UIC John Marshall Journal of Information Technology & Privacy Law

Volume 23
Issue 4 *Journal of Computer & Information Law - Summer 2005*

Article 2

Summer 2005

The Legal Status of Software, 23 J. Marshall J. Computer & Info. L. 711 (2005)

Daniel B. Garrie

THE LEGAL STATUS OF SOFTWARE

DANIEL B. GARRIE†‡

I. INTRODUCTION: THE WORLD OF SOFTWARE AND THE LAW

The objective of this guide is not to give a judge a PhD in computer science, but rather, to ensure that judicial decisions reflect an understanding of software's multiple facets and its relation to the boundaries of the law. When hearing legal arguments involving software, the judiciary should make a concerted effort to examine the software program's production methodology and intended purpose prior to issuing legal decisions because the decisions should reflect an in depth understanding of the product at issue. A court involved in litigation fraught with software themes should understand the software itself to ensure the delivery of fair and equitable legal decisions.

In the legal arena, judges should review software from different perspectives based on the legal issue before the court. For instance, a court hearing an anti-trust dispute should not be concerned with whether the software vendor tested the software sufficiently prior to introducing it to market, but rather with the software's intended and actual market effects. In a product liability suit, however, such testing would be a significant issue of judicial inquiry for the court. These examples demonstrate that judges should possess a broad general understanding of software, and should then refine their knowledge based on specific legal issues that are brought before the court. This reference guide, therefore, is designed to provide a high-level overview of software and then examines software in greater depth for specific legal areas likely to spawn legal disputes directly involving software.

---

† Daniel B. Garrie has received his J.D. from Rutgers University School of Law, specializing in cyberlaw litigation. He received his M.A. and B.A. in computer science from Brandeis University in 2000 and 1999, respectively, with his coursework focused on artificial intelligence. Over the past eight years, Mr. Garrie has worked with the U.S. Department of Justice and other large corporations as an enterprise technical architect, focusing on Web-enabled enterprise systems. He has penned several law review articles in the past year on a variety of technology and legal issues.

‡ Many thanks to Matthew J. Armstrong and Mandy E. Knox. I would like to express my gratefulness for Daniel H. Mandel's efforts and input in penning this paper.

I hope that the explanations provided will enable judges and lawyers who deal with software to understand the terminology, context and legal doctrines governing software in legal disputes. The reference guide is organized as follows:

- Section II provides a brief overview of the current legal framework that is evolving with respect to software development. It describes the software development trend of coding to skirt the law and avoid liability, examining the recent Supreme Court ruling in *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*[1]
- Section III provides a broad overview of software. It presents multiple facets of software and offers a guide of the various components involved in software, including: software development methods, software design, and a high-level overview of programming.
- Section IV provides a series of tutorials on cutting edge technology that is going to appear before judges with greater frequency in the coming years, including: Voice Over Internet Protocol Telephony,[2] Internet Cookies,[3] Spyware,[4] and Clickstream Data.[5]

---

1. 125 S.Ct. 2764 (2005).

2. *See* Wikipedia, *Voice over IP*, http://en.wikipedia.org/wiki/VoIP (last modified Dec. 27, 2005) (defining "Voice over IP" (also known as VoIP, IP Telephony, and Internet telephony) as the routing of voice conversations over the Internet or any other IP network. The voice data flows over a general-purpose packet-switched network, instead of the traditional dedicated, circuit-switched voice transmission lines. Protocols used to carry voice signals over the IP network are commonly referred to as Voice over IP or VoIP protocols. Voice over IP traffic may be deployed on any IP network, including ones lacking an Internet connection, for instance on a private building-wide LAN. For an exposition on Internet voice communications such as VoIP and privacy issues, *see* Daniel B. Garrie, Matthew J. Armstrong, Donald P. Harris, *Voice Over Internet Protocol and the Wiretap Act: Is Your Conversation Protected?*, 29 Seattle Univ. L. Rev. 97 (2005).

3. Cookies register information about a visit to a Web site for future use by the server. The server that operates with the Cookie technology may receive information of cookies from additional sites, which creates concerns of breach of privacy. The Oxford English Dictionary defines cookie as follows: "Computing. A token or packet of data that is passed between computers or programs to allow access or to activate certain features; (in recent use spec.) a packet of data sent by an Internet server to a browser, which is returned by the browser each time it subsequently accesses the same server, thereby identifying the user or monitoring his or her access to the server." Oxford English Dictionary Online (2d ed. 1989) (accessed May 16, 2005). *See also* Michael Gowan, *How It Works: Cookies*, available at http://www.pcworld.com/hereshow/article/0,aid,15352,00.asp (Feb. 22, 2000).

4. *See* Wikipedia, *Spyware* ¶¶ 1-2, *at* http://en.wikipedia.org/wiki/Spyware (last visited Dec. 27, 2005) (defining the term "Spyware" as "a broad category of malicious software intended to intercept or take partial control of a computer's operation without the user's informed consent. While the term taken literally suggests software that surreptitiously monitors the user, it has come to refer more broadly to software that subverts the computer's operation for the benefit of a third party. Spyware differs from viruses and worms in that it does not usually self-replicate. Like many recent viruses, Spyware is designed to exploit infected computers for commercial gain. Typical tactics furthering this goal include delivery of unsolicited pop-up advertisements; theft of personal information (including financial information such as credit card numbers); monitoring of web-browsing activity for

- Section V presents a topical guide to various legal areas, analyzes the current legal framework and presents specific questions that a judge hearing arguments may consider. These different legal areas were singled out because of the complex legal nature and the unique aspects of software that are involved when the matter is before a court.

II. OVERVIEW: WHAT IS "SOFTWARE" IN RELATION TO THE LAW?

Today, courts face a new dilemma where software[6] is being developed to side-step laws at a rate exceeding a commensurate legislative response.[7] A prime example of where a firm grasp of software has played a notable role in a judicial decision, absent direction from the legislature was in *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*[8] There the Supreme Court held that distributors of multiple-use software can be liable for third parties' copyright infringing activities where the distributor

---

marketing purposes; or routing of http requests to advertising sites"). *See also* Alan F. Blakely et al., *Coddling Spies: Why the Law Doesn't Adequately Address Computer Spyware*, 2005 Duke L. & Tech. Rev. 25, 26-27 (defining Spyware, as such, and with respect to propounding a legal framework to protect potential victims and its legitimate users).

5. Once a user has accessed a Web site that uses cookie technology or an affiliated site, the embedded cookie on the hard drive begins collecting data about the user's Web activities. There are four reported cases where cookie technology was used by a Web site to mine personal information from the user's machine: *Chance v. Avenue A, Inc.*, 165 F. Supp. 2d 1153, 1155 (W.D. Wash. 2001); *In re Intuit Privacy Litigation*, 138 F. Supp. 2d 1272, 1274 (C.D. Cal. 2001); *In re DoubleClick, Inc. Privacy Litigation*, 154 F. Supp. 2d 497, 502-03 (S.D.N.Y. 2001); and recently *In re Pharmatrak, Inc.*, 329 F.3d 9, 12 (1st Cir. 2003).

6. *See generally* William A. Fenwick & Gordon K. Davidson, *Admissibility of Computerized Business Records*, 14 Am. Jur. Proofs of Facts 2d § 173 (1977) (explaining that computer "software" is generally defined as that material, separable from the "hardware", or physical equipment, which comprises the computer program's instructions); *see* John G. Martin, *The Revolt Against the Property Tax on Software: An Unnecessary Conflict Growing out of Unbundling*, 9 Suffolk U.L. Rev. 118 (1975) (explaining that "software" has also been more widely defined as all those aspects of the computer which are not hardware, and thus includes such known programming elements as educational material, manuals, training of personnel, and perhaps even maintenance of the hardware). For the purposes of this annotation, the category of computer software includes, among other items, computer programs, the media on which they were recorded, and the services, which may be rendered to the computer purchaser by the manufacturer after purchase of the machine; only the computer machinery itself is considered hardware.

7. *See Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*, 125 S.Ct. 2764, 2780 (2005) (holding that "[i]t is undisputed that StreamCast beamed onto the computer screens of users of Napster-compatible programs ads urging the adoption of its OpenNap program, which was designed, as its name implied, to invite the custom of patrons of Napster, then under attack in the courts for facilitating massive infringement").

8. *Id.* (finding that although the black letter of the law had been followed in the context of determining liability, the Court explicitly held that the intent behind the specific actions directed the outcome against Grokster).

actively seeks to advance the infringement.[9] In its examination of that software, the Court further held that the judiciary must be mindful of technology developers who[10] successfully code around the law in bad faith.[11] *Grokster* compels lower courts to examine whether the design of a software application seeks to advance the infringement.[12] Although *Grokster* applies only to copyright infringement suits, the Supreme Court's focus on the inner-workings of the file sharing technology demonstrates the growing need for judges to possess a framework for understanding software.[13]

III. WHAT IS SOFTWARE?

From a technical perspective, the term software is defined as a program[14] and all of the associated information and materials needed to support its installation, operation, repair and enhancement. The view that software consists solely of code[15] is erroneous. Software is repre-

9. *Id.* at 2787 (Breyer, J., concurring) (explaining, "I agree with the Court that the distributor of a dual-use technology may be liable for the infringing activities of third parties where he or she actively seeks to advance the infringement").

10. *See, e.g. United States v. Microsoft Corp.*, 980 F. Supp. 537, 545 (D.D.C. 1997); *cf.* Richard A. Posner, *Antitrust in the New Economy*, 68 Antitrust L.J. 925, 937 (proposing that computer science and related technologies present formidable challenges for lay judges and jurors to absorb when being asked to reach a decision).

11. *Grokster*, 125 S.Ct. at 2781 (explaining that "[o]f course, in the absence of other evidence of intent, a court would be unable to find contributory infringement liability merely based on a failure to take affirmative steps to prevent infringement if the device itself was otherwise capable of substantial non-infringing uses).

12. *Id.*

13. *Id.* at 2785-2786. For example, Spyware developers act in a fashion analogous to the Grokster developers who attempted to avoid copyright law by designing a decentralized file distribution system allowing end-users to share files directly with each other to make file copyright enforcement cost-prohibitive. The *Grokster* Court is arguably reversing a long standing legal trend by moving away from a literal application of common law authority and statutes, especially those dealing with technological matters, when the "full transaction" clearly violates a legal principle or the drafters' intent. *Id.* at 2785-86. A different interpretation of *Grokster* is that the Court simply held that developer must not track or profit from file sharing technology. Arguably, the low cost of distribution of P2P software and a single coder not seeking profit, rather to simply share files, could "code" around the Court's holding in *Grokster*. Although the Court indicates that the intent of the software developer and its affiliated firm is important, it is unclear what would happen if such software were developed under the umbrella of good will and not monetary profit. Once this occurs, the Court will have to contend with the issue of software developers who "code" around the black letter law leaving the issue clouded. *Grokster* does indicate that courts must begin to examine in varying degrees the inner workings of software when applying the law.

14. *See* Wikipedia, *supra* n. 4.

15. Princeton University, *WordNet - 2.1*, http://wordnet.princeton.edu/perl/webwn?o2=&o0=1&o6=&o1=1&o5=&o4=&o3=&s=code (accessed Nov. 24, 2005) (explaining that "code" (computer science) is "the symbolic arrangement of data or instructions in a computer pro-

sented as an intelligible, usable, and extendible set of instructions that can be run on a computer. In the context of this paper, the term software refers to products that run on machines capable of translating and executing code.[16] Software basically instructs a computer how to perform actions, so in the broadest sense, software includes everything that is not hardware.[17] Computers are, in effect, incomplete machines when manufactured and acquire functionality only after being coupled with software.[18]

A. WHAT IS SOFTWARE DEVELOPMENT?

Software engineering ("SE") or software development is defined as the profession concerned with specifying, designing, developing and maintaining software applications by applying technologies and practices from computer science, project management and other fields.[19] The complexity and rapid growth of software development in the last decade began with a handful of developers designing individually fulfilling projects.[20] Today, software engineering has become a major defining mainstream industry.[21] Although computers have become a household

---

gram or the set of such instructions . . . [and] encoded [means to] convert ordinary language into code." Stating "[w]e should encode the message for security reasons."); *Evolution, Inc. v. SunTrust Bank*, 342 F. Supp. 2d 943, 947 n. 3 (D. Kan. 2004) (explaining software code may be source code or object code. Defining 'source code' as "human-readable software code used by programmers to develop a software program." Defining 'Object code' as a "machine-readable code that is a series of 1s and 0s. Source code is converted into object code through a process called 'compiling.'") For a technically accurate discussion of source and object code in the legal literature, *see, e.g.* Pamela Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 Duke L.J. 663, 764-69 (1984).

16. *See e.g. Massachusetts v. Microsoft Corp.*, 362 U.S. App. D.C. 152, 182 (D.C. Cir. 2004); *see also e.g. Law Research Service, Inc. v. General Automation, Inc.*, 494 F.2d 202, 204 n.3 (2d Cir. 1974); *see also e.g. Sherin v. Tennessee*, 601 F.2d 590 (6th Cir. 1979) (unpublished).

17. Pamela J. Garland, *Value & Cents: The Valuation of Computer Software*, 18(1) A.B.I.J. 24, (Robert F. Reilly ed., Feb. 1999).

18. *See* Fenwick *supra* note 6.

19. *See* Wikipedia, Software Engineering, ¶¶ 1, 1.2, *at* http://en.wikipedia.org/wiki/Software_engineering (last visited Dec. 27, 2005). The term software engineer is thought to have first been coined at the NATO Science Committee in Garmish, West Germany in October of 1968. *Scaling Up: A Research Agenda for Software Engineering*, 33 COMM. OF THE ACM 281 (Mar. 1990), available in WL, COMP-ASAP Database, at *24 & n.7 (excerpts from a report on software development by the Computer Science and Technology Board). Software engineering has been defined as: "The practical application of scientific knowledge in the design and construction of computer programs and the associated documentation required to develop, operate, and maintain them." B.W. Boehm, *Software Engineering*, in Classics in Software Engineering 325, 326 (Edward N. Yourdon ed. 1979).

20. *Id.* at ¶ 2.2.

21. *Id.*

item, the basics are still fraught with complexity, irregularity and difficulty for a majority of users.[22] Therefore, we should initially separate the software designer from the computer programmer in order to differentiate their abilities and job functions. Software designers are the architects of the software whereas the computer programmers are, analogously, the construction engineers. Software developers do not construct software; they design the basis of the software. The architectural design of a software program corresponds to the design sketches and layout schedules used in certain engineering disciplines. This design is then translated by the programmer.[23]

The boundary between design and construction is always clearly marked by an artifact: the blueprint.[24] Design encompasses all the activities needed to create the blueprint; construction encompasses all the activities needed to create products from the blueprint. In a perfect world, a blueprint would specify every detail of the product to be created; of course this is rarely, if ever, the case. Still, the purpose of a blueprint is to describe the product to be constructed as precisely as possible. Does the architectural design of a software system describe the software product "as precisely as possible?" Not usually. The architectural design usually describes a software product's essential elements, but certainly not all of its details. Thus, architectural design is clearly not a blueprint. Only high-level language code describes all the details of a software system, thereby qualifying as the software's blueprint. Moreover, since all activities leading up to the creation of the blueprint are design, all high-level software development is design.[25]

Software itself can be categorized in many different ways, but one important distinction is the difference between "operating systems" and other software. An "operating system"[26] ("OS") is software that provides an application platform and a user interface for end-user computers that often manage the computer's hardware.[27] The OS is the functional

22. *See The Computer Museum History Center*, http://www.computerhistory.org (accessed Dec. 23, 2005) (providing a detailed chronology of the history of computing); *see also Charles Babbage Institute*, http://www.cbi.edu (accessed Dec. 23, 2005) (providing resources on the history of computing).

23. *See* Wikipedia, Programming, *at* http://en.wikipedia.org/wiki/Programming (last visited Jan. 11, 2006) (elucidating the role of source code in programming paradigms).

24. *See e.g. Norwest Corp. v. Commissioner*, 110 T.C. 454, 456 (T.C., 1998).

25. *Id.*

26. The OS software manages the CPU (Central Processing Unit) and related hardware components including: keyboard, monitor, storage media and communication devices. The OS provides the hardware management and user interface functions into a single product commonly termed OS.

27. Prominent examples include the Apple Mac OS, Linux, and Microsoft Windows. For an illustrative list typifying application programs as such, *see* Definition of "Application Program" (accessed Dec. 23, 2005), http://searchserviceprovider.techtarget.com/

equivalent of a "software platform," since other software is constructed to operate within the parameters of the OS.[28] The platform itself contains Application Program Interfaces ("APIs") that instruct software vendors how to access useful modules of code built into the platform. The APIs and the OS's underlying code empower software developers to create software in a more expeditious fashion. Microsoft Windows,[29] for example, contains a multitude of APIs that can be accessed by software operating on the OS, such as word processing, spreadsheets, and games.[30]

B. WHAT IS SOFTWARE DESIGN?

The actual technical design process is almost universally recognized by the software engineering community as the basic building block for the future of well-engineered, reliable software products.[31] Software design begins with the designer conceptualizing a problem and developing a solution based on applied techniques and logically applied principles. Design principles have changed drastically throughout the years as the understanding of user needs and interface design has become a market driven phenomenon.[32] Due to previously recognized software inadequacies and poorly thought out software,[33] the design community has initiated a solution-based design evolution in which the designer and programmer have developed an almost symbiotic team-based relationship.[34] More recent methods encompass object-oriented design and computer aided software engineering tools that streamline how software is

---

sDefinition/0,,sid28<uscore>gci507192,00.html ("An application program is any program designed to perform a specific function directly for the user or, in some cases, for another application program.")

28. *See* Platform, FOLDOC (Free On-Line Dictionary of Computing), *at* http://foldoc.doc.ic.ac.uk/foldoc/index.html (last visited Jan. 11, 2006) ("Specific computer hardware, as in the phrase "platform-independent' may also refer to a specific combination of hardware and operating system and/or compiler, as in "this program has been ported to several platforms.' It is also used to refer to support software for a particular activity, as in "This program provides a platform for research into routing protocols.'")

29. Microsoft Windows is an Operating System. *Id.*

30. An OS product can serve as a software platform; however, another layer can be constructed on top of the OS.

31. Frederick P. Brooks, Jr., *No Silver Bullet: Essence and Accidents of Software Engineering*, IEEE Computer 10, 12 (Apr. 1987) (describing the complexity of conforming a program to other systems' interfaces). *See also* Mitchell Kapor, *A Software Design Manifesto*, 16 Dr. Dobbs J. 62 (1991), *available at* http://www.ddj.com/documents/s=1066/ddj9101c/9101c.htm (accessed Jan. 11, 2006).

32. Stephen R. Schach, Software Engineering 30-33 (2d ed. 1993) (describing the industry shift from the generation of computer code to the generation of software designs). *See generally* George T. Heineman & William T. Councill, Component-Based Software Engineering (Addison-Wesley 2001).

33. *See* Software Engineering, *supra* n. 19, at ¶ 2.1 (reporting on quality issues with software).

