Source Code Production" for additional security and risk management ideas.

As discussed earlier in the Meet and Confer section, consider offering reasonable alternatives of full source code production to mitigate risk and cost.

The case that follows is an example of reasonable options that were presented and agreed upon by the court in order to address risk.

In the discovery dispute of *Generac Power Sys., Inc. v. Kohler Co.,* (No. 11-CV-1120-JPS, 2012 WL 2049945 E.D. Wis. June 6, 2012), Generac argued that Kohler hadn't produced particular source code for a program that was used in operating the allegedly infringing product. Kohler argued that the source code at issue was used in multiple products and offered an alternative. The court approved Kohler's offer to give Generac's counsel and expert a demonstration of the software at issue along with other access by using a software product that limited the exposure of the full source code and disclosure of manuals.

CONCLUSION

As the magnitude of source code increases, and the occasions where courts mandate source code production also increase, it is vital that attorneys understand the details of source code production before it's too late. While the words "source code production" oftentimes induce fear, being prepared can make all the difference. In order to save time and money as well as keep one's company's interests represented, make sure to have a thorough understanding of what source code is, know what source code one's company might be subject to reviewing and producing, and follow some of the best practices discussed in this paper. If counsel has a grasp on these key items, it is highly likely counsel will be ahead of the game when the court issues a motion to compel for source code review.

REFERENCES

[1] Phil Johnson, IT World, *Curiosity about lines of Code* (8/8/2012), Retrieved June 3, 2013 from, http://www.itworld.com/big-datahadoop/288893/lines-code-apollo-curiosity

[2] Issued for the invention of a new and useful process, machine, manufacture, or composition of matter, or a new and useful improvement.

[3] U.S. Patent and Trademark Office, *U.S. Patent Statistics Chart Calendar Years 1963-2012*, Retrieved June 3, 2013 from, http://www.uspto.gov/web/offices/ac/ido/oeip/taf/us_stat.htm

[4] PricewaterhouseCoopers LLP (2012). *2012 Patent Litigation Study*. Retrieved June 3, 2013 from, http://www.pwc.com/en_US/us/forensic-services/publications/assets/2012-patent-litigation-study.pdf

[5] Lydia Pallas Loren, Andy Johnson-Laird, The Federal Courts Law Review Volume 6, Issue 1, *Computer Software-Related Litigation: Discovery and the Overly-Protective Order*, Retrieved March 11, 2013 from, http://www.fclr.org/fclr/articles/html/2010/Loren.pdf

[6] Lee A. Hollaar, *Requesting and Examining Computer Source Code*, Retrieved June 12, 2013 from, http://digital-law-online.info/papers/lah/BNAsourcecode.pdf

[7] Lydia Pallas Loren, Andy Johnson-Laird, The Federal Courts Law Review Volume 6, Issue 1, *Computer Software-Related Litigation: Discovery and the Overly-Protective Order*, Retrieved March 11, 2013 from, http://www.fclr.org/fclr/articles/html/2010/Loren.pdf

[8] A version control system is used to manage multiple versions of computer files and programs. It allows users to lock files so they can only be edited by one person at a time, and track any changes to files.

[9] Lee A. Hollaar, *Requesting and Examining Computer Source Code*, Retrieved June 12, 2013 from, http://digital-law-online.info/papers/lah/BNAsourcecode.pdf