ENGINEERING DATA RIGHTS

Peter M. Watt-Morse, Esq.*

INTRODUCTION

Just as a programmer applies software engineering concepts to create high quality computer code, a software owner must "engineer" data rights to create optimum legal protections for such code. To engineer data rights, a software owner must: (i) evaluate the software's intended uses and users; (ii) select the appropriate data rights considering the inherent advantages and constraints of the available legal "tools" (e.g., copyrights, trade secrets, patents or trademarks), and (iii) take the necessary steps to develop the selected data rights. If engineered correctly, the resulting data rights will effectively use available legal tools, be fully compatible with the software's intended uses and provide the owner with practical legal safeguards to protect the economic value of the software.

This chapter is intended to introduce software engineers to data rights available in the United States and the considerations in choosing the appropriate data rights. Although a treatise on intellectual property law is beyond the scope of an introductory chapter, a clear understanding of the basic legal rules for each type of data right is essential (Nimmer, 2000; Milgrim, 2000; Chisum, 2000; McCarthy, 2000). Therefore, this chapter will review the legal framework for the following data rights: copyrights, trade secrets, patents, trademarks and mask works. With this framework in place, the chapter will examine the considerations in selecting particular data rights. Finally, the chapter will highlight data right issues that software engineers encounter on a regular basis (e.g. extent of copyright protection, employee ownership, governmental data rights and Internet issues).

Before reviewing basic legal principles, it is important to debunk two common myths regarding software data rights: (i) copyright is the only data right available for software; and (ii) data rights are best

---

* The author is a partner with the law firm of Morgan, Lewis & Bockius LLP in Pittsburgh, Pennsylvania. The author wishes to thank the following colleagues for their helpful comments regarding initial and/or drafts of this manuscript: Jeffrey Glickman, Jeffrey Godish, Edward Pencoske, Esq., Robert Stephenson, Esq. and Richard James, Esq.. The author specifically wishes to thank Gary Regan for his work on the revised draft of this manuscript.

left to lawyers. First, just as C is not the only available programming language, copyright is not the only available data right. Depending on the intended uses of a software program, alternative data rights, such as patents, trade secrets and trademarks, may provide more effective legal protection (Epstein, 2000). Second, attorneys cannot develop intellectual property rights on their own. There are steps that software engineers must complete to create and maintain data rights in their programs. Unless such steps are taken, the data rights (and the economic value) of the software may be lost, Therefore, it is important for software engineers to have a basic understanding of software data rights.

A review of the history of the familiar 1-2-3 spreadsheet program distributed by Lotus Development Corporation illustrates the importance of understanding data rights (Lotus v. Paperback, 1990; Lotus v. Borland, 1995; Bartolik, 1992). The basic idea of a software program that creates an electronic spreadsheet on a personal computer did not originate with Lotus. A student at Harvard Business School, Daniel Bricklin, developed the idea in the late 1970's and created the first commercial spreadsheet software, VisiCalc, a program for the original Apple II computer. As author of the computer program, Bricklin retained copyright rights in VisiCalc and could prevent others from copying his program. However, as the inventor of the idea of an electronic spreadsheet, Bricklin also had the right to protect the program with trade secret or patent law. Bricklin did neither. Therefore, while VisiCalc could not be copied, Bricklin's idea for an electronic spreadsheet had limited legal protection (as will be noted below, patent protection for software was difficult to obtain at that time).

Into this opening stepped Mitchell Kapor and Jonathon Sachs of Lotus. They created 1-2-3, an improved spreadsheet program that ran on the new (in 1982) IBM PC. Lotus obtained the commercial success that alluded Bricklin and 1-2-3 became the dominant spreadsheet program for personal computers at that time. Moreover, just as Kapor and Sachs originally exploited a hole in VisiCalc's data rights to create their program, Lotus was able to maintain its dominant position throughout the 1980's in part by carefully exploiting its own data rights. First, Lotus developed creative and original ways of expressing the functions inherent in an electronic spreadsheet. Initially, these creative expressions permitted Lotus to develop a spreadsheet program without copying VisiCalc program and infringing Bricklin's copyright rights. Later, these creative expressions provided Lotus with valuable copyright rights which the company used to stop the marketing of competing spreadsheets (Lotus v. Paperback, 1990; Schwartz, 1990).

Moreover, copyright is not the only data right that protected 1-2-3's position. By the last half of the 1980's, the trade name LOTUS and the trademark 1-2-3 were equally important rights. Like McDonald's and fast food, LOTUS and 1-2-3 became synonymous with electronic spreadsheets. While competitors could develop and market alternative spreadsheet programs, they could not use "LOTUS" or "1-2-3" to identify their software packages. The value of these additional rights cannot be overestimated (McCarthy, 2000; Raysman, 1992). In fact, during the 1990's, Microsoft used its Microsoft and EXCEL

marks and strategic licensing of its copyright rights to replace 1-2-3 as the dominant spreadsheet program for PCs.

COPYRIGHT

Introduction. Copyright is the most common type of data right protection for software. In the United States, the power to grant copyrights was given to Congress in the original words of the Constitution:

> "Congress shall have the power ... to promote the progress of science and useful arts by securing for a limited time to authors ... the exclusive right to their respective writings." (U.S. Const. Art. I,' 8)

Under this Constitutional grant, Congress has adopted statutes governing the United States copyright system, including the Copyright Act of 1976 ("Copyright Act") (Copyright Act, 1976; 17 U.S.C. §§ 101-810; Copyright Office, 1977). The Copyright Act was effective on January 1, 1978 and preempted all state law and common law rules that previously applied to copyright.

Prior to 1978, it was generally recognized that computer code was a "writing" that was protected by copyright. The U.S. Copyright Office began accepting federal copyright registrations for software in 1964. The Copyright Act codified this practice. The Copyright Act's legislative history specifically recognized that computer programs were literary works that were subject to copyright protection (H. R. Report No. 1476, 1976; Nimmer, 2000). Moreover, the Computer Software Copyright Act of 1980 specifically amended the Copyright Act to include a statutory definition of "computer program" ("a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result") and to provide a statutory guide to permissible (archival and RAM) and impermissible (all other) copying of computer programs (Computer Act, 1980; 17 U.S.C. §101, 117; Scott, 1989). Finally, the recently adopted Digital Millennium Copyright Act of 1998 recognized the use of copyright management tools typically contained in software to protect the rights of copyright holders (17 U.S.C. §1201).

The Copyright Act protects a broad range of rights embodied in software. Most basically, copyright protects against the direct copying of the written expressions of a computer program, including the source code listing and the text produced by the program. In addition, textual expressions related to the program, such as specifications, user manuals and packaging information, are protectable by copyright. Moreover, non-textual embodiments of software are also protectable by copyright. In a manner similar to the protection of a movie recorded on a VCR tape, copyright protects code stored electronically and protects screen displays produced by the program, including images and other graphic expressions in those screen displays (17 U.S.C. §102; Apple v. Franklin, 1983; Midway v. Strohon, 1983; Baldy, 1984).

Although copyright protects a broad range of rights in software, the protection is limited. Section 102 of the Copyright Act states that copyright protection does not extend to any idea, concept, principle or

discovery. A software engineer's work papers provide an excellent example of this idea/expression dichotomy. Copyright protects the expression contained in the work papers from direct copying - i.e. copyright prevents copying a flow chart of a program module contained in the papers. On the other hand, copyright does not prevent other software engineers from using the ideas embodied in the papers in their own independent computer programs (CAI v. Altai, 1992; Apple v. Microsoft 1994).

One way to understand the idea/expression dichotomy is to review the balance established by the Constitution for copyright protection. Congress was directed ***to promote science*** by securing for a ***limited*** time to authors the exclusive right of their respective ***writings***. In fulfilling the Constitutional purpose of promoting science, the Copyright Act attempts to strike a balance between (i) granting limited ownership rights in writings to encourage the production of such writings, and (ii) preserving the public's right to free access to ideas and information as reflected in the First Amendment (Nimmer, 2000).

Rights. Section 106 of the Copyright Act grants to a copyright owner the exclusive rights to: (i) reproduction and public display, (ii) distribution of copies, and (iii) preparation of derivative works.

At its most basic level, the exclusive right to reproduction prevents third parties from copying written documents such as user manuals or reproducing recorded materials such as computer tapes (Synercom v. University, 1978). The restriction on copying also prevents transferring software from one medium to another such as copying a program stored on a portable disk to a hard drive (Apple v. Formula, 1984;). Moreover, copyright not only prevents exact reproduction, but it also prevents third parties with access to the original work from creating a new. expression that is substantially similar to the original. For example, a programmer with access to the source code listing of a copyrighted program could not escape liability for infringement by reproducing the software program with minor modifications to the code.