34. *See* Brooks, *supra* n. 31.

designed and recognize that software design is primarily and intuitively people driven.[35] The final result of this design effort – the high level language code – is the blueprint of the software.[36] It is the compiler/ linker who mechanically constructs the software product – a binary or executable program that will run on a computer from that high level code.[37]

Drawing a bright line between executable source code and non-executable software designs is not an easy task.[38] A software system's architectural design corresponds to the cardboard models or design sketches used in some engineering disciplines. It is particularly difficult to identify the point at which the process of "designing" software ends and the process of "implementing" or "coding" software begins.[39] Despite these difficulties, executability provides a reasonable basis for distinguishing between mere software designs and executable source code.[40]

1. *Coding, testing, and software*[41]

Computer code or coding is the process of writing detailed in-depth instructions using a variety of different coding languages that essentially

35. *See generally* Wikipedia, Object-Oriented Programming, *at* http://en.wikipedia.org/wiki/Object-oriented_programming (last visited Jan. 11, 2006). For a guide on a software engineer's task of evolving software interfaces to improve userability as a series of translations, from code to a natural language (such as English), *see* Bruce D. Abramson & Dmitri L. Mehlhorn, *The Fettered Liberty to Integrate: Legal Implications of Software Engineering*, 10 B.U. J. Sci. & Tech. L. 209, 212 (2004).

36. *Id.* Abramson ("Software engineers. . .program in these high-level languages to move the upward translation chain all the way to the user interface.")

37. *See e.g.* Samuelson, *supra* n. 15, at 672-89 ("Computers, however, cannot run source code; the code must first be converted to a machine-executable form called 'object code.' This is done with the use of programs, known as compilers or assemblers").

38. *See Robotic Vision Sys.*, 249 F.3d 1307 (Fed.Cir.2001)

39. This issue is longstanding in both the subject field of inquiry and legal scholarship, *see, e.g.*, Joseph G. Arsenault, *Comment: Software Without Source Code: Can Software Produced by a Computer Aided Software Engineering Tool Be Protected?*, 5 Alb. L.J. Sci. & Tech. 131, 164 (1994) ("Object-oriented representations, rapid prototype tools, and fourth-generation languages blur the line between design and code, and provide other high-level representations of software.")

40. *See e.g.*, Robert Plotkin, *Computer Programming and the Automation of Invention: A Case for Software Patent Reform*, 2003 UCLA J. L. Tech. 7, n. 29 (2003) (citing Dick Hamlet & Joe Maybee, The Engineering of Software: Technicla Foundations for the Individual 211-12 (Addison-Wesley 2001) stating, "Despite the similarities between the processes of designing and coding, there is one tremendous distinction between the end results: code can be executed on hardware—a design cannot.") *See also Id.* at 165.

41. For a comprehensive overview of "software testing," *see* Wikipedia, Software Testing, *at* http://en.wikipedia.org/wiki/Software_testing (last visited Jan 13, 2006) (testing is a process used to help identify the correctness, completeness and quality of developed computer software. With that in mind, testing can never completely establish the correctness of arbitrary computer software. In computability theory, a field of computer science, an elegant mathematical proof concludes that it is impossible to solve the halting

speak to the computer and tell it what to do. This code can accomplish single or multiple tasks and can often be re-used and edited to further future software changes or upgrades.[42] The programming language or computer language that is used to write the code is a standardized communication technique for expressing instructions to a computer. It is a set of syntactic and semantic rules used to define computer programs. The language enables a programmer to precisely specify what data a computer will act upon, how these data will be stored and transmitted, and precisely what actions to take under various circumstances.[43]

Software design is converted into a functional product capable of actual implementation through a complex set of instructions called source code. Software code represents the letters on the page that actually form the words that are used to implement the software. The language that is used varies; however, the code essentially implements the software's design.[44] Most software is sold or leased to end-users in object code form.[45] While software users can easily observe the outward functions of a program, they cannot as easily perceive a program's internal ideas, processes, structures and actual methods of operation.

In the typical software development process, programmers write code in a programming language using alphanumeric characters that can be understood by a person familiar with the language. This form of the program is referred to as "source code." After the source code is written, a "compiler" program translates the source code into a machine-readable format.[46] In order to understand the ideas and "inner workings" of a computer program, one must obtain either the original source code or detailed written specifications from the program's developer.

Software testing is defined as a process used to identify the completeness and quality of developed computer software. Actually, testing can never establish the correctness of computer software. Current testing methodologies can only find defects, not prove that there are none. Testing approaches that are used to do this range from the most informal ad hoc testing, to formally specified and controlled methods such as *automated testing and unit testing*.[47]

---

problem, the question of whether an arbitrary computer program will enter an infinite loop, or halt and produce output. In other words, testing is nothing but criticism or comparison, that is comparing the actual value with expected one).

42. *See* Julie Cohen & Mark Lemley, *Patent Scope and Innovation in the Software Industry*, 89 Cal. L. Rev. 1, 41 (2001) (depicting software innovation's incremental character as akin to most inventions patented).

43. *See generally supra* n. 17.

44. *See e.g.*, *supra* nn. 17, 25.

45. *See* Wikipedia, *supra* n. 35.

46. *See e.g.*, Samuelson, *supra* n. 15, at 672-89 (describing the function of compiler programs).

47. *Supra* n. 44.

The quality of the application can and normally does vary widely from system to system, but some of the common quality attributes include reliability, stability, portability, maintainability, and usability. For a more complete listing of attributes, it is suggested that the ISO standard ISO 9126 be consulted.[48]

C. WHAT ARE THE MAJOR METHODS FOR SOFTWARE DEVELOPMENT?

The development methods[49] used to create software shape the architecture and coding languages used to develop it, thereby influencing the actual fruition of the software itself. Although software development methods have been subdivided into nineteen developmental sub-types to ensure that discussion remains both manageable and focused, this section examines software development methods at a conceptual level, focusing on the general principals of implementation.[50] Software development is split into various process models that dictate a software project's life cycle to ensure that the software product meets the end-users' needs.[51] Most system development process models fall under the following three different approaches: (1) Ad-hoc Development; (2) the Waterfall Model;[52] and (3) the Iterative process.[53]

1. *Ad-hoc Development*

Systems development can occur in a chaotic manner depending on the skills and experience of the programmers and staff involved in the development process. The Ad-hoc technique is commonly used, either entirely or for a sub-set of development (*e.g.*, small projects). The process capabilities of the Ad-hoc Process are unpredictable because software development constantly evolves throughout the entire process.[54] Schedules, budgets, functionality, and product quality are generally variable

48. *See* Wikipedia, *ISI 9126, at* http://wikipedia.org/wiki/ISO_9126 (last modified Oct. 29, 2005) (ISO 9126 is an international standard for the evaluation of software. It classifies the areas in a structured manner as follows: *Functionality* is a set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs; *suitability, accuracy, interoperability, compliance, security*, and *reliability* are the set of attributes that bear on the capability of software to maintain its level of performance under stated).

49. Here, the focus is on the methods utilized by programmers to develop and evaluate their software.

50. *Pfaff v. Wells Electronics, Inc.*, 525 U.S. 55, 67-68 (1998).

51. *See* Wikipedia, Software Development, *at* http://en.wikipedia.org/wiki/Software_development_methodology (accessed Jan. 13, 2006).

52. *See* Wikipedia, Waterfall Model, *at* http://en.wikipedia.org/wiki/Waterfall_model (accessed Jan. 13, 2006).

53. Adjoining these three main models is the burgeoning extreme programming methodology.

54. *See* http://www.sei.cmu.edu/cmm/ (accessed July 5, 2005).

and inconsistent. The overall performance of the projects in these settings hinges on the capabilities of individuals and widely varies with their respective abilities, knowledge, and motivations. Some individual software projects, however, produce excellent results when utilizing this method.[55] Usually, a product's success hinges on the heroic efforts of the team delivering the product, and not the fact that they utilized the Ad-hoc method. The Ad-hoc model's reliance upon the variable capabilities of the parties involved in delivering the product render it difficult to use for the delivery of large projects.[56]

2. *Waterfall Model*

The Waterfall Model has been around in structured system development for quite some time.[57] It has been criticized recently as being too rigid and unrealistic for developing products that meet a customer's expectations. The Waterfall Model itself, however, is still widely used and is the core of most *process-based models.* The Waterfall Model is composed of the following steps:[58]

a. System Conceptualization

System Conceptualization refers to the consideration of all aspects of a targeted business function or process. The goal of system conceptualization is to ascertain how each specific aspect of a system interrelates, and to determine which aspects should be included in the system to be constructed.

b. Systems Analysis

Systems analysis refers to the actual gathering of system requirements, with the goal of ascertaining how these requirements will be included in the system. This process hinges on extensive communication between the customer and developer. Because of this dependency, third parties are often retained to facilitate communication between the customers and the system developer to ensure that the product satisfies all of the customer's needs.

c. System Design

Once system requirements have been gathered and analyzed, the developers must make a detailed blueprint of the system being constructed to ensure that it performs all necessary tasks in an efficient manner.

55. *See* Mark C. Paulk et al., *Key Practices of the Capability Maturity Model, Version 1.1*, Software Engineering Institute (Feb. 1993).

56. *Id.* at 19.

57. *Id.*

58. *Id.* at ¶ 2 (particularizing the usage of the Waterfall Model).

Specifically, the System Design phase focuses on the system's data requirements,[59] interface construction,[60] and software development.[61]

d. Coding

Coding is the process involved in the development of the software itself. In this step, the requirements and systems specifications from the System Design phase are translated into code, resulting in an executable machine code capable of operating on an electronic machine.[62]

e. Testing

Testing is performed throughout a product's developmental cycle to ensure that it is works correctly and efficiently. Testing focuses on two distinct areas: internal and external. Programmers perform internal testing to ensure that the code is efficient and consistent throughout the program. External testing is used to ensure that the software meets the system design requirements and all necessary details associated with those processes. Testing tends to be a labor-intensive process because it is iterative by nature when done in the waterfall setting.[63]

The Waterfall Model's main advantage is that it facilitates departmentalization and managerial control. Software development teams using the Waterfall Model can delegate specific tasks to different departments and can establish detailed schedules of deadlines for each stage of development. This production process enables software develop-

59. *See* Wikipedia, Systems Design, *at* http://en.wikipedia.org/wiki/system_design (last modified Nov. 9, 2005). Systems design is the process or art of defining the hardware and software architecture, components, modules, interfaces, and data for a computer system to satisfy specified requirements. One could see it as the application of systems theory to computing. Some overlap with the discipline of systems analysis appears inevitable.

60. John Wiley & Sons, *Glossary*, *available at* http://www.wiley.com/college/busin/icmis/oakman/outline/chap08/misc/glossary.htm (accessed Nov. 9, 2005) (A term that describes the way a database user communicates with the software, such as querying by example or using a natural language search strategy).

61. *See generally supra* n. 21 Software engineering ("SE") is the profession concerned with specifying, designing, developing and maintaining software applications by applying technologies and practices from computer science, project management, and other fields.

62. *See* Wikipedia, Coding, *at* http://en.wikipedia.org/wiki/Coding (accessed Nov. 16, 2005). The term coding has the following meanings: in communications systems, the altering of the characteristics of a signal to make the signal more suitable for an intended application, such as optimizing the signal for transmission, improving transmission quality and fidelity, modifying the signal spectrum, increasing the information content, providing error detection and/or correction, and providing data security (Note: A single coding scheme usually does not provide more than one or two specific).

63. *See* Wikipedia, Iteration, *at* http://en.wikipedia.org/wiki/Iterative (accessed Nov. 12, 2005). Iteration is the repetition of a process, typically within a computer program. Confusingly, it can be used both as a general term, synonymous with repetition, and to describe a specific form of repetition with a mutable state.

ment teams to build products in a series of well-defined steps, much like an automobile is built on a production line, ensuring that the software is developed on time. The various phases mentioned above proceed in order, without any overlapping or iterative steps.[64]

Although the Waterfall Model has been used for a long time, it is not without its critics. Critics first point out that real projects never actually follow the step-wise model utilized by the Waterfall Model. Second, at the start of most projects there is ambiguity about specific requirements and objectives, making it logically difficult for customers to identify specific system criteria requirements. Third, the Waterfall Model can be a laborious and time demanding process that does not create a working version of the system until late in the process. Finally, once an application is in the testing stage of the Waterfall Model, it is very difficult to go back and change an ill-conceived process that was created in the concept stage.[65]

1. *Iterative Model*[66]

The Iterative Model divides a software production project into small parts, enabling coders to demonstrate results earlier in the development process and to obtain invaluable end-user feedback regarding the system's functionality. It is usually a sub-set of the Waterfall Model with feedback from a particular phase providing vital data for the next design phase.[67] The basic idea behind Iterative development is to deliver a software system incrementally, allowing developers to capitalize on information garnered from incremental deliverable versions of the system. Learning in the context of the Iterative Model, however, comes from both system development and use, where data is gathered from both coders and end-user. At each Iteration, programmers make design modifications in addition to adding new functional capabilities.

IV. REFERENCES ON CUTTING EDGE TECHNOLOGY

Below is a review of cutting edge technologies that come before the court. This section provides a series of tutorials on cutting edge technology that will likely appear before judges with increasing frequency in the

64. *See* section Iterative Model for analysis discussion, *infra*.

65. *See supra* n. 56, at ¶ 5.

66. Other models include the following: (1) feature driven development; (2) Rational Unified Process. *See* Wikipedia, Rational Unified Process, *at* http://en.wikipedia.org/wiki/Rational_Unified_Process (last visited Jan. 13, 2006); *See also supra* note 56, at ¶ 6; *See* Cohen, *supra* n. 42, at 41 nn.171-73.

67. Several variations of this model exist and allow the team to produce a component of the software product early on in the process so they can get feedback from the team or they incrementally release the product, allowing for smaller but more frequent releases of the product. *See supra* n. 56, at ¶ 1.

coming years, including: Spyware, Voice Over Internet Protocol Telephony, Internet Cookies, and Clickstream Data.

A. WHAT IS SPYWARE?

Spyware for Windows is flourishing. Some statistics indicate that 85 percent of computers in the United States host at least one Spyware application.[68] Spyware is generally defined as software that once installed on a person's computer (usually without consent) collects and reports in-depth information about that end-user. The information gathered may encompass web-surfing habits, each and every keystroke, e-mail messages, credit-card information, and, in fact, any personal information on a user's computer. In the world of technology, "Spyware" is the umbrella under which numerous other malicious software technology falls, including: adware,[69] trojans, hijackers,[70] key loggers,[71] malware,[72] and dialers. While each of these technologies has its own unique behavior, for the most part they are all installed without a user's informed and explicit consent, and tend to do things the user might not want to have done.

Spyware itself is a sub-set of the different sorts of software that is designed to monitor end-user usage or obtain end-user information.[73] Spyware achieves this by transmitting information about the end-user in a variety of ways;[74] however, the most common mechanism used relies upon the transmission of data input by end-users over an Internet connection.[75] Utilizing this technology, Spyware operates in relative secrecy, gathering end-user information without the end-user's consent or

68. Richard H. Stern, *FTC Cracks Down on Spyware and PC hijacking, but not True Lies*, Micro Law 101 (IEEE Computer Society, Jan. – Feb. 2005).

69. James R. Hagerty and Dennis K. Berman, *Caught in the Net: New Battleground Over Web Privacy: Ads That Snoop*, Wall Street Journal A1 (Aug. 27, 2003). Spyware differs from adware technology because the primary purpose of Adware is to display advertisements on web pages or in programs such that those advertisements to generate income for the software owners.

70. *See generally*, J. Wilson. *How TopText Works*, *at* http://scumware.com/wm2.html (accessed Dec. 27, 2005).

71. *See generally* Schultz, E., *Pandora's Box: Spyware, Adware, Autoexecution, and NGSCB*, 22(5) Computers & Security 366 (2003).

72. *See generally* Pete Carfarchio, *The Challenge of Non-Viral Malware*, *available at* http://www.pestpatrol.com/Whitepapers/NonViralMalware0902.asp (Aug. 2, 2002).

73. Spyware is defined as: (a) any of various objects or pieces of equipment used for espionage; (b) Computing software that enables information to be gathered covertly about a person's computer activities, passwords, etc., and relayed to interested parties. Oxford English Dictionary (2d ed. 1989).

74. *See generally* David Moore et al., *Inside the Slammer Worm*, IEEE Security & Privacy 33-39 (July 2003).

75. *See e.g.*, Stephen H. Wildstrom, *How to Stymie the Snoop in Your PC*, BusinessWeek 28 (Apr. 5, 2004). In fact, the Internet method of transmission is preferred because it

knowledge.[76] Spyware blurs the existing fuzzy line between a malicious virus and an aggressive Internet marketing tool even further.[77] Spyware, however, can monitor more than just the Web pages an Internet surfer visits;[78] it is able to access the end-user's electronic file system,[79] e-mail system, Web pages viewed, and any other information the end-user accesses on the machine that is not encrypted.[80] The primary purpose of Spyware is to spy and to gather information that is transmitted to a third-party by invading a user's protected digital space,[81] unbeknownst to the end-user.[82]

1. *Spyware's Origins Lie Within Associated Cookie Technology*

The origins of Spyware lie in "associated cookies,"[83] greatly increasing a user's exposure and risk.[84] Associated cookies identify a single user each time that user connects to a member site on the World Wide

---

is done without the user ever being aware that data is transmitted because today people leave their computer connected to the Internet. *Id.*

76. When Spyware installs itself it is very difficult to remove because it embeds itself hidden within the system and it utilizes complicated techniques to detect and replace component parts. If an end-user rips out one or two parts, the undetected parts will come in and replace the files that were removed. The end-result is that even if the end-user is aware that Spyware is installed on their machine, it still is difficult for the end-user to remove, even if they are using Spyware removal tools.

77. Unfortunately, many end-users tend to be extremely short sighted and most of the time unable to discern the difference between what is and what is not Spyware. Furthermore, Spyware tends to be financially motivated, which is not true of past viruses/malware. *See* Jason Krause, *Prying Eyes*, 91(5) A.B.A.J. 60, Vol. 91 Issue 5, p60, 2p, 1c (May 2005). Today, a Spyware application might pop up a dialog box that warns you of a problem with your account only to redirect you to a look-alike site, which then relieves you of your financial resources. *Id.*

78. *See e.g.*, R.R. Urbach & G.A. Kibel, *Adware/Spyware: An Update Regarding Pending Litigation and Legislation*, 16 Intell. Prop. & Tech. L.J. 7, 12-16 (2004).

79. *See e.g.*, Elizabeth Prostic, Remarks, *Monitoring Software on Your PC: Spyware, Adware, and Other Software* (Spyware Workshop, Apr. 19, 2004), http://www.ftc.gov/bcp/workshops/Spyware/index.htm (last visited July 20, 2004); KaZaA Hack 2.5, http://www.kazaahack.net/home.html (last visited November 21, 2005); Radcliff, D., *Spyware*, 4(21) Network World 51 (2004).

80. C.J. Volkmer, *Will Adware and Spyware Prompt Congressional Action?(Or Does My Computer's CD Tray Open for No Apparent Reason?)*, 11(7) J. Internet L. 1 (2004).

81. *See generally*, S. Gibson, *OptOut*, http://www.grc.com/oo/news.htm (last visited May 10, 2005).

82. *See e.g.* E Foster, *The Gripe Line:The Spy Who Loves You-Some 'Free' Internet Services Come With the Kind of Surveillance You May Note Want*, 24 Infoworld 60, 60 (May 20, 2002).

83. I. Fayenson, *Cookies Challenge Meaning of Privacy*, 226 N.Y. L.J., No. 93, s10 (Nov. 13, 2001).

84. *See generally* Schultz *supra* n. 71.

Web.[85] Associated cookies are tools that track user activity and store data gathered from a specific user's interaction with a particular Web site.[86] Advertising companies such as DoubleClick typically form agreements with member sites allowing an advertiser that is DoubleClick's customer to place references to Spyware-like servers on participating corporate Web sites.[87] These references can be as simple as image files the size of a single pixel, commonly referred to as Web Bugs[88] that cause the end-user's browser to travel to a referenced Spyware site to acquire a referenced image file.[89] Once the end-user's browser communicates with that site, the Spyware site searches the particular user's system for a cookie.[90] If a cookie is found, the Spyware site uploads data recorded by the cookie since the user's last visit to a member site.[91] If a cookie is not found, the Spyware site transmits a globally unique identifier ("GUID") that identifies the user any time he or she visits a member site.[92]

The legal problem with this scenario is that the users do not see, access, or control the data that Spyware harvests from their machines;

---

85. *See generally* Daniel B. Garrie, Matthew J. Armstrong, Donald P. Harris, *Voice Over Internet Protocol and the Wiretap Act: Is Your Conversation Protected?*, 29 Seattle U. L. Rev. 97 (2005).

86. *Id.*

87. *See In re DoubleClick Inc. Privacy Litigation*, 154 F. Supp. 2d 497, 500; M.J. Berry, G.S. Linoff, Mastering Data Mining: The Art and Science of Customer Relationship Management (Wiley Computer Publishing 2000).

88. A Web Bug is a graphic on a Web page or in an e-mail message designed to monitor who is reading the particular communication. Web bugs are invisible to the naked eye and are usually positioned on Web pages by third party vendors interested in collecting data about visitors to those particular pages. The Privacy Foundation stated that Web Bugs are capable of tracking the following: (1) Number of times a page is viewed; (2) user demographics (*e.g.*, age, gender, etc.) for a particular Web Site; (3) personally identifiable information (user name, home address, phone number, email, date of birth, etc.) about users that visit a Web site – this data is usually utilized in conjunction with offline data to construct a complete user profile; (4) individual web page visitation pattern; (5) user search queries; (6) user's technology set (*e.g.*, Internet Browser, Screen Resolution, Plug-ins, etc.); and (6) allow a third party vendor the ability to execute server logging that the Web site can not itself perform. *See* D.M. Martin et al., *The Privacy Practices of Web Browser Extensions*, http://www.cs.uml.edu/~dm/pubs/bea (Dec. 2000); U.S. Fed. Trade Commn., *Privacy Online: Fair Information Practices in the Electronic Marketplace*, *available at* http://www.ftc.gov/reports/privacy2000/privacy2000.pdf (last modified May 2000).

89. *See generally* M.Pastore, *Inside Spyware: A Guide to Finding, Removing and Preventing Online Pests*, *available at* http://www.intranetjournal.com/Spyware (last visited Nov. 22, 2005).

90. W. Ames, *Understanding Spyware:Risk and Response*, IT Professionals IEEE Journal, 6(5), 25-29 (2004).

91. *See e.g.*, E. Doyle, *Not All Spyware is as Harmless as Cookies: Block it or Your Business Could Pay Dearly*, Computer Weekly 2003, at 32.

92. *See generally* Prostic, *supra* n. 79.

they are typically oblivious to the entire process.[93] Associated cookies' technical capabilities are limited since they are unable to query an end-user's system or install new applications.[94] Associated cookies can, however, record and share end-users' activities, browsing habits, and keystrokes, which are then transmitted to a central server perhaps for advertising purposes.[95] Associated cookies present a serious concern for individuals, governments, and corporations alike because they can hijack passwords,[96] financial account numbers,[97] and other personally identifiable information Internet users input into Web pages.[98]

Spyware today is usually application based,[99] which enables Spyware companies to assume control of end-users' systems, causing at least serial annoyance and at most severe security exposure and risk.[100] The security risk is particularly great because most end-users are usually unaware of the information breach, while those who are aware are unable to restrict application-based Spyware technology from functioning.[101] Spyware may operate like syphilis in the brain of the computer – slowly eating away computer function while being difficult or impossible to treat, perhaps even going undetected until too late. Once installed, Spyware can open direct receiving and transmitting channels to Spyware servers, advertisers, or to particular member sites.[102] These transmissions occur without the end-user's knowledge or explicit consent, yet the Spyware installer claims the end-user gave implicit consent based on initial consent granted by the end-user when installing the program or visiting the Web site.[103]

---

93. *See generally* Bruening, P. J. & M. Steffen, *Spyware: Technologies, Issues, and Policy Proposals*, 9(7) J. Internet L. 3 (2004).

94. Mark Gibbs, *The Rise and Rise of Scumware*, http://www.itworldcanada.com/Pages/Docbase/ViewArticle.aspx?ID=idgml-74e7e818-220f-4e0c-8201-5af2409c90b9 (accessed May 28, 2004).

95. *See* Doyle, *supra* n. 91.

96. Keith Furman, *Microsoft Presents Antispyware Strategy*, http://www.winnetmag.com/article/articleid/42432/42432.html (accessed May 18, 2004).

97. *See* R. Farrow, *Is your Desktop being Wiretapped,* http://www.itarchitect.com/shared/article/showArticle.jhtml?articleID=13000055&classroom (accessed Aug. 5, 2003).

98. *See generally* Prostic, *supra* n. 79.

99. Application based Spyware can be such that it has complete and total access to the user's system, beginning the moment that the system boots up. *See* Radcliff, *supra* n. 79.

100. *See* Avi Z. Naider, *Distinguishing Software-Based Contextual Marketing Technology from Spyware,* U.S. Senate Committee on Commerce, Science and Transportation (2004), *available at* http://commerce.senate.gov/ pdf/naider032304.pdf.

101. *See generally*, M. Klang, *Spyware: Paying for Software With our Privacy*, 17 Intl. Rev. L. Computers & Tech 313 (2003). .

102. Moshchuk, A., Bragin,T., Gribble, S., Levy, H., *A Crawler-based Study of Spyware on the Web Department of Computer Science & Engineering University of Washington*, www.cs.washington.edu/homes/gribble/papers/spycrawler.pdf. (accessed Mar. 2, 2006).

103. Spyware is used by the law enforcement and intelligence community for investigation, while criminal groups use it for identify theft, advertisers use it for spying on custom-

2. *Spyware's Internal Workings*

Application-based Spyware can be divided up according to the architecture of the Spyware itself. The first architecture includes Spyware applications that attach themselves to downloadable programs (*e.g.*, Kazaa (http://www.kazaa.com)).[104] In this case, the Spyware loads with the application, but the Spyware remains hidden from the user.[105] The Spyware could be mining data[106] on the respective machine, listening to its instant messaging ("IM") or monitoring voice conversations that utilize voice over internet protocol telephony ("VoIP").[107] Also, Spyware in this form installs itself by providing users with e-mail message enhancements[108] or perhaps a new toolbar of some type or another (*e.g.*, Yahoo toolbar (http://toolbar.yahoo.com) or Google toolbar (http://toolbar.google.com)).[109]

The other class of Spyware technology utilizes web-based applications running on Java or Active X platforms for installation on the end-user's system.[110] These applications download additional source code to install the Spyware.[111] Here, the end-user does not explicitly consent to the Spyware's installation,[112] and the capabilities of the Spyware installed are analogues to those installed in the other class. This Spyware is different because end-users normally never explicitly consent to the installation and are unaware that the Spyware invaded the computer[113]

---

ers, and the security industry uses it for observing persons under surveillance. *See generally* Ganesh K. Vanapalli & Kevin C. Desouza, *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05) - Track 1, Securing Knowledge Assets and Processes: Lessons from the Defense and Intelligence Sectors*, at 27b.

104. *See* KaZaA Hack 3.0, *at* http://www.kazaahack.net/home.html (last visited Nov. 17, 2005). *See also* Radcliff, *supra* n. 79.

105. *See generally* Nathaniel Leibowitz et al., *Deconstructing the Kazaa Network*, 3rd IEEE Workshop on Internet Applications (WIAPP'03), Santa Clara, CA, USA (2003).

106. *See generally* Roger Thompson, *Cybersecurity & Consumer Data: What's at Risk for the Consumer?* Testimony before the U.S. House of Representatives Subcommittee on Commerce, Trade, and Consumer Protection, http://www.iwar.org.uk/comsec/resources/consumerrisk/Thompson1799.htm. (accessed Dec. 27, 2005).

107. *See generally* Garrie, *supra* n. 85.

108. Martin, *supra* n. 88.

109. *Id.*

110. Schultz, *supra* n. 71, at 366.

111. *Id.*

112. Though, by visiting the Web site, the user may be implicitly consenting based upon the terms of use of that Web site that the user must "click through," typically without reading.

113. An unashamed example of commercial trojan Spyware is a tool that offers five different E-cards: romantic, joke and others with which to ensnare your victim. *See* Pete Simpson, *New Blends of Email Threats Credit Control*, Software World (Mar. 1, 2003). This new Spyware can be used to spy on unsuspecting parties and all that is needed to install the Spyware is the email address of the target. *Id.* This technology is able to snoop remotely on every action taken on the end-user's machine and can be remotely logged,

and it enables hackers to break into an end-user's machine, giving open access to the particular computer's information and programs.[114] These actions perpetrated by Spyware are analogous to associated cookies that the judiciary has permitted to transpire.[115]

B. INTERNET COMMUNICATIONS

This section presents a broad overview of the technology involved in both Internet voice and data transactions. It discusses, in a non-technical manner, how Voice over Internet Protocol (VoIP) transmits voice communications over the Internet. Additionally, this section provides an in-depth discussion and analysis detailing the inner workings of clickstream data and how it interacts with cookie technology.[116]

1. *Phone Conversations Using VoIP*

VoIP is a technology by which oral communications can be transferred from circuit-switched networks to or over Internet Protocol networks, and vice versa.[117] VoIP transforms standard oral telephone signals into compressed data packets that are sent over the Internet Protocol.[118] The audio signal at this point is captured either by way of a microphone or received from line input.[119] This analog representation is then converted to a digital representation at the audio input device. The

---

which has notable industrial espionage repercussions and judicial repercussions as well. *Id.*

114. For example, users that shared a program Surfer Bar via e-mail distribution which embedded in the HTML formatted e-mail a hidden link to a site that dropped an executable into the C drive, and then exploited a known vulnerability in Internet Explorer to automatically execute a Visual Basic script. Once installed, this application inserted multiple files on user systems and refreshed the system's registry keys, start-up page, and IE references every couple of seconds. The skill required by the end-user to remove this application extended beyond the average user's skill set. The application embedded many references to porno and gambling sites such that the user's browser was non-functioning. Surfer Bar was a from of adware, however, it could have just as easily been used to deliver a malicious Spyware application that stole and/or mined a user's machine.

115. *See generally* Garrie, *supra* n. 85.

116. *Id.* at 98 ("Overview of Voice Over Internet Protocol Privacy Rights").

117. For one overview of the emerging market for VoIP, *See* Peter Grant, *Ready for Prime Time: A new Internet-based phone technology has an un-catchy acronym: VoIP*, Wall St. J. Jan. 12, 2004 at R7. Growth projections for VoIP vary widely, but the Wall Street Journal reported that "[b]y the end of this year [2004], about 20% of the new phones being shipped to U.S. businesses will use VoIP technology, according to Yankee Group, a technology consulting firm based in Boston. By 2007 that figure should exceed 50%, and eventually almost all of the new phones shipped will use VoIP, Yankee Group predicts." *Id.*

118. *See* Ulysses Black, Voice Over IP (Prentice Hall 1995).

119. *See* ITUT (International Telecommunications Union) Recommendation H.225.0, Call Signaling Protocols and Media Stream Packetization for packet Based Multimedia Communication Systems (1998).

resulting digital samples are copied into a memory buffer in blocks of frame length. Here, a silence detector decides whether the block is silence or a portion of speech.[120] Prior to transmission over the Internet, the block itself is written to a socket. Once this is completed, the communication is transmitted to another VoIP terminal. This terminal parses the header information and the block of audio is decoded applying the same codec and the samples written into a buffer.[121] Once this step is complete, the block of samples is copied from the buffer to the audio output device.[122] The audio output device makes the digital to analog conversion and outputs the signal.[123] VoIP can be used with either a telephone or a PC as the user terminal.[124] This gives different modes of operation: PC to PC, PC to telephone, telephone to PC and telephone to telephone (via the Internet). All VoIP protocols are application layer protocols.[125]

People have known about the possibilities for wiretapping, but the public itself views such threats to be limited to corporate espionage and criminal activities.[126] To eavesdrop over the switched telephone network there must be physical access to the telephone line and access to some type of hardware device that may or may not be very sophisticated.[127] Wiretapping dangers increase considerably in the VoIP world.

---

120. Based on the detector's evaluation as to whether or not the block is part of talk, it is encoded with the selected codec, then header information is added to block. *Id.*

121. *See generally*, Philip Carden, *Building Voice Over IP*, Network Computing (May 8, 2000).

122. *See generally* Darrin Woods, *Connecting to the Voice World*, Network Computing (Apr. 17 2000).

123. *See* Jon-Olov Vant, IP telephony: Mobility and Security, at 20 (2005) (Ph.D. dissertation in Teleinformatics, Stockholm, Sweden).

124. *See* Rachael King, *Home of the Future*, Telephony at 10 (June 6, 2005).

125. An application layer protocol is a layer used to transmit Internet communications existing within the TCP/IP framework. Carden, *supra* n. 121. The application layer is defined within the TCP/IP protocols, which are an industry standard group of protocols through which computers find, communicate, and access one another over a transmission medium. *Id.* The protocol group is implemented in the form of a software package known as a TCP/IP stack, which splits the transmission into a number of discrete tasks. Each layer corresponds to a different form of communication. The TCP/IP architecture has four layers; application, transport, Internet, and the physical layer. The transmission of voice communications over the Internet initiates with data being sent from the application layer down the stack to physical layer, where it is then transmitted to the receiver and goes up the stack in the reverse order, ending at the application layer. *Id.*

126. *See* Jay Fitzgerald, *Team to Tie Net Phone Hackers; Industry Aims to Stop Scams Before They Start*, The Boston Herald at 31 (Apr. 26, 2005).

127. VoIP is a solid technology, however, it requires government regulation to ensure a certain level of product reliability and safety for the consumer. *See* Yumi Nishiyama, *Vulnerabilities in Electronic Commerce Communication: IM & VoIP*, World Bank, Washington D.C (2003). Up until today security issues in the data and voice worlds have been seen to be completely separate by the users. With advent of VoIP users are now exposed to the

The equipment or software needed is much more sophisticated, but well within the reach of a sixteen year old hacker. Data sniffing tools[128] are readily available and these tools will soon be enhanced to become aware of the new VoIP protocols.[129] Because in an office environment VoIP traffic travels over a data network that is used by all of the regular users of the corporate LAN,[130] any or all of the conversations traversing a network could theoretically be compromised by anyone with a regular connection on the network.[131] VoIP packets could be identified and stored for re-assembly to be played back at a later time.[132] The idea that only Internet traffic is at risk is simply wrong.[133] Privacy for oral traffic could be vastly enhanced by the use of encryption.[134] Most corporate

---

risks of sending data over the Internet while simultaneously having the expectation that telephone conversations are between the parties involved. *Id.* VoIP is vulnerable because convergent technologies lead to weakness from multiple points. *See id.*

In addition, VoIP must address the security holes in cell-phones that arise from the transport mechanisms used when mobile phones are in use. *Id.* Adjoining these problems is the reality that cell tracker tools have evolved and people can eavesdrop with much greater ease on cellular transmission. *Id.* Also, hackers can intercept data with greater ease than before when the data travels in soft zones (unprotected) between legitimate users and cell towers. *See* Martius Miettinen, Study, *Lehrstuhl für Kommunikationssicherheit [IT-Security in the Automobile Domain]*, Ruhr University at Bochum (Germany), *at* http://www.cs.helsinki.fi/u/mjmietti/seminaariS03/automobilesecurity.pdf (last visited Dec. 27, 2005). Thus, transmitting information in digital form raises new vulnerabilities and digital device can be used either for fiscal or and privacy violations. Also, the VoIP systems run on vulnerable software, so the systems must contend with all of these possible holes. *Id.*

128. Data sniffing tools are used primarily to steal or transmit end-user data from end-users' machines with or without their knowledge. P. J. Bruening & M. Stephen, *Spyware: Technologies, Issues, and Policy Proposals*, 7 J.Internet L. 9, at 3-8 (2004). Advertisers can use these tools to identify what sites end-users have visited and deliver targeted ads to the end-user's computer. *Id.* For example, if a user visits a Florida cruise site followed by a later visit to a golfing site, advertising using data sniffing tools will serve advertisements to the end-user's computer about golf course vacations in Florida. *Id.* Data sniffing tools encompass cookie technology, Spyware, adware.

129. *See* J. Daniels, *Scumware.biz Educates About Dangers of Adware/Scumware*, 5 Computer Security Update 2 (2004).

130. Local Area Network.

131. *See* Dale J Long, *The Lazy Person's Guide to Voice Telephony—Part II.*, CHIPS 43 (Spring 2004).

132. *See* Amie J Singer, *Debate Over Voice-Over Internet Protocol Benefits: Cost-Effectiveness, Security Concerns at Heart of Uncertainty*, San Diego Business Journal 51 at 19 (Dec. 17, 2001).

133. *See* Ian Shepherd, *VoIP The Maturity of Internet Telephony Technology Opens Up Network Safety Concerns Voice Over IP: Finding a Balance Between Flexible Access and Risk of External Attack*, Computer Weekly at Networks 34 (Apr. 19, 2005).

134. Philip Bednarz (President and CEO, Netergy Microelectronics Inc., Santa Clara, Calif), Communications Design Conference, *Security Considerations at Forefront of VoIP Design*, Electronmic Engineering Times 63 (Sept. 23, 2002) (adding word [e]ncryption and decryption are CPU intensive and take time. If the overall latency of a VoIP call is greater than approximately 250 milliseconds, the quality of the call will noticeably be affected).

networks, however, do not encrypt VoIP calls.[135]

One of the attractive features provided by VoIP is the ability to locate intelligence at various points in the network. Gatekeeper or call-manager type devices, which authenticate users and establish connections,[136] can physically reside on any server[137] on the network. This is really a two-edged sword. Logging information about user calls may be useful for billing or tracking purposes, but these logs can also become targets for hackers. If this type of information becomes compromised, it can create serious concerns for organizations or individuals.[138] Unfortunately, the home user is usually unaware of any of these vulnerabilities when they purchase or use VoIP technology.[139]

2. *Cookies, Clickstream Data, and Internet Commerce*

This section examines both the technical capabilities and uses of clickstream data, examining in detail the role played by cookie technology. Cookies are defined as information packets transmitted from a server[140] to an end-user's Web browser, such as Microsoft Internet Explorer or Mozilla Firefox, which then re-transmits information back to the server each time the browser accesses a specific server's Web page.[141] Cookies usually store information used for authentication, identification or registration of an end-user to a Web Site.[142] This infor-

---

135. *Supra* n. 34.