Defining what constitutes a "substantially similar" computer program is a complex question. The extremes are clear. On the one hand, a computer program that contains a substantial number .of identical lines. of computer code generally will constitute a copyright violation (Central Point Software v. Nugent, (1995); Midway v. Strohan, 1983; Atari v. Philips, 1982). On the .other hand, a computer program that contains no identical lines of code will not violate copyright rights even if it embodies many of the same ideas as the original work (CAI v. Altai, 1992; Apple v. Microsoft, 1994; Nimmer, 2000). Determining substantial similarity between these extremes can only be done on a case by case basis.

The exclusive right to reproduction is limited by the doctrine of "fair use." Section 107 of the Copyright Act provides that "the fair use of a copyrighted work ... for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship or research, is not an infringement of copyright." Although defining the precise limits of the "fair use" exception is complex, generally the exception is relatively narrow and permits limited reproduction solely for non-commercial purposes (17 U.S.C. §107; Sony v. Universal City Studios, 1984; Nimmer, 2000).

A copyright holder's exclusive right to distribution expands the prohibition on third party reproduction. For example, even if a software owner grants an end-user the right to copy a program for its own internal use, the end-user may not distribute such copies to third parties without the express permission of the copyright owner. It should be noted that exclusive distribution rights are limited by the "first sale doctrine." Under this doctrine, once a copyright owner sells a copy of his work to another party, the owner cannot prevent the purchaser from reselling that particular copy (17 U.S.C. § 109; Williams v. Arctic, 1982). This encyclopedia provides a classic example of the first sale doctrine. The ban on distribution prevents a purchaser of the encyclopedia from copying portions of the text and distributing the copies to others. On the other hand, nothing in the Copyright Act prevents the purchaser from reselling the original encyclopedia to a third party.

The exclusive right to prepare derivative works is equally important. The Copyright Act states that a derivative work:

> "... is a work based upon one or more preexisting works such as a translation ... abridgement, condensation, or any other form in which a work may be recast, transformed or adapted." .(17 U.S.C. §§101, 106).

In other words, the Copyright Act prevents development of software that includes portions of the original program or is merely a translation or adaptation of the original work. Therefore; courts have held that converting a computer program from one language to another constitutes a "derivative work" and is prohibited by the Copyright Act (Whelan v. Jaslow, 1986). On the other hand, the prohibition against derivative works does not prevent creation of independent works that use output data created by the original program (Synercom v. University, 1978).

Applying these concepts to an electronic spreadsheet program, the Copyright Act prevents third parties from taking a portion of the code of the EXCEL or 1-2-3 program to derive a competing spreadsheet program. In addition, competitors are prevented from creating a new program by compiling or otherwise translating EXCEL and 1-2-3 into alternative computer languages. However, the Copyright Act does not prevent competing spreadsheet programs from capturing the ideas underlying those spreadsheet programs and using those ideas in alternative spreadsheet programs.

The exclusive rights granted by the Copyright Act control the use of copyrighted software where no agreement exists between the copyright owner and the software user. On the other hand, owners and users frequently wish to modify these rights and obligations. For example, owners may wish to allow distributors to make and transfer copies of the original program and users may wish to modify the program for their own internal purposes. Based on these desires, software owners and users frequently enter into license agreements that specify the rights and obligations regarding use of the licensed software. Although the negotiation, content and enforcement of license agreements is beyond the scope of this chapter, a brief review of certain key provisions of software license agreements highlights the

scope and limitations of the protections provided by the Copyright Act (more complete treatments of licensing agreements are contained in Roditti, 2000 and Epstein, 2000).

A software -license agreement generally delineates the actual materials that will be provided to the licensee by the software owner including object code and/or source code and related documentation. The agreement also specifies restrictions on use of the software including limitations regarding copying and modifying the program, operating the program on particular hardware and transferring the program to third parties. If the license agreement involves distribution, the agreement will specify the terms of any sublicense agreements between the distributor and its customers along with the responsibilities of the original software owner and the distributor to such customers (Roditti, 2000; Gordon, 1986). If the license agreement permits modification of the original program, ownership rights will be established for the modifications. Finally, the license agreement will establish restrictions and obligations on the licensee to protect the data rights of the original program, including disclosure restrictions and obligations to maintain copyright notices and other proprietary legends (Epstein, 2000; Raysman, 2000).

Creation. Copyright data rights are easy to create. Copyright protection begins at the moment a software programmer writes a line of computer code on paper or records a line of code electronically. Moreover, copyright protection is relatively long lasting. For individual authors, copyright lasts for the life of the author plus 70 years. If the author is anonymous or the copyright is owned by a company under the work-for-hire doctrine (see further discussion below), the copyright lasts for the lesser of 95 years from the date the work is published or 120 years from the date the work is created. Finally, if the work is a joint work of two or more individuals, the copyright lasts for 70 years after the death of the last surviving author (17 U.S.C. §302).

A software owner is not required to file a copyright registration to obtain federal copyright protection. Copyright protection automatically starts when the expression is "fixed" in a tangible medium. Under the Copyright Act, an original work is "fixed" in a tangible medium of expression when it is sufficiently permanent to be perceived, reproduced or otherwise communicated for a period of more than transitory duration (17 U.S.C. §§101, 102; H.R. Report No. 1476). Applying this standard to software, a line of code is not "fixed" when a software programmer orally describes that line of code to another person. On the other hand, as soon as the code is written on paper or is stored on a hard drive, the work is "fixed" and copyright protection has begun (Midway v. Strohan, 1983; Williams v. Arctic, 1982).

Although federal registration is not required for copyright protection, registration offers many legal advantages. Most basically, a U.S. citizen cannot bring an action for copyright infringement in federal court unless the copyrighted work is registered (17 U.S.C. §411). Moreover, if registration is filed within three months of first publication, the copyright owner may receive statutory damages and attorneys fees for infringement that occurs after registration (17 U.S.C. §412). Finally, if a work is registered within five years of publication, federal courts presume the validity of the copyright and the defendant has the burden of proving any copyright deficiencies (17 U.S.C. §410).

In addition, copyright registration has practical benefits. Registration indicates to suspected infringers that the copyright owner is serious about enforcing its copyright rights. A cease and desist letter with an attached registration certificate can be an effective first step in stopping unauthorized copying. Moreover, the U.S. Customs Office will impound suspected copyright infringements only if the original work is registered (17 U.S.C. §§602, 603; 19 C.F.R. §133). This customs protection is particularly important because of the failure of other countries to enforce copyright protections for software as vigorously as the United States. Therefore, although copyright protection may technically exist in countries in the Far East and South America, often the best practical protection against software piracy in these countries is the U.S. Customs Office.

A final reason to register a copyrighted work is that registration is simple and inexpensive. A copyright registration application generally includes (i) a two page form, (ii) a deposit of the copyrighted work, and (iii) a small registration fee (currently $30). A complete description of copyright registration procedures is beyond the scope of this chapter and software engineers are urged to visit the U.S. Copyright Office web site (http://lcweb.loc.gov/copvrght/) or confer with a copyright specialist regarding particular questions (Nimmer, 2000; Register of Copyrights, 2000). However, for purposes of introduction, the following is a description of the registration procedures for a typical software program.

Generally, a software owner will use U.S. Copyright Form TX, the registration form for literary works. This form requires the author's name, the title of the work, a general description of the work, the date that it was created and the date, if any, that it was published. In addition to the registration form and filing fee, the software owner must deposit a copy of the software program with the Library of Congress. Under Copyright Office regulations, the software owner is required to file either the source code or object code version of the program (37 CFR §202; Register of Copyrights, 2000). To protect trade secrets that may be embodied within the program, the author may file representative portions of the program. Within approximately six months of receipt, the Copyright Office will review the submission and return a registration certificate if the submission is in proper form.

In addition to registration, additional copyright protection can be created through use of the familiar copyright notice (there is no requirement that a copyright registration be filed in order to use a copyright notice). For works published after March 1, 1989, the effective date of the Berne Implementation Act of 1988 ("Berne Amendments"), there is no requirement that works contain a copyright notice in order to retain copyright protection (Berne Act, 1988; H.R. Report No. 609). This substantial change in U.S. copyright law was necessary to conform U.S. law to the Berne Convention which establishes standards for international copyright protection. Prior to 1978, if a work was published without copyright notice, copyright protection was irretrievably lost (Data Cash v. JS&A, 1980). This harsh standard was relaxed from 1978-1989 during which time lack of notice did not automatically terminate copyright protection but there were certain obligations to use reasonable efforts to provide such

notice (17 U.S.C. §401; Midway v. Arctic, 1983). With the advent of the Berne Amendments, even these obligations have. been removed.