136. *See generally*, K. Percy & M. Hommer, *Tips From the Trenches on VoIP*, Network World Fusion (Jan. 2003) (a gatekeeper is an optional component of an H.323 enabled network that provides central management and control services. Gatekeepers usually deliver the following in relation to VoIP services: (1) address translation; (2) bandwidth management; and (3) routing functionality; H323 is a technical standard that enables VoIP companies to create interoperable Internet telephony solutions); *see also* Michele Rosen, *The Maturing of the Internet Telephony Market- Market is Maturing- Internet / Web / Online Service Information*, ENT at 48 (Mar. 18, 1998).

137. *See* Wikipedia, Server, *at* http://www.wikipedia.org/wiki/server (a server is a computer system or a set of processes on a computer system providing services to clients across a network).

138. *See* Edwin Mier et al, *Breaking Through IP Telephony:VoIP Security Wares,* Network World 84, at 84-88 (May 24, 2004).

139. *See generally* Mike Lee, *Beware! Bugs can attack Net phones; They may be cheap but they are also vulnerable to hackers, say experts, who advise installing anti-virus patches*, The Straits Times (Singapore) (Aug. 22, 2004), http://it.asia1.com.sg/newsdaily/news001_20040823.html (accessed July 13, 2005); *See* Fitzgerald, *supra* n. 126.

140. *See Supra* n. 137 and accompanying text.

141. *See generally* Shaun B. Spencer, *Reasonable Expectations and the Erosion of Privacy*, 39 San Diego L. Rev. 843, 910 (2002); *see* Kent Walker, *The Costs of Privacy*, 25 Harv. J.L. & Pub. Policy 87, 113-117 (2001).

142. *See generally* Daniel J. Solove, *Privacy and Power: Computer Databases and Metaphors for Information Privacy*, 53 Stan. L. Rev. 1393, 1458-60 (2001); Lawrence Jenab, Comment, *Will the Cookie Crumble?: An Analysis of Internet Privacy Regulatory Schemes Proposed in the 106th Congress*, 49 Kan. L. Rev. 641, 667-68 (2001); Rachel K. Zimmerman,

mation enables the end-user's Web browser to maintain state, a continuous relationship between the end-user's computer and the server of a specific site. In the mid-1990s Web portals began to use cookie technology,[143] enabling companies to deliver user-specific solutions for each machine that accessed their Web pages.[144] Cookies allowed Web sites to track end-user activities specific to particular Web portals by placing electronic tracks or markers on end-user machines.[145] Collectively these cookie-driven markers create a trail of information commonly referred to as "clickstream data."[146]

Initially, clickstream data was used to garner basic information from a web user,[147] such as the type of computer an individual used to access the Internet, the type of Internet browser utilized, and the identification of each site or page visited.[148] As technology evolved, however,

---

Note, *The Way the "Cookies" Crumble: Internet Privacy and Data Protection in the Twenty-First Century*, 4 N.Y.U. J. Legis. & Pub. Policy 439, 459-60 (2000).

143. *See In re DoubleClick, Inc.*, 154 F. Supp. 2d 497, 502-03 (S.D.N.Y. 2001) ("Cookies are computer programs commonly used by Web sites to store useful information. . . .")

144. A sampling of Web sites that would be impacted are as follows: www.yahoo.com; www.google.com; www.wamu.com; www.schwab.com; www.ibm.com. Adjoining these Web sites are a slew of intranet and Web applications that utilize cookies and clickstream data for authentication. Not only will business be impacted, also a large number of government enabled Web applications. Some government sites using this technology are mentioned, *at* http://www.ombwatch.org/article/ articleview/587/1/71?TopicID=1 (last visited Mar. 30, 2005). *See also*, United States Department of Commerce News, *Retail E-commerce Sales for the Fourth Quarter 1999 Reach $5.3 Billion*, Mar. 2, 2000, Census Bureau Reports, *available at* http://www.census.gov/mrts/www/ecomm.html (last visited Dec. 27, 2005

145. *See* Jerry Berman & Deirdre Mulligan, *Privacy in the Digital Age: Work in Progress*, 23 Nova. L. Rev. 551, 554 (1999) ("The data trail, known as transactional data, left behind as individuals use the Internet is a rich source of information about their habits of association, speech, and commerce. Transactional data, click stream data, or 'mouse droppings,' as it is alternatively called, can include the Internet protocol address ('IP address') of the individual's computer, the browser in use, the computer type, and what the individual did on previous visits to the Web site, or perhaps even other Web sites.")

146. Once a user has accessed a Web site that uses cookie technology or an affiliated site, the embedded cookie on the hard drive begins collecting data about the user's Web activities. In four reported cases: *Chance v. Avenue A, Inc.*, 165 F. Supp. 2d 1153,1155; *In re Intuit Privacy Litigation*, 138 F. Supp. 2d 1272, 1274; *In re DoubleClick*, 154 F. Supp. 2d at 502-03; and recently *In re Pharmatrak, Inc.*, 329 F.3d 9, 12 (1st Cir. 2003). Cookie technology was used by Web Sites to mine personal information from the users' machines. *See also supra* n. 7.

147. *See generally* Survey FTIT: *A Key Technology for Online Profitability*, Financial Times (London, England) at 5 (Apr. 3, 2002); Randolph E. Bucklin & Catarina Sismeiro, *A Model of Web Site Browsing Behavior Estimated on Clickstream Data*, 40 Journal of Marketing Research 249 (Aug. 2003).

148. *See* Fusun Feride Gonul, *Stereotyping Bites the Dust; Marketers No Longer Focusing On Demographic Profiling*, Pittsburgh Post-Gazette (Pennsylvania) Sooner Edition B3 (February 26, 2002); Karen Dearne, *You are Being Monitored Online*, The Australian, at Features 31 (Sept. 24, 2002).

so did clickstream data.[149] Today, when an individual discloses certain information during a visit to a Web site via their Personal Digital Assistant, cell phone, Blackberry, laptop computer, iPod, or desktop computer, it is possible that the Web site will be collecting clickstream data of a much more personal nature.[150] Clickstream data is used because centralized web server technologies cannot store and sort the vast amounts of data required to deliver the respective Web solutions to each individual user to a site or to authenticate a user.[151] Thus, Web sites off-load certain information to the end user's device where it is stored in special text files called "cookies."[152] These cookies provide the Web site with a mechanism through which they collect and store data on the storage device of the visitor's machine, thereby enabling a Web site to record, track, monitor, and generate customized dynamic pages reflecting the stored data.[153]

The functionality of the data mining industry and most web portals would be severely limited, if not rendered useless in the absence of clickstream data or cookies.[154] Although it is possible for authentication processes to be retooled so they require users to log in or to affirmatively consent to monitoring by cookies or clickstream data tracking, it is highly unlikely that fully informed end-users would interact with sites that track, monitor, and perhaps sell personally identifiable information.[155] Internet companies currently rely heavily on tracking clickstream data to deliver customized services and advertisements to

149. Elbert Lin, *Prioritizing Privacy: A Constitutional Response to the Internet*, 17 Berkley Tech. L.J. 1085, 1104 (2002) (clickstream data is a trail of information that a user leaves behind while browsing on the Web). *See generally,* Edelstein, H.A., *Pan for Gold in the Clickstream*, Informationweek.com at 77-91 (Mar. 12, 2001); Jane Kaufman Winn & James R. Wrathall, *Who Owns the Consumer? The Emerging Law of Commercial Transactions in Electronic Consumer Data*, 56 Bus. Law. 213, 234-35 (2000).

150. *See In re Pharmatrak, Inc.*, 329 F.3d 9, 15 (1st Cir. 2003) (This information could be in the form of passwords, e-mail addresses, credit card numbers, medication, stock trades, and other sensitive information that your machine stores).

151. *See generally*, Moeller, R. A., Distributed Data Warehousing Using Web Technology, American Management Association (AMACOM New York, 2001).

152. *See* M.J. Berry & G.S Linoff, Mastering Data Mining: The Art and Science of Customer Relationship Management (Wiley Computer Publishing, New York 2000); S. Colin, *The CRISP-DM Model: The New Blueprint for Data Mining*, 5 Journal of Data Warehousing 4, 13-22 (Fall 2000).

153. *See generally*, J. Han, M. Kamber, Data Mining: Concepts and Techniques, (Morgan-Kaufmann Academic Press, San Francisco 2001); B. Rajagopalan, R. Krovi, *Benchmarking Data Mining Algorithms*, Journal of Database Management 13, 25-36 (Jan-Mar 2001).

154. Blakley, A., Garrie, D. B., & Armstrong, M.J., *Coddling spies: why the law doesn't adequately address computer spyware*, available at http://www.law.duke.edu/journals/dltr/articles/2005dltr0025.html (Feb. 2, 2006).

155. *See Special Report – Online Marketing: Traffic Control*, Precision Marketing at 17 (Mar. 18, 2005).

Internet users.[156] For example, DoubleClick, an Internet advertising company, had stockpiled over 100 million user profiles[157] by the year 2002. Since then, the technology and ability to profile has greatly improved and Internet companies now rely on clickstream data more than ever before. Cell phones, PDAs, Web TVs, Xboxes, laptops, desktops, and other web-enabled applications utilize clickstream data in either a direct or derivative form. Thus, if the courts construe the Wiretap Act to protect clickstream data as it protects other electronic communications,[158] both the business world and government functions would be disrupted.[159] To demonstrate this expansive reliance on cookie technologies, simply view the cookies stored on your own computer.[160]

V. HOW DOES SOFTWARE RELATE TO THE LAW

I have developed a topical legal guide intended to help the judiciary navigate the application of the inner workings of software to a particular area of the law. The intent of the topical guide is to help the judge to learn and construct the necessary fact specific analysis for the matter that is presented before the court.

The first section examines tort litigation, noting that the sparse case law and the current state of the law makes the establishment of software specific analysis points for tort litigation impractical. The second section discusses antitrust litigation and examines the current law and develops a framework for courts to utilize specific to antitrust related litigation. The third section focuses on intellectual property, split into three subsections: (1) Patent; (2) Copyright; and (3) Trade-secret. Each sub-section reviews the current intellectual property as it relates to software and then discusses the specific steps of analysis a court may wish to apply when hearing litigation on the respective intellectual property area. The fourth section analyzes the blossoming field of trespass, which to date has usually been in the context of sending unsolicited e-mail

156. *See In re DoubleClick*, 154 F. Supp. 2d at 503-507; *See also* Linoff, *supra* n. 152.

157. *See In re DoubleClick*, 154 F. Supp. 2d at 505; *Chance v. Avenue A, Inc.*, 165 F. Supp. 2d 1153, 1155-57 (W.D. Wash. 2001); *In re Pharmatrak, Inc.*, 329 F.3d 9, 21 (1st Cir. 2003).

158. 18 U.S.C. §2510 (12).

159. See Chairman of the FTC Robert Pitofsky et al, *Privacy Online: Fair Information Practices In the Electronic Marketplace*, Federal Trade Commission's Report to Congress 7, 10-20, 29-33, (May 2000).

160. An end-user can view all of the cookies stored on a local machine using Internet Explorer by following these steps: (1) Open Internet Explorer; (2) Select "Internet Options" under the "Tools" menu; (3) Click on the "General" tab and click the "Settings" button; (4) Click the view files button; (5) Sort files by type by clicking on "Type"; (6) Find documents of the type labeled "Text Document." To see the information stored by the cookie in its raw and likely unintelligible format, double-click on one of these text files containing "cookie" in its file name.

messages. This section analyzes the current legal remedies that are applied by the judiciary and then recommends analysis specific to trespass and software. The fifth section examines digital discovery and discusses the current legal setting and proposes a set of questions applicable to the software systems involved in this process. In addition it presents the benefits to the court of having a firm grasp of how software inner workings impact this area of the law. The sixth and final section addresses software and the tort of invasion of privacy and develops the analysis that a court may utilize when seeking to analyze software in this context. This section also discusses the benefits that a court receives by having a firm grasp of how software works.

A. WHAT SPECIFIC ANALYSIS SHOULD A COURT PERFORM WHEN HEARING TORT LITIGATION?

A review of the literature demonstrates that software litigation rarely involves theories of tort law.[161] For example, in 1985 and 1986, there were several incidents involving computer software used to monitor radiation therapy.[162] These incidents involved Therac 25, a computerized therapeutic radiation machine manufactured that administered overdoses to several different patients causing serious injury or death.[163] Additionally, computer software defects have been blamed for jets flying to the wrong destination[164] and causing prototype F-16 fighters' computer software to potentially flip the jets upside down whenever they crossed the equator.[165] Today, the law surrounding software tort liability is relatively undeveloped considering the size of the software industry and the importance of software in our lives.

When software vendors face potential liabilities the cause of actions is usually under negligence, professional malpractice, implied warranty,[166] or strict liability.[167] Most of these cases settle or are dismissed. For example, courts have ruled there is no such thing as computer

161. *See e.g.*, P.E. Bradley & J.R. Smith, Prevention, *Liability Issues Regarding Defects in Computer Software*, 19 Prod. Liab. L. & Strategy 5 (Nov. 2000); M.R. Maule, Comment, *Applying Strict Products Liability to Computer Software*, 27 Tulsa L.J. 735 (Summer 1992); P.T. Miyaki, Comment, *Computer Software Defects: Should Computer Software Manufacturers be Held sStrictly Liable for Computer Software Defects?*, 8 Santa Clara Computer & High Tech. L.J. 121 (May 1992); L.A. Weber, Note, *Bad Bytes: The Application of strict Products Liability to Computer Software*, 66 St. John's L. Rev. 469 (Spring 1992).

162. Cheryl S. Massingale & A. Faye Borthick, *Risk Allocation For Injury Due to Defective Medical Software*, 2 J. Prod. Liab. 181, 181-84 (1988).

163. *Id.*

164. Michael Rogers & David L. Gonzales, *Can We Trust Our Software?,* Newsweek at 70 (Jan. 29, 1990).

165. *Id.*

166. *See Against Gravity Apparel, Inc. v. Quarterdeck Corp.*, 267 A.D.2d 44, 44, 699 N.Y.S.2d 368, 369 (1st Dept. 1999).

software programming malpractice.[168] There are no reported cases holding a software manufacturer strictly liable for defects in the software.[169]

Virtually all software comes with licensing agreements that disclaim nearly all claims of performance and all warranties[170] express or implied.[171] Programs downloaded via the Internet require the end-users to click on a box indicating that they agree to the limitations of the agreement. Programs sold as "shrink-wrap"[172] software come with a license that advises the user not to open the package unless they agree to the disclaimers of the software vendor. Most importantly these limitations have been held effective. Therefore, manufacturers have been able to escape liability under the implied warranties that apply to most legal products, including defective software. When software litigation has arisen and settled it usually has been in regards to the use of software in

167. One major impediment to the application of strict liability is the widely held belief that it is virtually impossible to guarantee that software is error-free. Several articles, however, have been written on the issue of strict liability and software including: Gemignani, *Product Liability and Software*, 8 Rutgers Computer & Tech. L.J. 173 (1981); Nycum & Lowell, *Common Law and Statutory Liability for Inaccurate Computer-Based Data*, 30 Emory L.J. 445 (1981); Comment, *Strict Products Liability and Computer Software: Caveat Vendor*, 4 Computer L.J. 373 (1983).

168. *Hospital Computer Systems, Inc. v. Staten Island Hosp.*, 788 F. Supp. 1351, 18 U.C.C. Rep. Serv. 2d 140 (D.N.J. 1992). *See also RKB Enterprises Inc. v. Ernst & Young*, 182 A.D.2d 971, 582 N.Y.S.2d 814 (3d Dept. 1992).

169. Frances E. Zollers et al., *No More Soft Landings For Software: Liability for Defects in an Industry That Has Come of Age*, 21 Santa Clara Computer & High Tech. L.J. 745, 766. There is a second group of cases that never came to a full determination because they were settled. In one case, as told by one commentator, radiation patients received overdoses of radiation due to a software bug in the accelerator that administered the doses. *See* Julia A. Tyde, Comment, *Medical Computer Software: Rx for Deadly Errors*, 4 Software L.J. 117 (1990). Two patients died and several others sustained serious injuries. *Id.* at 137. The estate of one of the deceased patients filed a lawsuit against the manufacturer of the accelerator and the cancer center where the patient received his treatments. *Id.* The complaint alleged that the product was defective and unreasonably dangerous and not fit for its intended use. In other words, the claim was brought in strict liability. The case was eventually settled for an undisclosed amount of money. *Id.* In the fall of 2004, Medtronic, Inc. undertook a voluntary recall of software application cards for one of its medical products in the face of reports of patient injury and death. *See Medtronic: Company Announces Voluntary Recall of Software Application Card*, Med. & Law Wkly. at 191 (Oct. 22, 2004).

170. Regarding advertising claims and puffing, *see* P.L. Dykas, Comment, *Opinion v. Express Warranty: How Much Puff Can a Salesman Use, If a Salesman Can Use Puff to Make a Sale?*, 28 Idaho L. Rev. 167 (1991/1992); W.K. Lewis, *Toward a Theory of Strict 'Claim' Liability: Warranty Relief for Advertising Representations*, 47 Ohio St. L.J. 671 (1986); C.R. Brown, *An Analysis of the Interpretation of the "Basis of the Bargain" Language of Section 2-313*, 104 Com. L.J. 316 (Fall 1999). *See also* 4 McCarthy on Trademarks and Unfair Competition § 27:38 (4th ed).

171. *See e.g.*, *M.A. Mortenson Co., Inc. v. Timberline Software Corp.*, 140 Wash. 2d 568, 998 P.2d 305, 41 U.C.C. Rep. Serv. 2d 357 (2000).

172. *Latham and Associates, Inc. v. William Raveis Real Estate, Inc.*, 218 Conn. 297, 589 A.2d 337, 14, 14 U.C.C. Rep. Serv. 2d 394 (1991).

potentially life-threatening situations.[173]

B. ANTITRUST AND SOFTWARE

Courts have only begun to grapple with the application of antitrust litigation to the software arena, resulting in an immature body of law where there are few clear guidelines for both the software industry and antitrust enforcement authorities. This section examines how software and the judiciary's understanding of software can influence antitrust laws prohibiting certain acts of monopolization, attempts to monopolize, refusals to deal, and tying arrangements.

Software litigation in antitrust cases frequently involves judicial findings as to whether a litigant has, or is likely to obtain, monopoly power or market power in some market.[174] The task of establishing the relevant market to determine whether a firm has acquired market power is difficult even where the products at issue are well understood by the judiciary. The determination of the relevant market is a question of fact,[175] which often cannot be resolved by the courts in a precise and mechanical fashion.[176] The elements of monopolization and the abuse of monopoly power are also questions of fact. This further establishes uncertainty in determining whether a company has an obligation to deal with relevant market players in an industry.[177]

When a court presides over an antitrust case involving software, it should initially seek to understand the underpinnings of the software at issue in the dispute. The *Microsoft Corp*[178] antitrust litigation demonstrates the need for courts to understand software from its initial conception, to the actual code, to the intent behind the code provisions, and culminating with the operation of the final software product.