However, like registration, there are both legal and practical reasons to use copyright notice. Under the Berne Amendments, copyright notice provides a statutory presumption against the defense of innocent infringement (17 U.S.C. §401(d); H.R. Report No. 609). In other words, courts will presume that a defendant who improperly copied a software program despite the presence of a copyright notice did so intentionally and is subject to damages for its actions (generally an innocent infringer can be enjoined but is not subject to compensatory damages for infringement prior to the initiation of the lawsuit). Practically, copyright notice indicates to third parties that the copyright owner is serious about enforcing its copyright rights and encourages third parties to self-police their use of a copyrighted work.

Adding a copyright notice is a simple procedure. A copyright notice should be placed on every copyrighted work when it is published - i.e., when copies are distributed to third parties. Unpublished works, such as unfinished code and engineering notes, do not need a copyright notice until they are distributed to third parties. A copyright notice contains three parts: the familiar ©, the date of first publication and the name of the copyright owner. For example, the copyright notice for the initial version of 1-2-3 would read: "© 1981, Lotus Development Corporation."

Three things should be noted about copyright notice. First, always use the symbol ©. Although U.S. law permits the words "Copyright" and "Copr." to be used in place of ©, these substitutes are not recognized internationally (Nimmer, 2000). Moreover, at least one U.S. court has held that a notice containing "(c)" rather than © could be ineffective (Videotronics v. Bend, 1984). Second, each time a new release of a software program is developed, a new copyright notice should be created to reflect the date of first publication of the new release. Third, to properly notify third parties of copyright claims, copyright notice should be placed such that users of the copyrighted work will generally encounter the notice. For software, the U.S. Copyright Office has indicated that notice should be placed on disks containing the object code, on initial screen displays and on all source code listings (17 U.S.C. §401(c); 37 C.F.R. §201.20).

TRADE SECRETS

Introduction, Although copyright protection is the data right currently most closely associated with software, for many years the primary software data right was trade secret protection (CONTU, 1978). Prior to the advent of personal computers and workstations, large centralized computer systems relied on proprietary operating software developed by hardware manufacturers and customized applications software developed for particular end-users. Computers were expensive and could only be used by corporations and individuals with the resources to obtain this high-powered competitive advantage. Not surprisingly, computer manufacturers and users attempted to maintain this competitive advantage by maintaining the confidentiality of their software. With the widespread use of personal computers in the 1980's, it became less practical to use trade secret protection as the primary data right for software.

However, software owners still rely on trade secret law for certain protections and it is important to understand the advantages and limitations inherent in this data right (Jager, 2000; Milgrim, 2000).

Unlike copyrights and patents, there is limited federal protection of trade secrets. The Economic Espionage Act of 1996 criminalized misappropriation of trade secrets, but provides no civil remedy for a violation (18 U.S.C. § 1832). Trade secret protection is generally governed by state law, and originated from the common law of contracts and unfair competition (Restatement, 1939; Kewanee Oil v. Bicron, 1974). Under the contract branch of trade secret law, violation of a trade secret involves the breach of a confidentiality agreement or relationship, such as an engineer taking research plans from a former employer. Under the unfair competition branch of trade secret law, violation of a competitor's trade secret involves obtaining valuable confidential information through burglary or other improper means.

Each state has its own trade secret law. The most common statement of American trade secret is contained in the Uniform Trade Secrets Act ("UTSA") which has been adopted in more than thirty states including such important states as California, Florida and Illinois. The UTSA defines a trade secret in Section 1:

> "Trade secret" means information, including a formula, pattern, compilation, program, device, method, technique or process that:
>
> (i) derives independent economic value, actual or potential, from not being generally known to, and not being readily ascertainable by proper means by, other persons who. can obtain economic value from its disclosure or use; and,
>
> (ii) is the subject of efforts that are reasonable under the circumstances to maintain its secrecy." (UTSA, 2000; Restatement, 1939).

What type of software meets this definition? Commercial computer programs such as EXCEL or 1-2-3 are generally known and cannot meet these standards. On the other hand, programs with limited distribution may suitable for trade secret protection if such software is not generally known to competitors and provides users with an economic advantage by remaining secret (MSA v. Cyborg, 1984). As long as the software is distributed to a limited number of users, such users are subject to confidentiality restrictions and the owner takes other reasonable secrecy precautions, the software will qualify as a trade secret under the UTSA definition.

The development of a specialized spreadsheet for calculating engineering costs illustrates this distinction. Neither the basic EXCEL or 1-2-3 program nor the idea of using an electronic spreadsheet for calculating engineering costs can be protected as a trade secret. On the other hand, if a developer creates a sophisticated spreadsheet application that contains algorithms and/or data that is not generally known in the industry and provides an economic advantage over competitors, the particular spreadsheet analysis could be protected as a trade secret.

Moreover, even if a software program is subject to widespread distribution, there may still be portions of the program that can be protected. Most importantly, if the source code is not readily ascertainable from the object code that is distributed, the source code can be protected as a trade secret (Telex v. IBM, 1975). Based on current technology, courts have held that, practically, source code cannot be derived from object code and therefore widespread distribution of object code of a software program does not destroy the confidentiality of the source code for such program (Data General v. Grumman 1993; Q-Co. v. Hoffman, 1985; Grogen, 1984). It should be noted that when it becomes technologically practical to convert object to source, trade secret protection for source code may effectively end for programs for which the object codes are widely distributed. At that point, owners of widely distributed software would be dependent on copyright and patent protection to protect their data rights in source code.

In addition to source code, other information associated with software can be protected by trade secret law, including research and development materials, know-how regarding manufacturing and maintenance, and business information such as financial data, customer lists and business plans (Restatement, 1939; Milgrim, 2000). It is important for engineers to recognize that such information has potential economic value that can be retained only if the information is kept secret. As described above, if the information becomes generally known to competitors or reasonable efforts are not instituted to maintain such secrecy, trade secret protection will be lost.

Rights. Information that meets trade secret definition is protected against "misappropriation" by third parties. As described above, misappropriation includes use or disclosure of a trade secret through breach of a confidential relationship or other improper means (UTSA, 2000; Kewanee Oil v. Bicron, 1974). The most common breach of a confidential relationship is disclosure of trade secrets by former employees. Even without written employment agreements, employees have an implied duty not to use or disclose trade secrets owned by their employers (NCH v. Broyles, 1985; Milgrim, 2000). This duty extends past the date at termination of employment and covers specific confidential information owned by the employer. However, the duty does not prevent an employee from retaining the general knowledge and skills acquired while working for the employer. Drawing the line between specific trade secrets and general knowledge and skills is difficult and can only be determined on a case-by-case basis. This chapter will review this issue more closely in the discussion of employee data rights.

In addition to an employee's implied duty, the owner of a trade secret can establish a confidential relationship through a written agreement (Roditti, 2000; Epstein, 2000). To be effective, a confidentiality agreement should identify the specific information that is to be protected or provide that the owner will identify such information in the future. Moreover, the agreement should prevent both the disclosure ***and the use*** of the confidential information without the prior express consent of the trade secret owner.

The second branch of misappropriation ("other improper means") is more difficult to establish. Trade secret law does not prevent competitors from using legal means to discover a trade secret (CVD v.

Raytheon, 1985; Jager, 2000). For example, trade secret law does not prevent a competitor of Microsoft from buying EXCEL and studying the program and user manual to understand and use the ideas embodied therein. On the other hand, if a competitor broke into the headquarters of Microsoft in Redmond, Washington and stole the source code for EXCEL, in addition to other criminal and civil penalties, the competitor could be enjoined from using the source code and held liable for trade secret damages resulting from such use (Kewanee Oil v. Bicron, 1974; E.I. Dupont v. Christopher, 1970).

Trade secret protection is both broader and narrower than copyright protection. Unlike copyright, trade secret protection is not limited to substantially similar copies. For example, a competitor that misappropriated source code could still be liable for trade secret infringement even if the software program it developed from such source code contained sufficient originality to avoid copyright infringement (Motorola v. Computer Displays, 1984; Cybertek v. Whitfield, 1977). Broadly, if the information meets the requirement for trade secret protection, the law will protect both the expression of such information ***and*** the ideas and information embodied therein.

Not only is the scope of trade secret protection broader, but the length of such protection is potentially longer. Unlike copyrights or patents, trade secret protection can theoretically last forever. By definition, a trade secret lasts until the information either becomes general knowledge in the relevant industry or the trade secret owner fails to take the necessary steps to protect its secrecy (CONTU, 1978; Milgrim, 2000). Thus, trade secrets may be protected for decades. For example, Coca-Cola has preserved its trade secret data rights in the formulation of Coca-Cola Classic for nearly a century.

Despite these advantages, in other respects, trade secret protection is more limited than copyright protection. While copyrights extend to all software, trade secrets are restricted to software that meets the UTSA or other applicable definition for economic value and secrecy. Moreover, even if a trade secret owner can establish compliance with the applicable definition, the protection applies only to "misappropriations" and does not apply to all reproduction, distribution and derivative works (Restatement, 1939; Wexler v. Greenburg, 1960).

Both trade secrets and copyrights fail to protect against independent creation of the same proprietary technology (Kewanee Oil v. Bicron, 1974; Novelty v. Joan Fabrics, 1977). For example, if a competitor can prove that it did not copy a software program and did not have access to trade secrets related to the program, even if the resulting program contains similar expression, ideas or information, the competitor may be able to prove independent creation and avoid liability for copyright or trade secret infringement. Therefore, to avoid claims of copyright or trade secret misappropriation, many software developers will take extensive precautions to isolate their programmers from "contamination" and document the fact that the software developers did not have access to copies of competing products or confidential information regarding such products (NEC v. Intel, 1988; Milgrim, 2000).

Creation. Unlike a patent where protection is dependent on approval by federal authorities, creation of a trade secret is entirely dependent on the independent actions of the trade secret owner. To

prove economic value of its confidential information, an owner should document the time and expense required to. create the confidential information and the competitive advantage that such information provides. For example, the owner of a confidential software program can initiate trade secret protection by documenting the programming time and expenses incurred in the development process and the economic advantages provided by the program, such as improved data accuracy and speed (Electro-Miniatures v. Wendon, 1985; University Computing v. Lykes-Youngstown, 1974). In addition, to demonstrate the economic value of its confidential information, the trade secret owner must prove that the information is not generally known to other members of its industry. For example, at the time of development of a custom software program, the owner could conduct a search of available commercial software products that demonstrates that the features of the custom software are not generally available (Jostens v. National Computer, 1982; Cybertek v. Whitfield, 1977).

In addition, to create trade secret data rights, a software owner must maintain a strong corporate secrecy program (In Re: Innovative, 1986; Gordon, 1985). A corporate secrecy program serves two purposes. Most fundamentally, a secrecy program is legally required to meet the UTSA trade secret definition (UTSA, 2000; Novell v. Timpanagos, 1998). In addition, the expense and effort involved in protecting a trade secret serves as evidence that the trade secret has economic value and that it was not widely known within the industry (Metallurgical v. Fourtek, 1986). The first step in creating a secrecy program is identification of information that is protectable as a trade secret. A secrecy program that attempts to protect all information will be overly expensive, will not protect information that is publicly known, and, most importantly, will dilute the effectiveness of the program for information that truly is confidential and economically valuable.

Once protectable information is identified, it should be labeled with a trade secret legend (e.g., "This document contains confidential information which is owned by XYZ Corporation and may not be copied, disclosed, or otherwise used without the express written consent of XYZ Corporation") (Jostens v. National Computer, 1982; Com-Share v. Computer Complex, 1971). For software that is protected as a trade secret, legends should be placed on all disks and packaging and embedded within both the object code and source code of the software so that they appear on the initial lines of the program displays and source code listings.

In addition to identifying and placing legends on confidential information, a corporate secrecy program must contain procedures to limit access to the information solely to those persons who are authorized to use the information. Strict control of all copies of the confidential information, restricted access areas, secured networks and computers and destruction of all unused materials are part of a typical trade secret protection program (Cybertek v. Whitfeld, 1977; Epstein, 2000; Milgrim, 2000). In addition, all disclosures of the confidential information to third parties should be governed by written agreements, including confidentiality agreements for consultants and non-disclosure provisions in license agreements for end-users.

In addition to such preventive measures, the trade secret owner must vigorously prosecute any misappropriation of its trade secrets (Epstein, 2000, Milgrim, 2000). For example, if a software owner has knowledge that its source code is being used by third parties and takes no legal steps to stop this use, the source code may cease to be protected as a trade secret. Just as the costs incurred in prosecuting misappropriations can be used as evidence of the economic value of a trade secret, the fact that a trade secret owner does not incur these costs is evidence that the confidential information does not have the economic value necessary to be a trade secret.

PATENTS

Introduction. While copyright and trade secret rights may be the most basic software data rights, patent rights are perhaps the most powerful (Bender, 2000; Chisum, 2000). In addition to protecting the expressions of a software program, patent law protects the ideas or processes embodied in the program. Unlike copyrights or trade secrets, patents protect against independent development of similar software or ideas. Moreover, recent developments at the US Patent and Trademark Office ("PTO") and in the courts have significantly encouraged the use of patents for software.

A software owner cannot independently create patent rights. Patent protection in the United States begins only after the PTO has reviewed a detailed patent application and determines that a software program contains an invention worthy of protection (Chisum, 2000; Lipscomb, 2000). In the United States, patent law is based on the constitutional directive to Congress "to promote the progress of science and useful arts by securing for a ***limited*** time to ... ***inventors*** the exclusive right to their respective ... ***discoveries***." (U.S. Const. Art. 1, §8). Pursuant to this mandate, Congress has enacted several patent statutes, the most recent being the Patent Act of 1952 ("Patent Act"). The Patent Act grants exclusive ownership rights for at least a 20-year period, from the time the patent application is filed, for inventions that meet statutory standards for utility, novelty and nonobviousness (Patent Act, 1952; 35 U.S.C. §§101, 102, 103 and 154; U.S.P.T.O., 1992).

The Patent Act attempts to balance a number of competing interests that are reflected in the constitutional language: (i) rewarding inventors for the time and expense incurred developing new processes; (ii) encouraging disclosure of such inventions to promote the further development of science; and (iii) limiting the exclusive rights granted to a patent owner to foster free and open competition (Kewanee Oil v. Bicron, 1974; Chisum, 2000). The Patent Act strikes this balance by imposing the following conditions on the patent applicant: the inventor must disclose his/her invention in a detailed patent application; upon granting a patent (or, in some instances, eighteen (18) months after the patent application is filed), the PTO will disclose the application to the public; and, at the end of 20 years, the invention will become available for general use.

Section 101 of the Patent Act defines patentable subject matter to include "any new and useful process, machine, manufacturer or composition of matter." The Supreme Court has broadly interpreted this definition to include "anything under the sun that is made by man" (Diamond v. Chakrabarty, 1981).

On the other hand, the Court has repeatedly held that laws of nature, abstract ideas and mathematical formulas are not patentable subject matter (Mackay v. RCA, 1939). The Court's reluctance to recognize algorithms and natural laws as patentable subject matter stems from a concern that if inventors could obtain exclusive rights in such items, they could effectively prevent other scientists from using such algorithms and 'natural laws to advance science or develop other processes, machines and compositions.

In the field of software patents, the PTO and the courts had a difficult time reconciling the broad language of the statute and the exception for abstract ideas. Therefore, for many years, the PTO refused to grant patents on software, fording them to be a collection of abstract algorithms or mental steps. However, after years of creative efforts by patent counsel, software is now recognized as patentable subject matter. A review of this tangled history is useful to understand the boundaries of patent protection.

Under guidelines published in 1968, the PTO took the position that software was merely a collection of mathematical algorithms and therefore could not be patented (U.S.P.T.O., 1968). As permitted in the Patent Act, software owners appealed denial of their patent applications to the federal courts. Originally, the Supreme Court sided with the PTO (Gottschalk, 1972; Parker, 1978). In Gottschalk v. Benson, the U.S. Supreme Court held that a software program to convert binary coded numbers into true binary signals was merely an unpatentable mathematical method. Six years later, in Parker v. Flook, the Court again upheld the denial of a software patent application, this time for a program that calculated the alarm limits for a catalytic conversion process. However, the language in the Parker case provided insight regarding the type of software patent application the Court would approve (Blumenthal, 1980). Based on this language, beginning in 1978 the Court of Customs and Patent Appeals ("C.C.P.A.") (now the Court of Appeals for the Federal Circuit) held that certain software was patentable. For example, software that was part of a new hardware/software combination was held to be a patentable "machine" (In Re: Bradley, 1979). Moreover, the C.C.P.A held that software could be part of a patented "process" if it was integrated with an industrial process that itself was patentable (In Re: Johnson, 1978).