The Microsoft case examines demanding antitrust questions on vari-

---

173. Richard Raysman & Peter Brown, *Strict Product Liability for Software and Data*, N.Y.L.J. at 3 (Sept. 15, 1988).

174. *See United States v. E.I. du Pont de Nemours & Co.*, 351 U.S. 377, 391 (1956) (finding that monopoly power is the power of a party to control prices or exclude the competition); *Eastman Kodak Co. v. Image Technical Servs. Inc.*, 504 U.S. 451, 464 (1992) (defining market power to be the ability of an entity to raise price and restrict the output.)

175. *See California Steel and Tube v. Kaiser Steel Corp.*, 650 F.2d 1001, 1003 (9th Cir. 1981) ("Defining the relevant market, the threshold requirement under Section Two, is essentially a matter of resolving factual issues.")

176. "[A] certain amount of fuzziness is often inherent in the task of defining a relevant geographic market, and the final decision must often be a compromise." *United States v. Empire Gas Corp.*, 537 F.2d 296, 303-304 (8th Cir. 1976), *cert. denied*, 429 U.S. 1122 (1977).

177. "As a general rule, a company has the right to deal with whomever it chooses. *Associated Press v. United States, 326 U.S.* 1, 14-15 (1945).

178. *See United States v. Microsoft Corp.*, 980 F. Supp. 537, 541-34 (D.D.C. 1997)

ous legal issues including tying.[179] Tying is the conditioning of the sale of one product (the tying product) on the buyer's agreement to buy another distinct product (the tied product) from the seller.[180] Tying can be unlawful under both Section 1 of the Sherman Act[181] and Section 3 of the Clayton Act.[182] The Court of Appeals for the District of Columbia Circuit sitting en banc was unable to conclude[183] that Microsoft had illegally tied its Web browser software product to its Windows Operating System in violation of section 1 of the Sherman Act.[184]

In *United States v. Microsoft Corp*[185] the Department of Justice ("DOJ") Antitrust Division alleged, among other things, that Microsoft illegally tied its Internet browser to its operating system,[186] because it was unnecessary for Microsoft to package its Internet browser with Windows 95.[187] Applying a per se tying analysis,[188] the district court held

179. *See* Andrew Chin, *Decoding Microsoft: A First Principles Approach*, 40 Wake Forest L. Rev. 1 (2005); Richard L. Gordon, Antitrust Abuse in the New Economy: The Microsoft Case, at 129 (Edward Elgar Publishing, 2002) (Including a browser as a component of Windows without a price increase and vigorously seeking to improve and promote that browser are not conventional predatory acts.)

180. Section 7 (15 U.S.C. §18) of the Clayton Act prohibits mergers or acquisitions that may substantially lessen competition or those that would tend to create a monopoly. Section 7a of the Clayton Act (15 U.S.C. §18a), requires parties to certain transaction to file notification with the Federal Trade Commission ("FTC") and the Antitrust Division of the Department of Justice prior to consummation of the transaction, assuming the transaction meets certain minimum dollar thresholds.

181. *See* 15 U.S.C. §§1,2.

182. *See Northern Pac. Ry. v. United States*, 356 U.S. 1, 3 (1958); *International Salt Co. v. United States*, 332 U.S. 392, 396 (1947).

183. The district court held Microsoft liable for tying under section 1 of the Sherman Act. *United States v. Microsoft Corp.*, 84 F. Supp.2d 9 (D.D.C. 1999); *United States v. Microsoft Corp.*, 87 F. Supp.2d 30 (D.D.C. 2000). However, the Court of Appeals reversed and remanded the legal question of Microsoft's tying liability finding that the government had failed to define what constituted a "browser" and what constituted a software product being tied to a "good market." *See United States v. Microsoft Corp.*, 253 F.3d 34, 95 (D.C. Cir. 2001). This outcome compelled the government to drop the tying claim and as result the tying claim was never adjudicated. *See New York v. Microsoft Corp.*, 224 F. Supp. 2d 76, 95 (D.D.C. 2002)

184. *See* Complaints, *United States v. Microsoft Corp.*, 84 F. Supp. 2d 9 (D.D.C. 1999) (Nos. 98-1232 & 98-1233).

185. *See United States v. Microsoft Corp*, 253 F.3d 34, 71 (D.C. Cir. 2001) (applying a rule of reason analysis, the court agreed with the district court that Microsoft's exclusive deals were anti-competitive, and that Microsoft's exclusive deals with ISP providers to foreclose competition in the market to Internet browsers violated § 2 of the Sherman Act.)

186. *United States v. Microsoft Corp.*, 147 F.3d 935, 935 (D.C. Cir. 1998). 1998 WL 327855 (D.C. Cir. June 23, 1998). *But see United States v. Loew's*, 371 U.S. 38 (1962) (holding that package licensing of multiple pieces of intellectual property may constitute an illegal tying arrangement).

187. *Microsoft*, 147 F.3d at 935.

188. *See* "Antitrust Guidelines for the Licensing of Intellectual Property," § 5.1 (U.S. DOJ F.T.C. 1995) (declaring that a tying arrangement will be deemed per se unlawful

that Microsoft did engage in illegal tying arrangements.[189] The district court found Microsoft's actions were the "result of a deliberate and purposeful choice to quell incipient competition before it reached significant proportions."[190] The unanimous, *en banc* Court of Appeals for the District of Columbia Circuit, however, vacated the district court's decision that Microsoft's bundling conduct resulted in a *per se* unlawful tying arrangement.[191] The Court of Appeals found that the unique interoperability of Microsoft's Windows operating system and Internet Explorer browser counseled against the use of the *per se* standard.[192] Rather, the court held that the alleged tying arrangement was better analyzed pursuant to the rule of reason approach[193] because such an analysis would provide Microsoft the chance to prove that the benefits from tying the products together outweighed any change in consumer choice.[194] This outcome compelled the government to drop the tying claim, and as a result the tying claim was never fully adjudicated.[195]

The Microsoft case demonstrates the important role software can play in antitrust litigation. The Court of Appeals became fixated on Microsoft's software code,[196] and perhaps would have benefited by understanding the difference between software design and software

---

under the following scenarios: (1) the tying and tied product are separate and distinct; (b) the seller conditions the sale of the tying product on the buyer's agreement to buy the second product; (c) the seller has market power in the market for the tying product; and (d) the tie-in affects a notable portion of commerce. The DOJ, tasked with enforcing per se tying, analyzes tying arrangements under the rule of reason.

189. There are two basic types of tying arrangements: (1) an agreement by a party to sell one product but only on the condition that the buyer also purchase a different product, or (2) an agreement that he will not purchase that product from any other supplier (sometimes called a "tie-out" or "negative tie")—could be present in reverse engineering situations. *See* Robert H. Lande & Sturgis M. Sobin, *Reverse Engineering of Computer Software and U.S. Antitrust Law*, 9 Harv. J.L. & Tech. 237, 236 (1996).

190. *Microsoft*, 253 F.3d at 51.

191. *See id.* at 92.

192. *Id.* Under the per se rule, a court does not conduct an extensive analysis of the market or entertain any defenses that the conduct had a procompetitive effect. Rather, the court merely considers if the conduct occurred and, if so, the court declares the conduct unlawful. *See* § 1 of the Sherman Act (15 U.S.C § 1).

193. Under a "rule of reason" analysis, a court will weigh the anticompetitive effects against the procompetitive effects of restraint. If the restraint is determined to have a net anticompetitive impact on competition within a relevant market, the restraint will be deemed unlawful. *See* §1 of the Sherman Act (15 U.S.C §1).

194. *Supra* n. 193, at 92.

195. *See New York v. Microsoft Corp.*, 224 F. Supp. 2d 76, 95 (D.D.C. 2002).

196. Some commentators argue that the failure of the court to differentiate between software code and software product substantially contributed to Microsoft's victory. *See* Andrew Chin, *Decoding Microsoft: A First Principles Approach*, 40 Wake Forest L. Rev. 1, 4 (2005) (arguing that if the focus had been on the software product that the code creates the outcome would have been different).

code.[197] The Court of Appeals' failure to fully examine the software at issue could have contributed to Microsoft's victory since the Court of Appeals seemingly did not examine the design of the software product and rather focused on the code itself.[198]

1. *Software Analyses in antitrust litigation*

This section develops recommended procedures for identifying roles played by particular software products in antitrust disputes. The simplistic view of equating code with software is misplaced, as software is more complex than just the lines of code. In general terms, a court should recognize that software is a product of software design, code, testing, and development methodology all applied in various degrees in delivering a software product. Consequently, courts hearing antitrust issues should examine each of these four aspects while also considering the interplay between various phases involved in software creation. For instance, if a court fails to consider the software's design, which violates antitrust laws, and instead examines only the code, which does not violate antitrust law, the court will provide guilty parties with the ability "to code around" the law. Courts that focus solely on the actual text of the computer code risk encouraging software programmers to create software where the code falls within the bounds of antitrust law, yet the cumulative operation of the software provisions operates in violation of the antitrust laws.

This process can also be perpetuated when the courts make formalistic inquires into the code itself while ignoring the software developer's intent and the multitude of other steps in the software production process (*e.g.*, high-level software design). Legal decisions drafted in this fashion risk encouraging software programmers to design code that performs an otherwise illegal task. This raises the issue that courts will be confronting which is that the low-level design can be viewed as a direct

197. *See generally* Software, *supra* Section III.

198. The court recognized that a less deferential standard would leave it in the position that Judge Jackson found himself, where the court became "enmeshed in a technical inquiry into the justifiability of product innovations." *Microsoft II*, 147 F.3d 935, 951 (quoting Response of *Carolina, Inc. v. Leaseco Response, Inc.*, 537 F.2d 1307, 1330 (5th Cir. 1976)). The court explained that "[c]ourts are ill equipped to evaluate the benefits of high-tech product design." *Microsoft II*, 147 F.3d at 952. Moreover, "the limited competence of courts to evaluate high-tech product designs and the high cost of error should make them wary of second-guessing the claimed benefits of a particular design decision." *Id.* at 949 n.12. The D.C. Circuit stated that the "commingling of code . . . alone is not sufficient evidence of true integration. Commingling for an anti-competitive purpose (or for no purpose at all) is what we refer to as 'bolting.'" Arguably, if the D.C. Circuit had viewed the software with a different perspective, then the outcome would have been different.

expression of the code itself.[199] However, the high-level design is an expression of the business requirements and is simply one interpretation of the high-level design requirements. For example, Judge Jackson's statement in the Microsoft hearing defined an application to be a "software program . . . that perform[s] specific user oriented tasks,"[200] and failed to differentiate the software program's code from its design.[201]

Generally, when a court is hearing a case involving a software antitrust dispute, it may wish to consider the following factors: first, the court should examine the software's code, focusing on the coding methodology used in the development phase,[202] which will allow the court to better determine whether the code and the methodology were intentionally selected to hinder integration of software products. For example, company X creates a highly successful software program which it initially markets as stand-alone product, however, over time the company X increasingly integrates the product and the software code into other software products it owns. Company Y files suit against company X asserting that company X unlawfully "tied" the software's functionality to its other products for anti-competitive purposes. In performing this evaluation, a court that has a firm grasp of the fundamentals of software would include in it a review of the low-level design, which may demonstrate that the company X's behavior is pro-competitive and the integration of their two products is intended to enhance the consumer's experience by creating a new product.[203] The court's understanding of software could perhaps result in a legal decision that is firmly grounded in an understanding of both the technology and the law. In summation, the court should review the software code, examining whether the code itself indicates intent to inhibit product integration with other products or whether it indicates intent to create a new, superior product in response to customer demand.

Second, the court should review the programming and design methodologies[204] utilized in the software development process. The programming and design methodologies provide the court with a unique insight as to whether the design of the software is indicative of the company's intent to design software that violated the law. For example, the court

199. *See e.g., Alcatel USA, Inc v. DGI Techs.*, Inc., 166 F. 3d 772 (5th Cir. 1999); *Lasercomb Am., Inc v. Reynolds*, 911 F.2d 970 (4th Cir. 1990).

200. *See United States v. Microsoft Corp.*, 84 F. Supp. 2d 9, 12 (D.D.C. 1999).

201. *See e.g.*, Victoria Slind-Flor, *Tackling High Tech: Jurists Learn to Cope with the Brave New World*, Natl. L.J. at 1, 28 (Oct. 19, 1992).

202. *See* Coding, Testing, Software, *supra* Section III(B)(1).

203. *See e.g.*, Davis, Steven J. and Murphy, Kevin M. "A Competitive Perspective on Internet Explorer." American Economic Rev. 90 (2000) (Pages and Proceedings of the One Hundred Twentieth Annual Mtg. of the American Economic Association, 184-187).

204. *See* Software Development Methodologies, *supra* Section III(C).

could ascertain that the design and programming methodologies that were applied establish sufficient evidence to merit the court's holding that the company violated the antitrust laws.

Third, courts should examine the business requirement documentation that is associated with the software since it usually contains a detailed list of business objectives.[205] These explicit requirements should not be evaluated focusing on the code or design, which may not violate the law, but rather on the issue of whether the software business requirements sought to hinder or eliminate rival products. For example, company X could have business requirements that make integration with comparable products difficult, require complimentary software specifically from that vendor for optimal software operation, or other analogous objectives, but the software's design and code do not violate black letter law. Consequently, courts hearing software antitrust disputes should be sure to examine the software's business requirements in addition to the code and the programming and design methodologies.

Finally, a court may seek to evaluate the software's design,[206] focusing on high-level design documentation, to determine whether the developers intended to hinder, halt, or thwart similar applications from operating on the same machine.[207] The high-level design is not always expressed in the code or low-level design because the code represents an interpretation of the high-level design. Finally, antitrust tying legal disputes at times become further complicated when the software at issue is an "operating system"[208] (hereinafter "OS") software. Challenges arise because the OS provides an application platform and a user interface for end-user computers while managing the computer hardware itself.[209] For example, Microsoft Office can be designed such that courts are tasked with the difficult role of differentiating OS aspects falling within the domain of the operating system from those that are tightly integrated, but not within the sphere of the OS.

In the case where a court determines that a disputed software program is classified as an OS, it must review the OS with different criteria because the OS provides the basic foundation upon which all other application-based software operates. Thus, dominant OSs are ripe for anti-

205. The court should pay particular attention to the context under which the software program was developed because this is likely to provide insight on what business factors motivated the software development.

206. *See* Software Design, *supra* Section III(B).

207. *See* Software Development Methodologies, *supra* Section III(C).

208. The Operating System ("OS") software manages the CPU and related hardware components including: keyboard, monitor, storage media and communication devices. The OS provides the hardware management and user interface functions into a single product commonly termed OS.

209. Prominent examples include the Apple Mac OS, Linux, and Microsoft Windows.

trust abuse because they can be designed to lack interoperability with certain other vendor programs, unfairly pushing them out of the market to the detriment of the consumer. Courts should utilize the following factors to determine whether the software component is a functional part of the OS or whether the component is a separate and distinct product:

- Whether the software product in question exposes Application Program Interfaces ("API")[210] to third-party users. For example, the court in *New York v. Microsoft Corp*[211] defines, examines, and discusses the role of APIs in the antitrust suit against Microsoft.[212] The court discussion demonstrates the context by which a court may wish to examine the API in an antitrust suit.
- Whether the software component is used by a large number of software applications (*e.g.*, Internet Explorer is used by multiple software components such as Adobe and Macromedia.)
- Whether the software component delivers services that are inaccessible to end-users.
- Whether the software component performs "low-level" services, meaning that the software interacts with specific devices. Here, the court should evaluate the relevant low-level design activity because low-level design defines the modules that comprise each of the components identified in the high-level design.[213]
- Whether the component performs housekeeping functions. Here, the court should review both the low-level design and the code because this functionality is expressed in the design or the code itself.[214]
- Whether the software component performs functions analogous to other software products available in the market from other vendors.

In summation, the complex nature of antitrust litigation when intertwined with software requires the court to firmly grasp the underpinnings of software before applying the law, otherwise the court runs the risk of issuing a legal decision that is not equitable to the parties.

---

210. API's support the functions of applications by exposing interfaces, called "application programming interfaces," or "APIs" in the context of the Operating Systems. *U.S. v. Microsoft Corp.* 84 F.Supp.2d 9, 12 (D.D.C. 1999). These are synapses at which the developer of an application can connect to invoke pre-fabricated blocks of code in the operating system. These blocks of code in turn perform crucial tasks, such as displaying text on the computer screen. Because it supports applications while interacting more closely with the PC system's hardware, the operating system is said to serve as a "platform." *Id.*

211. *Supra* n. 203.

212. *See generally id.*

213. *See* Neal Whitten, *Managing Software Development Projects: Formula for Success* 39 (2nd ed. 1995).

214. *See id.* The low-level design "identifies each programming decision path and may be documented by using a design language, graphic flows, and so on, or simply by writing English narratives." *Id.* Low-level design is also sometimes called detailed design.

C. Software Disputes Involving Intellectual Property

Software can be protected under copyright, patent and trade secret laws and litigation often involves the combination of these issues. Software litigation in the intellectual property arena can be divided into three distinct areas: (1) patents; (2) trade secrets; and (3) copyright. Below is a brief discussion of relevant software litigation case law in each of these three legal areas. However, the section focuses on describing the software components that courts should consider when evaluating these disputes.

1. *Software & Copyright*

Today's law relevant to copyright and software covers a wide range of legal areas.[215] A recent example of where such analysis is demonstrated in *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*,[216] where all nine justices agreed that distributors of technology that enables copyright infringement can be found liable for their users' actions, but only if a plaintiff can point to evidence that the distributor took "affirmative steps" to foster infringement.[217] The Court in reaching this conclusion analyzed not only the marketing actions of *Grokster*,[218] but the code of *Grokster* and the various software components that *Grokster* has utilized in developing its technology.[219]

a. What is the case law in software copyright suits?

The case that captured how software influences the outcome of copyright litigation is the Supreme Court's recent decision in *Grokster* where the Court reversed the Ninth Circuit's decision, holding *Grokster* liable for infringement by third parties using the device, regardless of the device's lawful uses.[220] In that case, *Grokster* provided at no cost to the

215. *See e.g.*, Pamela Samuelson, Symposium: *The Semiconductor Chip Protection Act of 1984 and its Lessons: Creating a New Kind of Intellectual Property: Applying the Lessons of the Chip Law to Computer Programs*, 70 Minn. L. Rev. 471, 516 (1984); John Swinson, *Copyright or Patent or Both: An Algorithmic Approach to Computer Software Protection*, 5 Harv. J.L. & Tech. 145 (1991).

216. *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*, 125 S.Ct. 2764 (2005) (finding that merely distributing such a device, without more, is not sufficient to give rise to liability. Those "who distributes a device with the object of promoting its use to infringe copyright, as shown by clear expression or other affirmative steps taken to foster infringement, is liable for the resulting acts of infringement by third parties.") (Souter, J., majority opinion). *Id.* at 2780.

217. *Id.*

218. *Id.* at 2774.

219. *Id.* at 2776–2779.