Despite these C.C.P.A. decisions, the PTO continued to deny all software patent applications. This changed in 1981 when the Supreme Court held for the first time that a software program was patentable (Diamond v. Diehr, 1981; Nimtz, 1981). The software program in Diehr regulated a catalytic process by recording measurements and adjusting curing times and other parameters based on such measurements. The Court noted that although the software included mathematical algorithms, the software was merely part of an application that used such algorithms to control a catalytic process.

Based on Diehr, the PTO and the federal courts developed a two-part test to determine whether software is patentable under Section 101: (i) is there a mathematical algorithm claimed in the patent application, and (ii) considering the patent application as a whole, is the mathematical algorithm claimed as the "invention" or is it merely applied as part of a patentable process? (U.S.P.T.O., 1992; In Re: Abele,

1982; In Re: Walter, 1980; In Re: Freeman, 1978). In determining whether a claimed mathematical algorithm is merely part of a patentable process under the Freeman-Walter-Abele test, the PTO considered a number of factors including the extent of the process after completion of the mathematical algorithm (post-solution activity), limitations in the patent to particular industrial processes, and actual transformation of physical materials by the overall process (Barrett, 1989).

While it was possible to obtain patent protection for software, the application by the PTO of the Freeman-Walter-Abele test limited the protection to certain types of software. However, in 1994, the Federal Circuit announced a much broader test to determine software patentability, focusing on whether the software was linked to a practical application (In Re: Alappat, 1994). Alappat claimed software that created a smooth waveform display for a digital oscilloscope. The court held that software programs and mathematical algorithms were not distinct categories of unpatentable subject matter, and held that only if the subject matter as a whole represented a "disembodied mathematical concept" or "law of nature" would the software be denied as unpatentable subject matter. The Court upheld Alappat's claim, holding that software to produce a useful, concrete and tangible result could not be characterized as an abstract idea.

In 1998, the Federal Circuit confirmed that the mathematical algorithm exception was invalid and that the Freeman-Walter-Abele test had little, if any, applicability to determining statutory subject matter State Street v. Signature, 1998). State Street involved a "Hub and Spoke" data processing system allowing mutual. funds (Spokes) to pool their assets into an investment portfolio (Hub) organized as a partnership, calculating percentage shares of assets and expenses for each partner, allowing for tax advantages and administrative cost savings. The lower court had rejected the validity of the patent, stating that the mere change of one set of numbers into another, without more, is insufficient to confer patent protection.

The Federal Circuit reversed, holding that the proper inquiry is whether the application produces a "useful, concrete, and tangible result." The court did not abandon the limitation that a patent was only available for useful ***applications*** of mathematical algorithms, but broadened applications to include even the manipulation of data. The court went on to reject the business methods exception to patentable subject matter as well, removing a barrier for a broad range of software. Earlier courts and the PTO had held that while a means of doing business may be patentable, a method of doing business is not (In re Schrader, 1994; Ex Parte Murray, 1988). In State Street, the Federal Court rejected this exception, finding that such an exception "blurs in the complexity of the modem world" (State Street v. Signature, 1998; AT&T v. Excel, 1999).

In recognition of this direction from the Federal Court, the PTO has also liberalized the availability of patents for software. In 1996, the PTO published Examination Guidelines for Computer-Related Inventions to assist PTO personnel in their review of the growing number of software patent applications. The Guidelines recognized the patentability of software and focused the PTO review of software patent

applications on the traditional tests for novelty and non-obviousness. The Guidelines do not have the force of law but they have had a major impact on the number of software patent applications being filed and granted. Moreover, since the PTO's issuance of a patent carries a presumption of validity, the Guidelines have had the effect of increasing the number of patent enforcement actions for software (Amazon v. Barnes & Noble, 1999).

The Guidelines require that, to be patentable, a software patent claim must define structural and functional interrelationships between the software and other claimed aspects of the invention. The Guidelines break down software into functional and non-functional aspects. Functional aspects includes computer programs and data structures, and, when encoded on to a computer-readable medium, are patentable. Non-functional aspects, such as music and literary works, are outside the scope of patent law protection.

Statutory Requirements. Determining that software is patentable subject matter under Section 101 of the Patent Act is merely the first step in obtaining a software patent. The software patent application must also meet the same stringent standards applied to all patent applications. Those standards include four basic tests:

(i) **Utility**. The software must have "utility", i.e., the patent application must include a useful process (35 U.S.C. §101, 112; RCA v. Applied Digital Data, 1984). Inventions must possess a certain level of "real world" value (1996 Guidelines). Generally, software is developed to implement desired functionality and therefore easily meets the utility requirement.

(ii) **Novelty**. The novelty standards under Section 102 of the Patent Act limit the time in which a patent application can be filed (35 U.S.C. §102; U.S.P.T.O., 1992). An invention is not considered to be novel and is ineligible for patent protection if it was known or used by others in the United States prior to the date the patent applicant "invents" the claimed process. Moreover, even if the applicant is the first to invent the process, if he/she fails to file an application within one year of the date the process is first described in a printed publication in the U.S. or a foreign country or in public use or on sale in the United States, the applicant will be barred from obtaining patent protection.

The process of software engineering raises complex issues regarding novelty In Re: Hall, 1986; Alexander, 1988). For example, when is software "invented": at the time of conception, specification, initial coding, beta testing or commercialization? Specifically, courts have held the software is invented when there is an operable form of the software code demonstrating the program's utility (King v. Otari, 1985; CCH, 2000). Thus "invention" for purposes of Section 102 can occur prior to the completion of a final commercial version of a program, and, if portions of the code demonstrating the program's unique functionality are complete, can occur prior to the time that the first version of the code for the program is fully complete.

(iii) **Nonobvious**. Under the Patent Act, in addition to the objective standards for novelty described in Section 102, a patent application must also meet a subjective "nonobvious" standard: the claimed invention, considered as a whole, is not patentable if it would have been obvious at the time of the invention to persons having ordinary skill in the field to which the invention pertains (35 U.S.C. § 103; Graham v. John Deere, 1966). In 1996, the Supreme Court established the current four part inquiry relating to obviousness: (i) review the relevant prior patents and other public information available at the time of the invention (the "prior art"); (ii) ascertain the differences between the prior art and the claimed invention; (iii) determine the level of "ordinary skill" in the field of the claimed invention; and (iv) examine other objective evidence of nonobviousness, including commercial success of the claim, the failure of others to invent a similar process, and long unresolved needs in the field that are solved by the claimed invention (Graham v. John Deere, 1966; U.S.P.T.O., 2000).

As indicated above, a determination of "nonobviousness" is a subjective determination. The four part inquiry set out in Graham is merely the starting point for a case-by-case analysis by a determination of the PTO (and ultimately the courts) and obviousness can only be made by an experienced patent practitioner. However, it is clear that the mere conversion of a manual calculation to a computerized calculation is not sufficient to overcome the hurdle of nonobviousness. For example, a software program that replaces a manual system for calculating horse racing information with a computerized system for calculating the same information is not patentable (Digitronics v. N.Y. Racing, 1977).

(iv) **Specification**. Section 112 of the Patent Act requires that, in return for patent protection, an inventor must satisfy three requirements: (1) adequate disclosure; (2) enablement; and (3) best mode (35 U.S.C. § 112; Kewanee Oil v. Bicron, 1974). The first requirement is aimed at preventing the grant of a patent before the invention has been fully realized. Adequate disclosure is satisfied if the specifications in the patent "reasonably convey to those skilled in the art that the applicant was in possession of the claimed invention as of the date of invention" (1996 Guidelines).

The enablement requirement seeks to ensure the public benefits from the patent bargain. To comply, disclosure in a patent application must provide a description of the process in such sufficient detail to enable a person skilled in the art to make and use the claimed invention without undue experimentation. Applying this standard to software, the Federal Circuit has held that the disclosure in a patent application was inadequate where it would take a skilled computer engineer over two years to develop a software program based on the patent disclosure (White Consolidated Industries v. Vega Servo-Control, 1983). On the other hand, it is not usually necessary to disclose the source code for the software. It is generally sufficient if the functions of

the software are disclosed since creation of the specific source code is within the ability of a programmer skilled in the art of software development (Robotic Vision v. View, 1997).

Finally, to satisfy the best-mode requirement the inventor must disclose the best form of the invention contemplated at the time of the application.' This restrains inventors from obtaining a patent while at the same time concealing the preferred mode of operation.

Generally, determining the level of disclosure in a patent application is a balancing act between competing goals. In addition to meeting the disclosure requirements of Section 112, the inventor would like a broad, detailed disclosure to cover all potential competing software programs. On the other hand, in a competitive environment, software owners will strive to limit the disclosure of important trade secrets associated with their software. Striking this balance is the "art" of the patent attorney (Chisum, 2000; Lipscomb, 2000).

Rights. A patent owner is granted the exclusive right prevent others from making, using or selling its invention in the United States for a period beginning on the date the patent issues and ending 20 years after the date of the filing of the patent application (35 U.S.C. § 154). During the period of exclusivity, the patent owner may prevent others from using the invention and making or selling products created by the invention, and, unless there are prior patents or other third party rights that are necessary to use the invention, the patent owner has the exclusive right to use the invention internally and/or to license it to any third parties (35 U.S.C. §271; U.S. v. SK mbH, 1981).

Unlike copyrights or trade secrets, patents protect against independent invention (Kewanee Oil v. Bicron, 1974; CONTU 1978). In other words, even if a competitor were to develop software without copying the original program or misappropriating confidential information regarding the original program, the patent owner could still prevent the competitor from making, selling or using the competing product as long as it was described in the patent claims approved by the PTO.

Based on the broad scope of patent protection, owners frequently use license agreements to tightly control use of their invention. Many of the same considerations discussed earlier with regard to copyright licenses apply to patent licenses. However, because of the anti-trust concerns regarding misuse of the patent holder's exclusive rights, such agreements must be very carefully drafted and should be reviewed by counsel (Holmes, 2000; Epstein, 2000).

Creation. As noted earlier, to obtain a patent, the software owner must file a patent application with the PTO. Under the novelty standards discussed above, the patent application must be filed within one year of the public use or sale of the invention (for software, the date of first licensing could qualify as such use) or, if a foreign patent is filed first, within twelve months of filing a patent application for the invention in a foreign country. In contrast to copyright registration, filing an application for patent protection is costly and requires the skills of a patent attorney (Chisum, 2000; Lipscomb, 2000). Generally, a patent application includes a review of the prior art, abroad description of the invention and,

most importantly, specific statements or claims describing the novel features that constitute the invention. The claims define what is included and excluded from patent protection and are the focus of the PTO's examination process (Chisum, 2000; Bender, 1991). Once a patent application has been submitted, it will be reviewed by a patent examiner that specializes in software examinations. The patent examination process typically takes from one to three years. Generally the patent examiner will object to language within the patent claims and the patent attorney will attempt to overcome the objections while providing the patent owner with the maximum protection possible. Once the examiner is satisfied, the PTO issues a notice of allowance and after the payment of an issue fee, issues a patent for the invention. The period of exclusivity continues as long as the proper patent maintenance fees are paid (35 U.S.C. §§41, 154; CCH, 2000).

Although a detailed review of the patent examination process is beyond the scope of this chapter, two unique aspects of U.S. patent law should be noted. First, if a patent applicant is dissatisfied with the decision by the patent examiner, the decision can be appealed to an appeal board within the PTO and ultimately can be appealed to the federal courts (35 U.S.C. §§7, 141, 145, 146). Second, if the claims recited in one application overlap with claims of a second application filed by another party, ownership of the invention will be determined in an interference proceeding before the PTO (35 U.S.C. §135; Newkirk v. Lulejian, 1987). The interference proceeding determines which party was the first to invent the novel process and therefore has the right to obtain a patent. It should be noted that most foreign countries grant patent rights to the "first to file" rather than the first to invent and interference proceedings are uniquely American (35 U.S.C. § 102(g); Melville, 2000).

It should also be noted that unlike copyright, United States patent protection only applies to domestic use and manufacture and software owners must obtain separate patents in other countries if they want protection outside of the U.S. Although a description of international patent practice is beyond the scope of this chapter, a brief introduction to the Patent Cooperation Treaty ("PCT") is useful. Under the PCT, a U.S. patent owner can file a single international patent application to be filed in countries that have signed the PCT, including such important countries as England, Germany, Japan and Russia. Filing a PCT application preserves the right to file individual patent applications in countries designated by the applicant for up to 30 months. During this time, the software owner can determine whether it is worthwhile to incur the significant expense of obtaining individual patents in the designated countries (P.C.T., 2000; Melville, 2000; CCH, 2000).

Once a patent is granted, to fully protect its rights, a patent owner must vigorously prosecute infringement of its patent. In addition, the owner should place a patent notice ("U.S. Patent Number _____") on every piece of patented software that is distributed. Similar to a copyright notice, a patent notice should be placed not only on the disks and tapes that contain patented software, but should also be added to the source code and appear on the first screen of the video display of the software. Like

copyright notices, patent notice prevents a competitor from raising the innocent infringer defense (35 U.S.C. §287; CCH, 2000).

TRADEMARKS

Introduction. The most frequently overlooked data right regarding software is trademark rights (Gilson, 2000; McCarthy, 2000). In fact, based upon the familiarity of marks such as 1-2-3, WINDOWS and QUICKEN, it can be argued that trademarks may be the most important data right in the retail software market (Raysman, 1992). Like trade secrets, the law of trademarks arose from general notions of fair competition under the common law. Courts held that it would be unfair to both trademark owners and consumers if competitors used confusingly similar names to sell competing goods or services. For consumers, trademarks and service marks provide information and clarity by designating the origin of goods and services. For owners, trademarks and service marks serve as useful marketing tools and as repositories for goodwill developed regarding the owner's goods and services.

A number of attributes related to software can be protected with trademark rights. Most commonly, the name of the software program can be protected as a trademark (e.g., 1-2-3 or EXCEL). In addition, consulting, data, maintenance and retail services can be protected with service marks (e.g. BLOOMBERG or COMPUTERLAND). Finally, company names can be protected with tradenames (e.g. APPLE or LOTUS) (it should be noted that the law governing and protecting trademarks, service marks and tradenames is essentially identical (15 U.S.C. § 1053; AT&C v. AC&T, 1987)).

Rights. Trademark data rights prevent competitors from identifying their goods or services in a manner that creates "a likelihood of confusion" with the original mark and takes advantage of the goodwill established by the original trademark owner (15 U.S.C. § 1114; Polaroid v. Polaraid, 1963). Returning to the 1-2-3 example, trademark data rights prevent third parties from selling spreadsheet programs under the name 1-2-3. More broadly, the 1-2-3 mark prevents competitors from using similar names (e.g. 3-2-1 or ONE-TWO-THREE) for similar products (financial software programs).

Courts consider a series of factors in determining likelihood of confusion: (i) the relative strength of the original mark, (ii) the similarity between the original and infringing marks, (iii) the similarity between the goods or services associated with each mark, and (iv) actual evidence of consumer confusion between the original and infringing marks (Hasbro v. Lanard, 1988; McCarthy, 2000). Therefore, the breadth of trademark protection is dependent in part on the strength of the original mark. Strong marks provide protection even where the competing marks and goods or services are not identical.

Generally, the strongest marks are arbitrary and fanciful marks such as APPLE or LOTUS (Gilson, 2000). Like MCDONALD'S for hamburgers, such marks have no natural connection to the goods or services they identify. Therefore, it is assumed that competitors using similar marks are purposely infringing the original mark. On the other hand, the weakest marks are those that are descriptive of the good or service they identify since competitors may use such marks merely to describe their goods and

services rather than to attempt to impermissibly use the mark owner's goodwill. For example, on grounds of descriptiveness, courts have refused to protect the marks CONTINUOUS PROCESSING or INTELLIGENT MODEMS (Stratus v. NCR, 1987; Hayes v. Business Computer, 1983). In fact, a descriptive mark cannot be protected unless the owner can show that buyers associate the mark with the owner despite the descriptive nature of the mark ("secondary meaning") and there is evidence that infringement would cause actual consumer confusion (15 U.S.C. § 1091; Armstrong v. Nu-Enamel, 1938). Moreover, even if a descriptive mark develops secondary meaning, trademark protection for such marks is weak. For example, although COMPUTERLAND is now a recognized tradename, the tradename has not prevented competitors from using similar names such as BUSINESSLAND or COMPUTER FACTORY.

There is no limit on the length of trademark protection. Protection lasts from the first use of the mark until the trademark owner abandons the mark or the mark becomes generic. Abandonment takes place when a trademark owner clearly indicates that it intends to stop using the mark (15 U.S.C. § 1127; Exxon v. Humble Exploration, 1983). A mark becomes generic when it becomes a descriptive name for a general category of goods or services rather than a distinctive mark identifying the source of the goods or services. For example, originally the term CELLOPHANE was a trademark identifying tape produced by DuPont (DuPont v. Waxed Products, 1936). However, CELLOPHANE became "generic" and lost its trademark protection when the public began to use 'the mark to describe transparent tape regardless of the manufacturer (Examples of other generic marks: ASPIRIN, LIGHT BEER, THE COMPUTER STORE, HOG) (Bayer v. UDC, 1921; Miller v. G. Heileman, 1977; In Re: The Computer Store, 1981; Harley-Davidson, Inc. v. Grottanelli, 1999).