220. *Id.* at 2775-2783 (holding that "[o]ne who distributes a device with the object of promoting its use to infringe copyright, as shown by clear expression or other affirmative steps taken to foster infringement, going beyond mere distribution with knowledge of third-

end-user, software that MGM claimed was exclusively utilized for illegal downloads of music and video.[221] *Grokster's* software architecture was designed to be decentralized, thereby making it difficult for copyright holders to litigate directly against end-users because the software was unable to centrally track and monitor the end-user's actions.[222] The issue before the Court was whether the software companies could be found secondarily liable for the copyright infringement of their users under the Copyright Act, thereby enabling MGM to (1) halt the infringement at the source and (2) pursue monetary compensation for the violations committed by the end-user of the software.[223] Initially, the District Court granted *Grokster* summary judgment,[224] and the Ninth Circuit of Appeals affirmed that decision,[225] finding that the precedent of *Sony v. Universal Studios*[226] ruled out liability for a distributor whose product has "substantial non-infringing uses."[227]

The Supreme Court in *Grokster* ruled that while *Sony* bars some claims of secondary liability, it does not preclude all, and that the preliminary findings establish that both companies actively and intentionally promoted the illegal use of their software.[228] The Court's holding emphasizes the importance of reviewing software characteristics in the context of copyright litigation. The opinion discuses the software mechanisms that were utilized to enable file sharing.[229] For example, Justice Souter (writing for the majority) discussed *Grokster's* software's ability to distribute files that an end-user requests, noting that Grokster's software platform utilized a software engine that was similar to that of the engine used to operate Napster.[230] In summation, the Court's reversal of the Ninth Circuit holding that no liability existed and in-depth discussion of the software design demonstrates the role software can play in copyright disputes.[231]

---

party action, is liable for the resulting acts of infringement by third parties using the device, regardless of the device's lawful uses.")

221. *Id.* at 2775.

222. *Id.* at 2785-86.

223. *Id.* at 2771–75.

224. *Id.* at 2774.

225. *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*, 380 F. 3d 1154 (9th Cir. 2004).

226. *Sony Corp. of America v. Universal City Studios, Inc.*, 464 U.S. 417 (1984) (addressing the question of whether VCR manufacturers could be held liable for infringement).

227. *Id.* at 423-24.

228. *Grokster*, 125 S.Ct. at 2772. *See generally*, Jane C. Ginsburg, *Copyright and Control Over New Technologies of Dissemination*, 101 Colum. L. Rev. 1613 (2001).

229. *Grokster*, 125 S.Ct. at 2773 – 2774.

230. *Id.* at 2773.

231. *Id.* at 2780 – 81.

b. Analysis: Copyright from Software Perspective

The steps of analysis for a court to apply when hearing a copyright suit, arising from the use of software, will require the court to examine the software design, architecture, code, and perhaps at times the software documentation.

The courts should examine both the *high-level design* and *the design methods* used in comparison with the software code. The aspects of the high-level design and the design methods that the courts should focus on are the development and design methodologies utilized. For example, if a court, in reviewing the design method, determines that it utilized the Rational Unified Process[232] (hereinafter "RUP"), the court should examine the objects and ascertain whether their design intended to skirt the legal constraints imposed by the law. When the court reviews the design of the software code at a low-level, it should see if the low-level design indicates that the software developers coded with the intent to either utilize copyright infringing technology or designed the software to carefully skirt prior judicial interpretations.

The *Grokster* holding demonstrates that the courts must consider software at various levels, including but not limited to the design of the software itself.

- What does the software documentation indicate with regards to the end-user's abilities? For example, does the software documentation discuss and describe how end-users can utilize the software to infringe upon copyrighted material?
- Does the software design indicate that the software designer intended to utilize and profit from technology that carefully skirts the black letter of the law? Here, the court should focus on whether the design and relevant documentation indicates whether the software developers had knowledge and seemingly coded around the legal holding,[233] and had knowledge that the product violated the spirit of the law.[234]

232. *See* Wikipedia, Rational Unified Process, http://www.wikipedia.org/wiki/Rational_Unified_Process (accessed Dec. 3, 2005). Rational Unified Process is an iterative software design method created by the Rational Software Corporation, now a division of IBM. It describes how to deploy software effectively using commercially proven techniques. It is not a process but a process framework or meta-model. It encompasses a large number of different activities, and is designed to be tailored, in the sense of selecting only the needed features suited for a particular software project, considering its size and type.

233. *Metro-Goldwyn-Mayer Studios Inc. v. Grokster, Ltd.*, 125 S.Ct. 2764, 2779 (*citing Sony Corp. v. Universal City Studios*, 464 U.S. 417, 439 (1984)). The Supreme Court in *Grokster* stated that "where evidence goes beyond a product's characteristics or the knowledge that it may be put to infringing uses, and shows statements or actions directed to promoting infringement, Sony's staple-article rule will not preclude liability."

234. *Sony*, 464 U.S. at 439 ("If vicarious liability is to be imposed on Sony in this case, it must rest on the fact that it has sold equipment with constructive knowledge" of the potential for infringement).

In the coming years, it is likely that these theories will be tested further in courts as the number of claims is likely to grow as peer-to-peer software develops substantial non-infringing utility. Bryer's opinion in *Grokster* wrote that the "now-unforeseen noninfringing uses that develop for peer-to-peer software . . . the foreseeable development of such uses, when taken together with an estimated 10 percent noninfringing material, is sufficient to meet Sony's standard."[235] Unless a court examines the software as a whole, it is unlikely for a court to ascertain whether or not the software violates copyright law.

2. *Software & patent litigation*

The general consensus held by the courts is that computer programs are entitled to some form of patent protection. When hearing patent infringement disputes involving software, courts should examine a combination of the software's design, code, business requirements, and architecture. Courts for instance often hear cases involving claims for overly broad software patents[236] that prevent competitors from entering the market by restricting the software they can write which operates with the patented system software. If the these courts examine the software's code, they could mistakenly hold that the patent was properly granted and permissible, instead of finding that it was[237] overly broad and invalid due to the developer's intentions and the software's design.

a. What is the current state of patent law and software?

In *In re Alappat*,[238] a case from the Federal Circuit Court,[239] the court held that software programs containing mathematical algorithms that perform specific functions are patentable.[240] Interestingly, the

235. *Grokster*, 125 S.Ct. at 2789-90.

236. Wikipedia, *System Software*, *at* http://www.wikipedia.org/wiki/System_software (last visited Dec. 27, 2005) (System software is a generic term referring to any computer software whose purpose is to help run the computer system. Most of it is responsible directly for controlling, integrating, and managing the individual hardware components of a computer system).

237. *See* Steven C. Carlson, *Patent Pools and the Antitrust Dilemma*, 16 Yale J. On Reg. 359, 364-65 (1999).

238. *In re Alappat*, 33 F.3d 1526 (Fed.Cir.1994).

239. *Contra Gottschalk v. Benson*, 409 U.S. 63, 71-72 (1972) (holding that programs were considered "algorithms" and not patentable); *but see Diamond v. Diehr*, 450 U.S. 175, 184-185 (1981) (holding that a particular software-related invention contained patentable subject matter, despite the fact that in several steps of the process a mathematical equation and a programmed digital computer were used); *see generally State Street Bank v. Signature Fin. Group*, 149 F.3d 1368 (Fed. Cir. 1998) (finding that a computer software program that produces a "useful, concrete, and tangible result" is patentable subject matter under §101 of the United States patent laws).

240. *Id.*

Alappat majority's finding enabled patents to be awarded to software while noting in dicta that certain types of mathematical subject matter cannot be patented unless they can be reduced to some type of a practical application.[241] The Alappat court further found that software could be considered a machine when used in conjunction with a digital computer. Software programs running on general-purpose computers create new machines because a general-purpose computer becomes a special purpose computer when performing the program's instructions. The court concluded that a computer operating pursuant to software might represent patentable subject matter, provided that the subject matter meets all the other requirements of Title 35.[242] Currently, under U.S. patent law a mathematical algorithm is not patentable if the patent claim preempts the entire algorithm, but may be patentable if it applies the algorithm to accomplish a specific technical purpose.[243] The strong protection provided by patent laws is increasingly important because it induces software development firms to create and market new software since their hard work and effort can now be protected by patent law.[244]

b. How should a court analyze software arguments in a patent dispute?

Courts hearing software patent litigation claims should examine the software's code, design, architecture, and documentation. Different disputes will require different applications of these four concepts. Software patent litigation cases can be divided into two general areas where software patent disputes are likely to arise: (1) Novelty[245] and (2) Subject Matter.[246]

---

241. *In re Alappat*, 33 F.3d 1526, 1543 (Fed. Cir. 1994).

242. *See also Diehr*, 450 U.S. at 185 ("This Court has undoubtedly recognized limits to § 101 and every discovery is not embraced within the statutory terms. Excluded from such patent protection are laws of nature, natural phenomena, and abstract ideas.")

243. *In re Alappat*, 33 F.3d 1526 (Fed. Cir. 1994).

244. *See* Jeffrey J. Blatt, *Software Patents: Myth vs. Virtual Reality*, 17 Hastings Commun. & Ent. L.J. 795, 806 (1994-1995). A party may seek a submarine patent which is a patent application that has been pending in the Patent Office for several years while the relevant industry evolved, potentially embracing the technology that is the subject of patent application. The holder of the submarine patent, once granted, can unfairly gain from an industry that has unwittingly moved forward and adopted technology on which it must now pay royalties to the patentee.

245. 35 U.S.C. § 101 (2005) ("Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefore, subject to the conditions and requirements of this title.")

246. *See Kewanee Oil Co. v. Bicron Corp.*, 416 U.S. 470, 483 (1974) (finding that "[n]o patent is available for a discovery, however useful, novel, and nonobvious, unless it falls within one of the express categories of patentable subject matter of 35 U.S.C. § 101.")

If the patent dispute centers on the issue of the patent's novelty, courts should examine the software's code, development methodologies, and functional market impact.[247] Complex software programs seldom include substantial leaps in technology, but rather consist of adept combinations of many ideas. Whether a software program is good does not generally depend as much on how new a specific technique may be, but instead depends on the unique combination of known algorithms and methods. Consequently, courts should always delve deep into the prior art of a software patent when novelty is at issue. Additionally, when courts are determining whether software code establishes novelty, they can also analyze the code by examining the software's low-level design.

In addition, when the patent dispute centers on whether the software patent encompasses valid subject matter[248] the court should examine various aspects of a software application, including: the software methodology; software design documents; and the software code.[249] The software methodology may provide insight into what the software product was based upon and business reasons driving its development. The software design document can be indicative of the basis of the software and what it was intended to perform. Finally, the software code[250] and comments may establish the functional intent of the code in the context of the specific software.[251] Nonetheless, the court can use these criteria as a guide when considering subject matter software patent disputes.

Finally, courts can be asked to determine whether the software is in compliance with the requirement that the claims within the patent are sufficiently described in the specification.[252] In this area, the analysis would utilize the same aspects as those used for Subject Matter disputes.

3. *Trade Secret*

The law today relevant to trade secret actions and software hinges on state law and not federal law. The passage of the Uniform Trade

247. *See* Greg Aharonian, *Does the Patent Office Respect the Software Community?*, IEEE Software at 87-89 (July/Aug., 1999).

248. *See e.g. Diamond v. Chakrabarty*, 447 U.S. 303, 309 (1980); Bonito *Boats, Inc. v. Thunder Craft Boats, Inc.*, 489 U.S. 141, 154 (1989); *Diamond v. Diehr*, 450 U.S. 175, 182 (1981); *In re Alappat*, 33 F.3d 1526, 1542 (Fed. Cir. 1994) ("Thus, it is improper to read into § 101 limitations as to the subject matter that may be patented where the legislative history does not indicate that Congress clearly intended such limitations.")

249. For further discussion on methods for software development, review *supra* nn. 58-71 and accompanying text.

250. For further discussion on the Software Code, review Coding, Testing, Software, *supra* Section III(B)(1).

251. *See e.g.*, Wesley L. Austin, *Software Patents*, 7 Tex. Intell. Prop. L.J. 225, 252 (1999) (providing an in-depth discussion on software patentable subject matter under 35 U.S.C. § 101).

252. 35 U.S.C. § 112, para. 2 (1994 & Supp. 1998).

Secrets Act (hereinafter "UTSA") by a majority of states in one form or another has resulted in many similarities in state trade secret laws. For a litigant to have a cause of action under the UTSA there must be *misappropriation*[253] of a trade secret;[254] the stealing of a trade secret is perhaps the clearest example of misappropriation of a trade secret.[255] Courts have also found that a breach of an agreement, confidential relationship, or duty resulting in the acquisition or disclosure of confidential information can be classified as a misappropriation of a trade secret.[256] In addition, the courts have found that trade secrets can include computer code and software in general.[257]

The UTSA states that in order to qualify as a trade secret, two elements must be present. The first is "independent economic value . . . from not being generally known."[258] The second is that it must be protected by "efforts that are reasonable. . .to maintain its secrecy."[259] Once a plaintiff establishes that the information qualifies as a trade secret, a "misappropriation" must be proved. UTSA Section 1 defines misappropriation as obtaining through "improper means."

The law today relevant to trade secrets actions and software is demonstrated by reviewing *Trandes Corp. v. Guy F.Atkinson Co.*, a case from the United States Appeals Court for the Fourth Circuit.[260] In that case, the Trandes Corporation brought suit against the Guy F. Atkinson Company (Atkinson) and the Washington Metropolitan Area Transit Authority (WMATA).[261] Trandes alleged that WMATA improperly disclosed and Atkinson improperly acquired and used the "Tunnel System." The Tunnel System is a computer program written by Trandes's presi-

253. *See e.g., BP Chemicals Ltd. v. Jiangsu Sopo Corp.*, 285 F.3d 677, 683 (8th Cir. 2002); *Sip-Top, Inc. v. Ekco Group, Inc.*, 86 F.3d 827, 833 (8th Cir.1996) (defining misappropriation of trade secrets under Minnesota law as either "(1) improper acquisition of a trade secret; or (2) disclosure or use of a trade secret without consent"); *Sokol Crystal Prods., Inc. v. DSC Communications Corp.*, 15 F.3d 1427, 1429 (7th Cir.1994); *accord DTM Research, L.L.C. v. AT & T Corp.*, 245 F.3d 327, 332 (4th Cir. 2001); *Nora Beverages, Inc. v. Perrier Group of Am., Inc.*, 164 F.3d 736, 750 (2d Cir.1998).

254. *See* Restatement (Third) of Unfair Competition, § 39 cmt. f (1988) (requiring secrecy for trade secret status); *see also id.* at § 40 (finding defendant liable for misappropriation when she uses trade secret information without owner's consent).

255. *Hexagon Packaging v. Manny Gutterman & Associates*, 120 F. Supp. 2d 712, 720 (N.D. Ill. 2000).

256. *See* William E. Hilton, *What Sort of Conduct Constitutes Misappropriation of a Trade Secret*, 30 IDEA 287, 285-196 (1990).

257. *See e.g., University Computing Co. v. Lykes-Youngstown Corp.*, 504 F.2d 518 (5th Cir. 1974); *see also* Ohio Rev. Code Ann. § 1333.61(D)(Anderson 1995).

258. *IDX Systems Corp. v. Epic Systems Corp.*, 285 F.3d 581, 583 (7th Cir. 2002).

259. *See e.g., IDX Systems Corp. v. Epic Systems Corp.*, 285 F.3d 581, 583 (7th Cir. 2002); *see also* Ohio Rev. Code Ann. § 1333.61(D)(Anderson 1995).

260. *See generally Trandes Corp. v. Guy F. Atkinson Co.*, 996 F. 2d 655 (4th Cir. 1993).

261. *Id.* at 657.

dent, James Brusse, to perform survey calculations for the construction of subway tunnels. The Court of Appeals affirmed the District Court finding that the owner of trade secret computer software can maintain the secrecy of the source code but freely distribute the object code,[262] which means that the object code was not a trade secret.[263] For example, a software company can provide their software to the end-user as object code and protect their source code by not releasing that with the software, but only the object code is needed to execute the software on the end-user's machine. Interestingly, *Trandes*, by restricting circulation of the software, maintained the secrecy of the object code, and this court concluded that the object code was also a trade secret.[264] The *Trandes* Court also ruled "no reasonable jury could have concluded that the structure and organization of the software was unique or was not generally known in the industry."[265] There are limits, however, to what parts of computer software constitute trade secrets.[266] In *IDX Systems Corp v. Epic Systems Corp.*, the Seventh Circuit rejected a claim "effectively asserting that all information in or about its software is a trade secret," adding that such a claim was "not plausible."[267] In summation, the case law on trade secret litigation is varied in accordance with the underlying state laws.

a. What are the best steps of analysis?

When hearing trade secret suits courts must examine the development methodology used, the architecture, and the code. Trade secret claims arise under state common law, and therefore, their effectiveness depends on whether a particular jurisdiction is willing to classify the actions to constitute a theft of trade secrets, as well as the requirements that individual jurisdiction may have for proving sufficient security and value of the information. However, the analysis pertaining to the software itself will be fairly uniform. The court should examine the source code because it is virtually impossible to duplicate source code without access to the original source code.[268] Typically, if the source code for the new software is virtually identical to the original, a court should

262. *Id.* at 663.

263. *Id.*

264. *Id.* at 662-63.

265. *Id.* at 662.

266. *See* Software Development, *supra* Section III (explaining what software development is).

267. *Id.*

268. The concept being that one of the critical pieces of information to examine in a lawsuit for misappropriation of software is the source code for the "new" software.

find this indicative that the software code was misappropriated.[269]

While misappropriation of trade secrets is unlawful, it is perfectly legal for a former employee to use general knowledge and skills gained through former employment.[270] Therefore, the court may also need to consider the coding language that was used to develop the software itself.[271] However, if the plaintiff alleges that the software design was misappropriated, rather than the source code, the court should review the "new" software design documents.[272] Since identical design documents are highly unlikely, one of the pieces the court should examine is the language and structure of the software design documents. The court may examine the extent to which software developers physically protected not only the software itself, but the related documentation, manuals, and technical bulletins that outline or give insight into the structure and operation of the software. Branding materials, advising customers to restrict access to them, and requiring physical protections are all practical and important ways of protecting these valuable clues in trade se-

269. *See e.g., Aries Info. Systems, Inc. v. P. Mgt. Systems Corp.*, 366 N.W.2d 366, 369 (Minn. App. 1985) (finding likely misappropriation where defendants produced no evidence of how new software was created); *Healthcare Affiliated Servs., Inc. v. Lippany*, 701 F. Supp. 1142, 1149 (W.D. Pa. 1988) (finding defendants' failure to turn over new products evidence of misappropriation). Indeed, in *Computer Assocs. Intl., Inc. v. American Fundware Inc.*, 831 F. Supp. 1516, 1520 (D. Colo. 1993), defendant's destruction of the source code was originally viewed as a basis for entering a default judgment of liability, a decision that was set aside only when it appeared that plaintiff had also lost or destroyed critical evidence. *But see Comprehensive Technologies Intl., Inc. v. Software Artisans, Inc.*, 3 F.3d 730, 736-7 (4th Cir. 1993) (accepting defendants' explanation that many small software companies do not keep source code but instead overwrite it or use an erasable "whiteboard" for development work), *Vacated* on consent of parties, 1993 U.S. App. LEXIS 28601(4th Cir. 1993).