Creation. Development of trademark data rights involves three steps: (i) selecting a protectable mark; (ii) using the mark to identify particular goods and services; and (iii) enforcing the mark by preventing competitors from using confusingly similar names.

As described above, arbitrary names such as APPLE make the strongest trademarks and descriptive names such as COMPUTERLAND provide limited protection. Unfortunately, there are practical considerations that limit the use of arbitrary marks. Until an arbitrary mark is well known, it will be of limited use as a marketing tool or in designating the origin of goods or services. On the other hand, even when a descriptive mark such as COMPUTERLAND is not well known, consumers can quickly deduce what goods or services are represented by the mark. To alleviate this chicken and egg problem, trademark owners attempt to find "suggestive" marks that are neither completely descriptive or totally arbitrary (Union Carbide v. Ever-Ready, 1976; McCarthy, 2000). While such marks suggest the relevant goods or services, they are not merely descriptive and therefore can provide substantial trademark protection. For example the mark SIDEKICK provided Borland International with significant trademark rights and also was suggestive of the program it identified (memory resident software that is accessible while other programs are running).

In addition to the arbitrary/descriptive aspects of selecting a name, a trademark owner must also determine whether a potential name is available - i.e. is the proposed name confusingly similar to marks already in use. Not only must the owner protect itself from potential infringement suits, but the strength of the mark is determined in part by its uniqueness. If there are no similar marks for competing products, the scope of trademark protection is potentially broader (Miss World v. Mrs. America, 1988; Tektronix v. Daktronics, 1975). To review the availability of proposed names, trademark owners typically conduct a search of federal and state trademark registrations and other data bases. In the past, trademark searches were completed by professional search fines that charged substantial fees. More recently, with the advent of data bases and other information accessible on the Internet, the time and cost of these searches has been reduced (Gilson, 2000).

Once a mark is selected, the next step to developing trademark data rights is proper use. The mark should be placed on all goods and services that are distributed to third parties. Trademarks for computer programs should be placed on all packaging, magnetic disks and tapes, source code and screen displays associated with the program. In addition, to prevent the mark from becoming generic, the mark should be used as an adjective, not a noun. For example, "the 1-2-3 program" is proper trademark usage; "I am going to use 1-2-3" is not. In addition to using the name correctly, a trademark owner should include a proper trademark notice with each use of the mark ("®" for registered marks; "^{TM}" for unregistered marks). Notice alerts competitors that the owner is serious about enforcing trademark rights and precludes the innocent infringer defense (15 U.S.C. §§1072, 1111; Polo v. Extra Special, 1980).

Like all data rights, the final step in developing trademark rights is vigorous enforcement. All alleged infringements must be tracked down and prosecuted. If a trademark owner fails to take action against infringing users, this failure will be used as evidence that the value of its mark is weak (P&G v. Johnson & Johnson, 1979; McCarthy, 2000). On the other hand, the time and expense incurred by a trademark owner successfully stopping use of a similar name will add to the value and goodwill associated with the mark.

United States registration of a mark is governed by the Lanham Act under which Congress established a national registration system to bring uniformity to use of marks in interstate commerce (Lanham Act, 1946; U.S.C. §§ 1051-1127; McCarthy, 2000). Like copyrights, trademark registration is not mandatory but provides both legal and practical advantages. A registered mark is presumed to be valid in any action in the federal court system. Moreover, federal registration applies on a national basis. Even if a registered mark has not been used in all 50 states, competitors are prevented from adopting similar marks anywhere in the United States (15 U.S.C. § 1072; Dawn Donut v. Hart's Food, 1959). Finally, the Lanham Act provides statutory damages for infringement of registered marks. In addition to these legal advantages, federal registration provides strong practical protection (Gilson, 2000; McCarthy, 2000). First, prior to granting a trademark registration, the PTO conducts a search to verify that there are no existing marks that are confusingly similar to the new mark. Second, the publication of a registered

mark has the practical effect of preventing others from adopting similar marks. As indicated above, prior to selecting a new name, trademark owners generally search prior federal registrations and reject proposed names that conflict with registrations. Third, federal registration notifies suspected infringers that a trademark owner is serious about enforcing its rights. Fourth, federal courts are experienced in handling trademark actions and provide a good forum for trademark owners to enforce their rights.

Obtaining a federal registration is more complicated than obtaining a copyright registration and generally requires a trademark specialist (15 U.S.C. §§1051-11.13; Kramer, 2000). An application includes a drawing of the mark, a description of the goods or services associated with the mark and, if registration is based on actual use, specimens of the mark as it is used in interstate commerce. A trademark examiner reviews the application to determine whether registration is appropriate and may respond with a series of objections to the language of the application. Following negotiations regarding these objections, if the application is accepted, the PTO publishes the mark so that competitors have an opportunity to oppose the registration. If there are no oppositions filed, a registration certificate is issued for the mark. The examination process takes six to eighteen months and federal protection lasts ten years from the date of the registration certificate. To maintain the registration, in the seventh year following the registration, a trademark owner must file an affidavit indicating that the mark is still in use and providing a specimen of such use. Thereafter, if a trademark owner renews its mark every ten years, the registration can be maintained indefinitely.

There are two types of federal registration applications. Under the traditional method, a trademark owner files an application after the mark has been used in interstate commerce. In addition, in 1988 the Lanham Act was amended to add an alternative application process where there is an "intent to use" the mark after filing the application (15 U.S.C. §§1051(b), 1057(b); Trademark Revision Act, 1988). Under this system, if the PTO accepts the application, to obtain the registration, the trademark owner must file an affidavit within a specified time period indicating that the mark has been used in interstate commerce.

SEMICONDUCTOR PROTECTION ACT

Introduction. Copyrights, trade secrets, patents and trademarks evolved long before the invention of the computer. Commentators have noted that individually, each of these intellectual property rights does not adequately protect the proprietary interests of software owners (CONTU, 1978; Samuelson, 1985). The inadequacies of traditional intellectual property law were particularly troublesome for. software programs that were embedded on silicon chips. Although copyright protected the design drawings for these chips, since the actual chips were utilitarian articles, the chips themselves were not protected by copyright. Trade secrets were unavailable because chips were publicly distributed and patents were frequently unavailable because the chips were mere implementations of software or manual processes that previously had been used in other formats. With no clear intellectual property rights and a

general lack of respect for such rights in many countries that manufactured silicon chips, piracy was a serious problem for U.S. chip designers (H. R. Report No. 781, 1984; Brown, 1985).

To alleviate these difficulties, based on the constitutional directive to promote science through protection of writings, Congress passed the Semi-Conductor Chip Protection Act of 1984 ("Chip Act") (17 U.S.C. §§901-914; Semi-Conductor Chip Act, 1984). The Chip Act is modeled after the Copyright Act: it protects original works upon fixation, the copyright office is responsible for registration, and protection only extends to the expressions embedded in the silicon chips, not the ideas or functionality associated with the chips. Specifically, the Chip Act protects so-called "mask works" that are embedded on semi-conductor material. The Chip Act defines a mask work as a series of related images of metallic, insulated or semi-conductor materials set down in layers that relate to one another, which are wired to perform electrical circuit functions (17 U.S.C. §901(a)). Mask protection is available to United States residents and is available to foreign parties if they first exploit their chips in the United States.

Rights. Chip owners have the exclusive right to copy, distribute and sell mask works protected by the Chip Act. Like the Copyright Act, nothing in the Chip Act prevents competitors from independently creating chips containing ideas from the original mask work as long as they do not copy the actual circuit design (17 U.S.C. §902; Brooktree v. AMD, 1988). Moreover, although third parties cannot copy the chip, under the first sale doctrine, chips that are purchased legally can be resold without limitation.

Recognizing the relatively short commercial viability of mask works, the Chip Act only provides protection for ten years after either (i) the date of registration of the mask work or, (ii) if the mask work is registered after distribution, the first date the mask work is distributed commercially (17 U.S.C. §904; H. R Report No. 781, 1984). In addition to the injunctive relief and compensatory damages, the Chip Act provides for statutory damages of up to $250,000 in the event of violation of mask works rights.

Creation. Unlike copyright, a chip owner must register its mask work to obtain mask work protection. Therefore, it is important for chip owners to register mask works prior to any third party distribution of their chips and to protect proprietary rights in such chips prior too registration with copyrights and trade secrets. By statute, in order to obtain mask work protection, the registration application must be filed with the copyright office within two years of commercial exploitation (17 U.S.C. §908). Like copyright, registration applications for mask works are relatively simple including filing forms published by the copyright office along with a deposit of the mask work. Moreover, based on the Chip Act's intent to protect against piracy, the copyright office has issued regulations limiting the deposit requirements for registration (37 CFR §211.5).