270. For example, object-oriented programming encourages programmers to re-use the programming from a previous project. This, of course, works well provided the programmer does not change jobs and takes to his new employer the ideas or the complete class libraries of previously developed and tested objects. In such a scenario, it is possible that the code will look remarkably similar but may not have been misappropriated; instead the court will have to evaluate the employment relationship to determine whether misappropriation occurred. In another example, the Java programming language also exerts a considerable amount of control over how a Java program is written. In the interests of making "better" programs (that is, those that are more reliable and that do not do inappropriate things on users' computers when they are downloaded by the user from the Internet), Java lacks some of the more sophisticated (meaning complicated and often misused) features of C++. These constraints also serve to induce similarities in two programs, even though these programs were truly developed independently.

271. Generally, trade secrets law tries to strike a reasonable balance between protecting intellectual property rights and an employee's right to earn a livelihood. *See generally supra* note 261.

272. *See* Software Design *supra* Section III(B) (explaining what software design is).

cret disputes.[273]

D. Software and Trespass

Software and the issue of trespass have arisen in the context of sending unsolicited e-mail messages. For example, Spyware could be designed to trespass on the machine while the code itself may not be in violation of the respective trespass statute. However, the software design may support a claim.

1. *What is the current state of the law for trespass actions pertaining to software?*

In *eBay, Inc. v. Bidder's Edge, Inc.*,[274] eBay sought an injunction against Bidder's Edge (hereinafter "BE") to prohibit Bidder's Edge from accessing the eBay Web site with automated search technology.[275] eBay's terms of use included a prohibition on the use of "any robot, spider, other automatic device, or manual process to monitor or copy our web pages or the content contained herein, without our prior expressed written permission."[276] eBay, in its complaint, alleged a variety of causes of action including trespass to chattel.[277]

Interestingly, this court begins with the proposition that "electronic signals sent by BE to retrieve information from eBay's computer system are . . . sufficiently tangible to support a trespass cause of action."[278]

---

273. *See Picker Intl. Corp. v. Imaging Equip. Servs., Inc.*, 931 F.Supp. 18, 37 (D. Mass. 1995); *Anacomp, Inc. v. Shell Knob Servs., Inc.*, 1994 U.S. Dist. LEXIS 223 (S.D.N.Y. Jan. 7, 1994); *ISC-Bunker Ramo Corp. v. Altech, Inc.*, 765 F.Supp. 1310, 1322-23 (N.D. Ill. 1990); *Technicon Data Systems Corp. v. Curtis 1000, Inc.*, 224 U.S.P.Q. 286, 290 (Del. Ch. 1984).

274. 100 F. Supp. 2d 1058 (N.D. Cal. 2000).

275. *Id.* at 1063-64.

276. *Id.* at 1060 (admitting that it was not clear whether the version of the User Agreement in effect at the time BE began using its automated search program prohibited such activity or that BE agreed to eBay's terms of use).

277. *Id.* at 1063.

278. *Id.* at 1069 (citing *Thrifty-Tel Inc. v. Bezenek*, 46 Cal. App. 4th 1559, 1566 (Cal. App. 4th Dist. 1996) (holding that electronic signals generated over a telephone line are sufficient for use in a trespass to chattels claim). In *Thrifty-Tel*, a long distance telephone company brought an action against parents based upon the children's use of computer access to make long distance telephone calls without accruing charges. The court held that "trespass to chattel, although seldom employed as a tort theory in California (indeed, there is nary a mention of the tort in Witkin's Summary of California Law), lies where an intentional interference with the possession of personal property has proximately caused injury." *Id.* at 1566 (citation omitted). The court explains the development of trespass to chattel beginning with a physical touching, moving to an indirect touch, such as dust particles, through microscopic particles, and to "electronic signals generated by the Bezenek boys' activities." *Id.* at n. 6. That court also cites to Indiana and Washington State courts criminalizing the activity of computer trespass. *Id.* at 1567 n. 7 (*citing State v. McGraw*, 480 N.E.2d 552, 554 (Ind. 1985); *State v. Riley*, 846 P.2d 1365, 1373 (Wash. 1993)).

Most courts are satisfied that invading invisibly with electronic signals is a sufficient invasion for trespass to chattels.[279]

The court in eBay begins its analysis with two elements necessary "for trespass based on accessing a computer system."[280] First, the plaintiff must show that the defendant "intentionally and with out authorization interfered with plaintiff's possessory interest in the computer system."[281] Furthermore, plaintiff must prove that "defendant's unauthorized use proximately resulted in damage to plaintiff."[282] In applying the facts of the case to this law, the court holds that even if eBay's Web site were publicly accessible, the fact that eBay "explicitly notifies automated visitors that their access is not permitted,"[283] and the fact that BE continued to use an automated system "even after eBay demanded BE terminate such activity,"[284] eBay had shown that BE's activities lacked authorization because they exceeded the scope of any consent.[285] eBay repeatedly and explicitly notified BE to cease using an automated system.[286]

The eBay courts seem to gloss over the requirement of damage to the chattel itself. It seems to assume that the use of another's computer constitutes intermeddling that creates sufficient damage for liability.[287] The court finds that even though eBay does not claim BE's sending between 80,000 and 100,000 requests to eBay's computer systems each day "has led to any physical damage to eBay's computer system, nor does eBay provide any evidence to support the claim that it may have lost revenues or customers based on this use. eBay's claim is that BE's use is appropriating eBay's personal property by using valuable bandwidth and capacity and necessarily compromising eBay's ability to use that capacity for its own purposes."[288] The court finds that even though BE's use of the system may only be a small portion of that system's capacity, "BE

279. *See State v. Riley*, 846 P.2d 1365, 1373 (Wash. 1993) (criminalizing the activity of computer trespass).

280. *eBay*, 100 F. Supp. 2d at 1069.

281. *eBay*, 100 F. Supp. 2d at 1069-70.

282. *Id.* at 1069.

283. *Id.* at 1070.

284. *Id.*

285. *Id.* (citing *City of Amsterdam v. Daniel Goldreyer, Ltd.*, 882 F. Supp. 1273, 1281 (E.D.N.Y. 1995) for the proposition that exceeding the scope of consent can subject one to liability in trespass to chattels even though there is not a complete conversion of the chattel. Also citing to *Civic Western Corp. v. Zila Industries, Inc.*, 66 Cal. App. 3d 1, 17 (1977) in the context of trespass to real property for a holding that when limited consent is given and the defendant exceeds that limited consent, a trespass has occurred).

286. *Id.*

287. *Id.* at 1071 (citing the Restatement (Second) of Torts §218 comment (e) which notes the requirement that the person intentionally intermeddling with chattel must have harmed that chattel.

288. *Id.* (citations omitted).

has nonetheless deprived eBay of the ability to use *that* portion of its personal property for its own purposes."[289] The court goes on to find that eBay need not wait for a disaster before applying for relief.[290] The court seems to hold that using a portion of the computer system to the exclusion of the rightful owner is sufficient trespass to chattels.

Unfortunately, the Supreme Court of California created uncertainty with a decision in an e-mail case.[291] In the majority decision, the court distinguishes *eBay* on the basis that some injury must be shown.[292] Furthermore, the court claims that the *Oyster Software* court incorrectly applied California law in stating that actionable trespass to chattel exists simply based on use.[293] The California Supreme Court majority opinion clearly requires some impairment of the function or some impairment of the property.[294] The court also refuses to extend California's common law trespass to chattels to include "otherwise harmless electronic communication whose contents are objectionable."[295] The court analyzes a series of arguments. The opinion eviscerates any potential trespass to chattels cause of action for consumers in California state courts.

2. *What are the best steps of analysis?*

The steps of analysis for a court to apply when hearing trespass suits arising from the use of software will require the courts to examine the development process, the architecture, the code, and the testing that was involved in the development of the software.

In disputes asserted under the common law tort theory of trespass to chattels action,[296] the plaintiff argues that by inserting a code into another person's computer system, the Spyware perpetrator may have entered into the end-user's computer by intermeddling with it and the defendant argues that this was not the intent.[297] "One who commits a trespass to chattel is subject to liability to the possessor of the chattel if, but only if, . . . (b) the chattel is impaired as to its condition, quality, or value, or (c) the possessor is deprived of the use of the chattel for a substantial time, . . ."[298] Trespass to chattels claims arise under state common law, and therefore, their usefulness depends on whether a particular jurisdiction is willing to classify Spyware violations as tres-

289. *Id.* (emphasis added).
290. *Id.* at 1072.
291. *Intel Corporation v. Hamidi*, 1 Cal. Rptr. 3d. 32 (2003).
292. *Id.* at 44.
293. *Oyster Software, Inc. v. Forms Processing, Inc.*, No. C-00-0724 JCS, 2001 WL 1736382 (N.D. Cal. 2001); *Hamidi*, 1 Cal. Rptr. 3d at 44.
294. *Id.*
295. *Id.* at 47.
296. Restatement (Second) of Torts § 218.
297. *Id.*
298. *Id.*

passes, as well as the requirements that individual jurisdiction may have for proving trespass to chattels. Although, trespass to chattels claims can also be hindered if a court finds that an end-user granted the Spyware consent.

The legal analysis by the court will hinge on the respective state law. However, the analysis pertaining to the software itself will be fairly uniform. Here, the court must examine whether the Spyware is constructed such that it takes information from a computer without the person's permission and generally without the person's knowledge. Typically, when one thinks about trespasses to chattels, the actual physical taking of a personal possession has occurred.[299] However, nothing in the definition of trespass to chattels[300] requires that the perpetrator actually possess the chattel. Merely interfering with it, impairing its condition,[301] or depriving the rightful owner of its use for a substantial time[302] will suffice to create liability. Consequently, the court must examine the design and architecture of the Spyware and determine whether Spyware impairs access to the chattel.[303] Arguably, recipients of Spyware are deprived of their ability to use their computer for its intended purposes, however, a court must examine the extent to which the software was designed and coded respective to the loss of value the end-user experiences of their computer as a repository of private information. The court must also examine the software code to determine when and how the Spyware hijacks the computer, for instance, redirecting any Internet search or homepage to a cite the user does not desire and until the user can remove the Spyware, the user is deprived of the use of the as desired chattel for a period of time.[304]

E. Discovery Litigation

In electronic discovery cases, software is used to store, archive, search, and manipulate requested data in different formats. For example, a court hearing a discovery motion may examine whether software applications are constructed in such a way to render the data production costs "unduly burdensome" to the producing party, compelling cost shifting to transpire. In this example, the court should review the design documents, business requirements, and system requests of the producing party to determine whether the litigant had knowledge and consciously

299. *See e.g.*, *Bogart v. Chappell*, 396 F.3d 548, 557 n. 7 (4th Cir. 2005).

300. *Restatement (Second) of Torts* § 218.

301. *Id.* at § 218(b).

302. *Id.* at § 218(c).

303. *Id.* at § 218(b).

304. *Id.* at § 218 (c).

chose not to upgrade their system.[305] The litigants could have based this decision in part on their desire to limit access to the data by making the system itself proprietary, which thereby generally increases the costs of discovery.[306]

1. *What is the current state of the law for e-discovery pertaining to software?*

The amendments in 1970 to the Federal Rules of Civil Procedure ("Rules") attempted to clarify the issue of e-discovery.[307] The Advisory Committee Notes for the 1970 amendments to the Federal Rules of Civil Procedure revised the description of "documents" in Rule 34(a) to make clear that Rule 34 applies to electronic data compilations and that when the data can as a practical matter be made usable by the discovering party only through respondent's devices, respondent may be required to use his devices to translate the data into usable form."[308] In June of 2005, the Standing Committee on the proposed Rules approved amendments that amend Rules 26(b)(2)(B),[309] (C)[310] and create a new Rule,

305. *See generally*, Daniel B. Garrie & Matthew J. Armstrong, *Electronic Discovery and the Challenge Posed by the Sarbanes-Oxley Act*, 2005 UCLA J.L. & Tech. 2 (2005).

306. The company may begin with an "off the shelf" software product, but most companies must modify the software sufficiently that it becomes a custom proprietary piece of software.

307. In 1970, Rule 34(a) of the Federal Rules of Civil Procedure was amended to broaden the definition of the word "document" to include "other data compilations from which information can be obtained, translated, if necessary, by the respondent through detection devices into reasonably usable form." *Anti-Monopoly, Inc. v. Hasbro, Inc.*, 1995 WL 649934 1,1 (S.D.N.Y. 1995) (order compelling production of documents).

308. *See* Proposed Amendments to the Federal Rules of Civil Procedure Relating to Discovery, 48 F.R.D. 487, 527 (1970).

309. *See* proposed Fed. R. Civ. P. 26(b)(2)(B): (stating that "[a] party need not provide discovery of electronically stored information from sources that the party identifies as not reasonably accessible because of undue burden or cost. On motion to compel discovery or for a protective order, the party from whom discovery is sought must show that the information is not reasonably accessible because of undue burden or cost. If that showing is made, the court may nonetheless order discovery from such sources if the requesting party shows good cause, considering the limitations of Rule 26(b)(2)(C). The court may specify conditions for the discovery.")

310. *See* proposed Fed. R. Civ. P. 26(b)(2)(C) (stating that "[t]he frequency or extent of use of the discovery methods otherwise permitted under these rules and by any local rule shall be limited by the court if it determines that: (i) the discovery sought is unreasonably cumulative or duplicative, or is obtainable from some other source that is more convenient, less burdensome, or less expensive; (ii) the party seeking discovery has had ample opportunity by discovery in the action to obtain the information sought; or (iii) the burden or expense of the proposed discovery outweighs its likely benefit, taking into account the needs of the case, the amount in controversy, the parties' resources, the importance of the issues at stake in the litigation, and the importance of the proposed discovery in resolving the issues. The court may act upon its own initiative after reasonable notice or pursuant to a motion under Rule 26(c).")

37(f).[311] Unfortunately, neither the rules nor the comments discuss how or when a federal court should examine the storage software system itself to determine whether e-discovery evasion is occurring.

The result of the current discovery rules have been to a wide range of court order requiring litigants to produce computerized information, including e-mail messages, support systems, software, voice mail systems, computer storage media and backup tapes and telephone records, sometimes at considerable expense.[312] The language of the proposed and current Federal Rules, as well as recent case law, both affirm that e-discovery requests are controlled by the traditional discovery rules provided by Federal Rules 26 and 34.[313] Today, courts have recognized that digital documents are within the realm of discoverable materials[314] and impose penalties when parties act in bad faith in response to production requests.[315] In all cases involving discovery motions, courts are, by some degree, basing their decisions on the software applications involved in the discovery motion.[316] The test applied by the majority of the courts today is based on the test formulated by the *Zubulake* court.[317] The *Zubulake* court developed seven factors which courts should consider

311. *See* proposed Fed. R. Civ. P. 37(f) ("Electronically stored information. Absent exceptional circumstances, a court may not impose sanctions under these rules on a party for failing to provide electronically stored information lost as a result of the routine, good-faith operation of an electronic information system.")

312. *See* Peter Brown, *Developing Corporate Internet, Intranet, and E-mail Policies*, 520 PLI/Pat 347, 364 (1998) (citing *In re Brand name Prescription Drugs Antitrust Litigation*, 1995 WL 360526 (N.D. Ill. June 15, 1995)); Fed. R. Civ. P. 34; *Rowe Ent, Inc. v. The William Morris Agency, Inc.*, 205 F.R.D. 421, 425 (S.D.N.Y. 2002) (one party estimated that it would cost approximately $9,750,000 to restore all of its existing backup tapes).

313. *See* Fed. R. Civ. P. 26, 34; *Anti-Monopoly*, 1995 WL 649934 at 1; *Crown Life Ins. Co. v. Craig*, 995 F.2d 1376, 1383 (7th Cir. 1993); and *Natl. Union Elec. Corp. v. Matsushita Elec. Indus. Co.*, 494 F. Supp. 1257, 1259 (E.D. Pa. 1980).

314. *See e.g., Bills v. Kennecott Corp.*, 108 F.R.D. 459, 461 (D. Utah 1985) ("It is now axiomatic that electronically stored information is discoverable under Rule 34 of the Federal Rules of Civil Procedure if it otherwise meets the relevancy standard prescribed by the rules, although there may be issues in particular cases as to the form of what must be produced.")

315. *See e.g., United States v. Philip Morris USA Inc.*, 327 F.Supp.2d 21 (D.D.C. 2004) (sanctioning defendant $2,750,000 for its failure to follow court's preservation order); *Brick v. HSBC Bank USA*, 2004 WL 1811430 (W.D.N.Y. Aug. 11, 2004) (sanctioning law firm $147,635.74 for discovery failures); *Invision Media Comm., Inc. v. Fed. Ins. Co.*, 2004 WL 396037 (S.D.N.Y. Mar. 2, 2004) (noting plaintiff's discovery misconduct, including misleading statements regarding existence and location of evidence and failure to make reasonable inquiries, warranted sanctions in the form of costs and reasonable attorneys' fees expended by defendant in connection with sanctions motion and certain discovery events).

316. *See e.g., Residential Funding Corp. v. DeGeorge Fin. Corp.*, 306 F.3d 99 (2d Cir. 2002) (acting party's "purposeful sluggishness" that resulted in the non-production of evidence before trial could be sufficient to warrant sanctions).

317. *See Zubulake v. UBS Warburg, LLC*, 217 F.R.D. 309, 320 (S.D.N.Y. 2003).

when deciding whether to grant cost-shifting relief:[318] (1) the extent to which the request is specifically tailored to discover relevant information; (2) the availability of such information from other sources; (3) the total cost of production, compared to the amount in controversy; (4) the total cost of production, compared to the resources available to each party; (5) the relative ability of each party to control costs and its incentive to do so; (6) the importance of the issues at stake in the litigation; and (7) the relative benefits to the parties of obtaining the information.[319] The *Zubulake* court stated that the first six factors of the seven-factor test correspond to the three explicit considerations of Rule 26(b)(2)(iii).[320] The *Zubulake* court held that these changes were necessary because the *Rowe* test generally favored cost shifting, undercutting the presumption[321] that the producing party bears document production costs.[322] Thus, *Zubulake* represents a step towards creating a cost-shifting test that relies on Rule 26(b)(2)'s proportionality factors[323] to maintain consistency with the Federal Rules' presumption that the producing party bears the cost of production.[324] Presently, courts rely upon the seven-factor *Zubulake* test, or some modification thereof, to resolve these e-discovery cost-shifting disputes.[325] In the aforesaid case law analysis, the courts examined in varying degrees the litigants' document storage systems, investigating the document storage software systems themselves. Since software can be designed and coded in several ways, a court that examines the software itself will be better informed in ascertaining whether the e-discovery motion is overly broad or unduly burdensome.

---

318. *Id.* at 322.

319. *Id.*

320. *Id.* at 323. Fed. R. Civ. P. 26(b)(2)(iii) states that "the burden or expense of the proposed discovery outweighs its likely benefit, taking into account the needs of the case, the amount in controversy, the parties' resources, the importance of the issues at stake in the litigation, and the importance of the proposed discovery in resolving the issues."

321. This presumption was established by the Supreme Court in *Oppenheimer Fund, Inc. v. Sanders*, 437 U.S. 340, 358 (1978), where the court interpreted the federal discovery rules as presuming that "the responding party must bear the expense of complying with discovery requests, but he may invoke the district court's discretion under Rule 26(c) to grant orders protecting him from 'undue burden or expense . . . .'"

322. *Zubulake*, 217 F.R.D. at 320.

323. *See id.* at 321; *see also* Fed. R. Civ. P. 26(b)(2)(iii).

324. *See Zubulake*, 217 F.R.D. at 321 (stating that the amount in controversy and the importance of the issues at stake in the litigation should be added to the cost-shifting test to make the test parallel Rule 26 and to balance the Rowe factors that weigh in favor of cost-shifting).

325. *See id.* at 322.

2. *What are the best steps of analysis?*

The steps of analysis for a court to apply when hearing discovery motions under federal law will require the courts to examine the respective storage system software, focusing on high-level business requirements,[326] system architecture, and relevant system documentation. A court's examination of these various aspects will hinge on the context of the motion. Below is an analysis of a court hearing a discovery motion, in which the plaintiff argues that the court should require the defendant to bear the burden of discovery and produce the requested digital documents. The defendant argues that the plaintiff's request is unduly burdensome and overly broad, citing to exorbitant costs of production of documents and amount of resources required to provide the documents. In this example, the court should examine the software applications that contain the alleged data and determine based on the system design whether the system was designed to ensure that retrieval would be cost prohibitive for production or such that the information is not discoverable.

Another aspect of software that a court may seek is the relevant technical and business documentation for the storage system in determining whether the design decisions were based on valid business needs. For example, if the defendant were a cigarette company and the plaintiffs made a motion seeking relevant documents that were stored in a legacy data storage system, a court may elect to focus on the software storage system design documentation and relevant business drivers associated with the delivery of the legacy storage system. Here, the documentation may indicate that the legacy storage system was designed with the intent of making the combined costs of finding and producing data to be excessive thereby increasing odds of settlement, because such documentation provides insight into the knowledge and reasoning behind a decision to migrate, upgrade, or leave untouched a legacy system.[327]

326. Such business requirements can be found embedded in multiple places, such as the financial statements of the business units that discuss the software they purchased or built this year for various projects.

327. When a court hears these arguments it should be remembered that software in comparison to the brain's abilities to search, access, and translate information is simply unequal. The human eye is able to process for typos, synonyms, antonyms, implied meaning, and etc. while software search technology can only execute this in rudimentary form. The best analogy to describe this is that while a human can walk into a storage facility and look for that single page that could be discovered, searching digital information repositories it is like sending a blind man into a document repository, perhaps ten times as large and giving him tools that help identify in an expeditious fashion documents that may contain the smoking gun.

The court must also examine whether the production costs were in line with that of the industry. Most official computer standards are set by one of the following organizations: ANSI (American National Standards Institute); ITU (International Telecommunication Union); IEEE (Institute of Electrical and Electronic Engineers); ISO (International Standards Organization); or VESA (Video Electronics Standards Association). Therefore, a court should review the applicable industry standard. A court may also examine whether the company designed the system such that internal costs of access were significantly lower than external user requests. For example, the storage system may be designed such that the internal costs of displaying and accessing documents is a fraction of an external user's because the storage system requires the end-user to have access to proprietary technology. An additional factor is whether the system documentation indicates whether the company intentionally deferred upgrades to the storage system after litigation began.[328] This is relevant because it may indicate that the company had knowledge that system would have notably lower production costs than that of the existing computer system.[329] These factors should be applied in context and will usually help a court ascertain if there was any intent to subvert the judicial process itself by way of the document storage system.[330]

F. PRIVACY

Software and the tort of invasion of privacy have arisen before the courts primarily in the context of Internet communications,[331] where the user's consent to monitoring was at issue. A court should focus on the following aspects of software: (1) architecture; (2) development methodology; and (3) software code. The court can utilize these different perspectives in determining whether the software was designed and developed with the intent to violate a party's privacy rights. For example, the software product itself may not violate the black letter law. However, it may mine data and instead of intercepting "communications," the

328. For example, a company could design their software storage system to ensure that the costs to produce a document were really high, which in turn would limit the judiciary's willingness to grant the discovery motion and increase the likelihood of the court granting equitable relief to the producing party.

329. *Zubulake*, 217 F.R.D. at 320.

330. *See generally supra* n. 313.

331. *See supra* n. 7. In each case, the court held that no unlawful interception had occurred because, even if the transmission to the third party constituted an "interception" of the user's communications with the Web site, this was done with the consent of the Web site, which was a party to the communication. *But see, In re* Pharmatrak, Inc., 329 F.3d 9, 15 (1st Cir. 2003) (finding that there was no consent under the Wiretap Act, 18 U.S.C. § 2511(2)(d), where a corporate entity had an explicit agreement prohibiting a third-party data from collecting personal identifiable information).

software mines stored data, which is not in itself an illegal act.[332] This example demonstrates that although the software code is not in violation of the law, it would seem that the software violates the spirit of the law by "coding" around the law; therefore, a court may wish to consider reviewing the software to help ensure that it correctly enforces the law.

1. *What is the current state of the law for privacy and software?*

In the law today relevant to privacy actions under the tort of invasion of privacy,[333] success depends on the facts of the litigation, the amount of damages, data mining methods, and nature of consent implied in the plaintiff. Litigants bring a cause of action under the tort of invasion of privacy, or as Restatement (Second) of Torts call it, intrusion upon seclusion.[334] The victim claims that the software perpetrator, by inserting the software without the victim's permission, "intrudes . . . upon the solitude or seclusion of another *or his private affairs or concerns*," and there will be liability "if the intrusion would be highly offensive to a reasonable person."[335] The authors of the restatement specifically envision intrusions that are not physical.[336] The restatement specifies that the intrusion "may be by some other form of investigation or examination into his private concerns, as by opening his private and personal mail, searching his safe or his wallet, examining his private bank account, ..."[337] The only concern may be to show that "the intrusion has gone beyond the limits of decency"[338] leading to liability on the part of the perpetrator. Victims, therefore, will be more likely to recover under an invasion of privacy theory if the Spyware steals personally identifiable information, such as private bank accounts, credit card numbers, and social security numbers.[339]

---

332. For example, users that shared a program Surfer Bar via e-mail distribution which embedded in the HTML formatted e-mail a hidden link to a site that dropped an executable into the C: drive, and then exploited a known vulnerability in Internet Explorer to automatically execute a Visual Basic script. Once installed, this application inserted multiple files on user systems and refreshed the system's registry keys, start-up page, and IE references every couple of seconds. The skill required by the end-user to remove this application extended beyond the average user's skill set. The application embedded many references to porno and gambling sites such that the user's browser was non-functioning. Surfer Bar was a form of adware, however, it could have just as easily been used to deliver a malicious Spyware application that stole and/or mined a user's machine.

333. *See e.g.*, Restatement (Second) of Torts § 652B et seq. (1977).

334. *Id.*

335. *Id.* (emphasis added).

336. *Id.* at cmt. b.

337. *Id.*

338. *Hamberger v. Eastman*, 206 A.2d 239, 242 (N.H. 1964) (quotation omitted) (citing to Restatement (Second) of Torts § 652B (comment d) concerning the need to show that the limits of decency have been exceeded).

339. *Id.*

Specifically, the courts have held a person is liable for invasion of privacy "if the intrusion would be highly offensive to a reasonable person"[340] and when that person "intentionally intrudes, physically *or otherwise*, upon the solitude or seclusion of another *or his private affairs or concerns.*"[341] Based upon the definition in the Restatement (Second) of Torts intrudes upon the private affairs or concerns of the owner of the computer even the comments of the Restatement make it clear that using electronic means is a method that fits within the definition.[342] The standard case of violation of an individual's rights of privacy leading to tort liability involves some sort of eavesdropping by one individual on another individual or groups of individuals.[343] From a digital standpoint, the closest analogies to such violations are those that involve electronic surveillance and eavesdropping cases.

A New Jersey court in a family law case had occasion to consider the tort of invasion of privacy with respect to computer records.[344] In that case, the husband had unwittingly saved e-mails from his girlfriend.[345] The husband had believed that only by using a password could the e-mail be found.[346] The computer, left in a location of the marital residence where anyone could access it,[347] gave the wife easy access to the information.[348] Even though the court held that the wife's rummaging through the e-mail files was no "different than rummaging through files in an unlocked file cabinet,"[349] the case is instructive concerning the tort of intrusion upon seclusion with respect to electronic records.

The court begins with the Restatement (Second) of Torts definition[350] and has no difficulty finding that accessing computer records fits within one of the comments, that is that the intrusion need not be physical.[351] However, the court has problems with whether the intrusion would be highly offensive "when the actor intrudes into an area in which the victim has either a limited or no expectation of privacy."[352] Consequently, because he left the computer in a room to which his wife had access, the court held he had *no* expectation of privacy as to the contents

340. Restatement (Second) of Torts § 652B.

341. *Id.* (emphasis added).

342. *Id.* at cmt. b (mentioning "with or without mechanical aides," "tapping his telephone wires," and, "examining his private bank account").

343. *See* e.g., *Hamberger*, 206 A.2d 239.

344. *White v. White*, 344 N.J.Super. 211, 781 A.2d 85 (N.J. Super. Ct. 2001).

345. *Id.* at 216.

346. *Id.* at 215.

347. *Id.* at 223.

348. *Id.*

349. *Id.* at 224.

350. Restatement (Second) of Torts § 652B.

351. *Id.* at comment (b).

352. *White*, 344 N.J. Super. at 222.

of the computer.[353] This case is troubling from the standpoint of the consumer who is attempting to acquire a remedy against the Spyware perpetrator. The court states, "a person's expectation of privacy to a room used for storage and to which others have keys and access is not reasonable. Defendant's subjective belief that the room was private is 'irrelevant.'"[354] The user of a computer may or may not know that being connected to the World Wide Web by telephone line or otherwise, gives another access to certain information in the computer. Analogizing from this case, Spyware perpetrators can argue that computer owners' beliefs that their information is private is irrelevant because the standard knowledge in the industry is that computers communicate with other computers online.

A more helpful case comes from the Supreme Court of New Hampshire.[355] The United States District Court for the District of New Hampshire certified certain questions of law to the New Hampshire Supreme Court. Essentially, the federal court wanted to know whether a private investigator who acquired private information (such as social a security number), and gave that information to another (the person hiring the private investigator) would be liable under the tort of intrusion upon seclusion.[356] Liam Youens contacted an Internet-based investigation and information service, known as Docusearch, to acquire information about Amy Lynn Boyer.[357] Docusearch sold Youens her social security number through a "pretext" telephone call and acquired Boyer's employment information.[358] "On October 15, 1999, Youens drove to Boyer's workplace and fatally shot her as she left work."[359] The court's analysis begins with a blanket statement that "all persons have a duty to exercise reasonable care not to subject others to an unreasonable risk of harm."[360] The court goes on to hold that "a party who realizes or should realize that his conduct has created a condition which involves an unreasonable risk of harm to another, has a duty to exercise reasonable care to prevent the risk from occurring."[361]

353. *Id.* at 223.

354. *Id.* (citing to *State v. Brown*, 660 A.2d 1221 (App. Div.). This is particularly troubling in light of the Supreme Court of Washington's holding in *State v. Townsend*, 57 P.3d 255 (Wash. 2002). That court held that a person using e-mail might be assumed to know that it is being recorded somewhere even if the person lacks actual knowledge of digital processes. *Id.* at 260. Therefore, people using e-mail are deemed to have consented to recording of messages). *Id.*

355. *Remsburg v. Docusearch, Inc.*, 816 A.2d 1001 (N.H. 2003).

356. *Id.* at 1004-1005.

357. *Id.* at 1005.

358. *Id.* at 1005-1006.

359. *Id.*

360. *Id.* (citation omitted).

361. *Id.* at 1007 (citation omitted).

With respect to the software perpetrator, this case seems to indicate that not only the perpetrator, but also the beneficiary of the software insertion, namely the merchant receiving the information or receiving the redirected web browser, may be liable. Under the reasoning in *Remsburg*, the software perpetrator and the merchant receiving the benefit of the invasion of a person's computer, not only must *realize* that the conduct creates a condition involving an unreasonable risk to a person's privacy, but actually *intend* to create that unreasonable risk. The court in *Remsburg* identifies two risks that are reasonably anticipated, and in the case of Spyware perpetrators, should lead the software vendor to anticipate the potential for identity theft.[362] The court concludes that "an investigator has a duty to exercise reasonable care in disclosing a third person's personal information to a client."[363] The investigator is to exercise reasonable care to ensure that nothing harmful comes from the release of information.

The court holds that whether an intrusion "would be offensive to persons of ordinary sensibilities is ordinarily a question for the fact finder and only becomes a question of law if reasonable persons can draw only one conclusion from the evidence."[364] The court refuses to hold that, as a matter of law, a person has a reasonable expectation of privacy in a social security number.[365] This, the second part of the tort of intrusion on seclusion, seems to be a question, like negligence, that normally must be determined by a jury. Although given the volume of personal information in one's computer, it seems likely that courts may hold that as a matter of law, the intrusion and interception of that information is offensive to persons of ordinary sensibilities and no reasonable juror could hold otherwise. At least that is the argument plaintiff's counsel should make in consumer civil liability. The court cites to a Minnesota federal court opinion that lists the following factors: the degree of intrusion; the context; the conduct; the circumstances surrounding the intrusion; as well as the intruder's motives and objectives; and, the expectation of privacy of the person invaded.[366] Consumers will argue that based upon the context, the degree of intrusion, and the intruder's profit motives, courts should hold as a matter of law that invasion of a computer by Spyware is objectively unreasonable and offensive.

The problem with tort liability, for either this tort or the tort of trespass to chattels, is damages and, from a practical standpoint, who will bring these claims? First, consumer's actual damages will at best be

362. *Id.* at 1007.
363. *Id.* at 1008.
364. *Id.*
365. *Id.* at 1008-1009.
366. *Id.* at 1009 (citing to *Bauer v. Ford Motor Credit Company*, 149 F. Supp. 2d 1106, 1109 (D. Minn. 2001).

minimal. Spyware may annoy consumers and even require them to spend hours trying to remove the offending applications, but few individuals will have sufficient damages to lead them to pursue a remedy. Moreover, with limited damages, attorneys will not institute litigation – the client will not pay an hourly rate; the attorney will starve on contingency arrangements unless the firm can identify enough victims for a class action.

2. *What are the best steps of analysis?*

The steps of analysis for a court to apply when hearing the tort of invasion of privacy suits will require the courts to examine the software architecture, code, and design that were involved in the development of the software. Below is an analysis of a court hearing arguments brought by its victims under the tort of invasion of privacy, or as Restatement (Second) of Torts calls it, "intrusion upon seclusion,"[367] against a software company that mined data using an advanced embedded web tool in a file-sharing program that mined their email and other personal documents.[368] The victim will claim that the Spyware perpetrator, by inserting the Spyware without the victim's permission, "intrudes, . . . , upon the solitude or seclusion of another *or his private affairs or concerns*," and there will be liability "if the intrusion would be highly offensive to a reasonable person."[369] The authors of the restatement specifically envision intrusions that are not physical.[370] The restatement specifies that the intrusion "may be by some other form of investigation or examination into his private concerns, as by opening his private and personal mail, searching his safe or his wallet, examining his private bank account . . . ."[371] The only concern may be to show that "the intrusion has gone beyond the limits of decency"[372] leading to liability on the part of the perpetrator. Spyware victims, therefore, will be more likely to recover under an invasion of privacy theory if the Spyware steals personally identifiable information, such as private bank accounts, credit card numbers, and social security numbers.[373]

When hearing such arguments, the courts face a new situation where software creators continually side-step laws meant to prevent their actions by breaking up software actions into multiple separate

367. Restatement (Second) of Torts § 652B (1977).

368. *See generally* Klang, *supra* n. 101.

369. *Id.* (emphasis added).

370. *Id.* at comment (b).

371. *Id.*

372. *Hamberger*, 206 A.2d, at 242 (quotation omitted). This court cites to Restatement (Second) of Torts § 652B (comment d) concerning the need to show that the limits of decency have been exceeded.

373. *Id.*

steps.[374] A prime example of the effects of technological "evolution" is the Wiretap Act, which has been rendered significantly less effective by both the legislature and the courts that have limited its provisions to protect only interceptions of communications in transit.[375] While this Wiretap Act construction adequately protects point-to-point telephone calls placed over the Public Switched Telephone Network ("PSTN"), it does not adequately protect electronic communications sent over the Internet, such as through Voice over Internet Protocol ("VoIP") which may be instantaneously stored on a server before, after, or during their transmission. These temporary stops along the way enable programmers to take advantage of the Wiretap Act by developing programs that copy or intercept the communications while they are in the temporarily stored state, and are not in transmission. The steps of analysis for a court to apply when hearing cases as to whether a software developer is liable for the invasion of privacy as a result of their software requires the courts to examine the development process, the architecture, the code, and the testing that was involved in the development of the software. It is evident that Spyware programs capitalize upon this loophole since the design of the software is such that it intercepts data prior to transmission, thereby avoiding potential liability under the Wiretap Act.[376] The analy-

374. *See e.g. Grokster*, 125 S.Ct. 2764 (holding that "[i]t is undisputed that StreamCast beamed onto the computer screens of users of Napster-compatible programs ads urging the adoption of its OpenNap program, which was designed, as its name implied, to invite the custom of patrons of Napster, then under attack in the courts for facilitating massive infringement.")

375. Intertwined with the intent and consent elements in interpreting the Wiretap Act is the storage – transit dichotomy. Circuits that narrowly read the Wiretap Act require the interception to be contemporaneous with transmission. *See In re Pharmatrak*, 329 F.3d, at 21. Under this standard it is possible for a defendant to argue that there are two separate communications: one between the end-user and the intended Web Portal, and the second between the end-user and the Spyware technology. *See generally Chance*, 165 F. Supp. 2d, at 1155-57; *DoubleClick*, 154 F. Supp. 2d, at 503-04; *In re Toys R Us*, 2001 U.S. Dist. Lexis 16947 at 3; *In re Intuit Privacy Litig.*, 138 F. Supp. 2d, at 1274. Under this argument, a Spyware program becomes a party to the conversation authorizing its interception of the data under the Wiretap Act. *See In re Pharmatrak*, 329 F.3d, at 19-22. Since the Wiretap Act allows either party to consent to the recording a data communications, the Spyware program is not violating the Wiretap Act. This is permissible because the Wiretap Act presupposes that both parties to the conversation had knowledge that a conversation was in fact taking place. 18 U.C.S.A. 2511(2)(d) (a party may consent to the interception of only part of a communication or to the interception of only a subset of its communication). Here, the end-user can assert that they lacked such knowledge and did not consent to the communication, but unfortunately the law has precluded the end-user from asserting that the transmission occurred without their consent. *Id.*

376. *See e.g. U.S. v. Ropp*, 347 F. Supp. 2d 831 (2004) (finding that because the captured keystrokes were not transmitted by a system that affects interstate commerce, spying with the device did not violate the federal act because it did not intercept the communication while it was being transmitted.)

sis that a court would need to apply requires that a court examine the high-level software design and architecture focusing on whether the software avoids Wiretap Act liability by mining end-user data and electronic communications while it resides on the end-user's system prior to actual transmission. Unless, a court performs this analysis it is likely that such software vendors will continue to be able to capitalize on this legal loophole because the software creators are circumventing the law by "coding" around it. On the other hand, even if a court analyzes the software if the court declines to construe the laws in a broader sense, not a great deal can be done and such software will continue to operate.