There is no requirement that notice be placed on the mask work in order to maintain Chip Act protection. However, like all data rights, notice is relatively simple and has many practical advantages. A mask work notice (identical to a copyright notice with a circle "m") should be placed on the actual chip, packaging for the chip and on any video display produced by the chip. Notice precludes the innocent infringer defense and therefore makes compensatory damages available for infringement prior to the filing

of a lawsuit (17 U.S.C. §909). In addition, notice has the practical effect of putting competitors on notice that a mask owner is serious about establishing and protecting data rights.

SELECTING AND MAINTAINING DATA RIGHTS

Introduction. Once software owners understand the broad menu of available data rights, they must select and maintain the best set of data rights for their particular software programs. This portion of the chapter divides this issue into four general rules:

(i) Copyright everything;

(ii) Protect limited distribution software through trade secrets; and

(iii) Where possible, protect widely distributed software with patents, trademarks, trade secrets and, where appropriate, mask works; and

(iv) Use licensing agreements to plug the holes.

Copyright. As described above, copyright data rights have weaknesses. Copyright only protects expression and does not prevent third parties from taking an idea such as an electronic spreadsheet and creating competitive products (CAI v. Altai, 1992; Lotus v. Paperback Software, 1990). However, copyright does protect against the most common breach of software data rights: direct copying of a software program without permission of the original owner. Moreover, copyright protection extends to many different aspects of the software development process described in this encyclopedia including specifications, flowcharts, test versions, object code, source code, screen displays, installation instructions and user manuals. Copyright is inexpensive, effective immediately upon creation of the work and protected on a national and international basis. Therefore, it is important that software owners effectively "copyright everything" and review the steps necessary to maintain and strengthen their available copyright data rights.

A strong copyright protection program includes policies and procedures to regulate the copying and distribution of copyrighted works (Nimmer, 2000; Epstein, 2000). For example, placing serial numbers on all copies of a software program and requiring users to register with the original owner is an effective means of policing illicit copying. In addition, copyright notices should be placed on all written materials that are distributed to third parties. Notices should be prominently displayed throughout the work to warn potential infringers that the owner intends to enforce its copyright data rights vigorously. A copyright program should also include a system whereby the most valuable aspects of a written work are registered with the copyright office, such as the final versions of object code, source code and user manuals that are distributed to software users.

A copyright protection program should resolve ownership issues that arise regarding written works developed or marketed by the company. Therefore, a software owner should execute written agreements that clearly establish ownership rights in original works created for the company by

independent contractors and in any modifications or derivative works to the original program developed by third parties (17 U.S.C. §201; C.C.N.V. v. Reid, 1989; Kreiss, 1991). When it is unclear who owns copyright rights, the software owner should obtain written copyright assignments from all parties involved in the development of a software program.

Limited Distribution. The second step in protecting software data rights is filling the "holes" left by copyright. The type of supplemental data rights to fill such holes depends on the type of software to be protected. For software that is solely used on an internal basis or where distribution is otherwise limited, . trade secrets can effectively and economically bolster copyright protection. In addition to protecting the expression contained in software programs and their related written materials, trade secrets protect ideas and information embodied in those materials. Moreover, without the expense of filing federal registration applications, software owners can protect confidential information regarding the ideas and content contained in a protected program.

Once a software owner determines which programs and related materials are appropriate for trade secret protection, it is essential that an overall security program be `adopted (Epstein, 2000; Milgrim, 2000). As described above, not only is it practically important for maintaining a competitive advantage, but from a legal standpoint, courts will not protect trade secrets unless a software owner can show that it operates a strong secrecy program. Therefore, all written materials that contain confidential information should be identified, isolated and legended. The trade secret owner should adopt a confidentiality policy that is well known to employees, suppliers and customers and limits access to the confidential information. Procedures should be implemented to record the names of all persons with access to trade secrets including visitor registration logs and software copy inventories. The policy should include standard confidentiality agreements that are executed by all third parties that come into contact with the trade secrets. Perhaps most importantly, the secrecy program should include a consistent policy of investigating and prosecuting unauthorized trade secret disclosures.

It should be noted that although there is no legal reason that patent, trademark or mask work protection cannot be used with limited distribution software, practical considerations restrict use of these additional data rights. First, trademarks and mask works are specifically designed to protect products that are widely distributed. Second, patents preclude the use of trade secret protection by publicly disclosing valuable information regarding the software. Third, the time and expense in creating and maintaining these additional data rights may not be cost effective for software with limited distribution potential. On the other hand, if a software owner has previously adopted a secrecy program to protect general business information, the incremental cost of adding software to the list of protected information may be minimal.

Widespread Publication. The "holes" in copyright protection grow exponentially when software is widely distributed. It becomes increasingly more difficult for copyright owners to prevent unauthorized reproduction as the number of authorized copies grows. Moreover, widespread distribution of software

code is also widespread distribution of the ideas embodied in the software and provides competitors with information they need to develop effective competing programs that avoid copyright infringement. Therefore, widespread distribution requires the software owners to employ every available data right to protect their proprietary position.

Ironically, although object code, screen displays and user manuals lose their trade secret protection upon widespread distribution, trade secrets are still useful for plugging certain copyright leaks.. Information regarding software programs that is not widely distributed, such as source code and confidential business and development information, can still be protected as a trade secret. Identification and isolation of this .information should be completed prior to initiating distribution so that these items will not be inadvertently disclosed. Moreover, even if such information is disclosed in a patent application, the PTO retains the application in confidence and trade secret protection continues until the date the patent is granted (unless a foreign application is filed in which case the application will be disclosed 18 months after filing). In fact, if the patent is denied, the PTO will seal the "file wrapper" that contains the application proceedings and trade secret protection may still be available for confidential information contained in the application (35 U.S.C. §122; Kewanee Oil v. Bicron, 1974).

The ultimate supplemental protection for software is patent law (Bender, 1989; Chisim, 2000). Because a patent prohibits all unauthorized use of protected software, including similar programs that are independently created, patent rights are often more effective against competitors than copyrights or trade secrets. Moreover, while trade secrets and copyrights technically can have longer terms, practically, the right to independent creation limits the effective length of these protections to less than five years. Although patent rights last only 20 years, because of the rapid advances in the software industry, this additional time generally provides more than sufficient time to protect the useful life of a software program.

As noted above, despite these advantages, patent protection is available only to certain software programs. The program must meet strict standards for utility, novelty, nonobviousness and disclosure. In addition, obtaining a patent is expensive and time consuming. Before filing a patent application, a software owner must determine whether the software is a commercially significant invention that cannot be adequately protected by copyright. In other words, will extra protection gained from a patent generate sufficient revenues to justify the costs of obtaining the patent (Bender, 1992; Smith, 2000). If the answers to these questions are positive, the software owner should consult a patent attorney as early as possible during the software development process. The attorney can be helpful in structuring development and use of the software in a manner such. that patent protection is more likely to be available. Once the software is publicly disclosed or offered for sale, it is important that action be taken quickly to obtain such protection because of the strict one-year time limits for patent applications (in the U.S. - generally, foreign patent protection is not available unless an application is filed prior to public disclosure).

Moreover, any software program that is commercially distributed, even if it is protected by copyrights and/or patents, can benefit from trademark protection. Even if a competitor avoids copyright and patent infringement, trademarks will protect against misappropriation of good will established in the original program (in fact, in situations where other data rights are unavailable, trademarks may be the only available data right). Therefore, prior to widespread distribution of a software program, the software owner should select a protectable mark for the program, use the selected mark as frequently as possible and apply the "TM" notice to all uses of the mark. Like patents, the software owner must then analyze whether the commercial potential of the software program justifies the cost of federal registration (McCarthy, 2000; Raysman, 1992). Since the cost is significantly less than patent registration, generally the answer will be yes.

Finally, even where data rights do not provide adequate protection, contract rights may provide certain protections for a software owner. A good example of this final line of protection is found in the recent federal case upholding the enforceability of shrink-wrap license agreements (ProCD v. Zeidenberg, 1996). ProCD produced SelectPhone, a software package that contained a national directory of approximately 100 million telephone listings and other data for individuals and businesses residing within the United States. The SelectPhone user guide contained a Single User License Agreement that stated that by using the software, the licensee agreed to be bound by the license terms that the software may only be used for personal use. While SelectPhone was being installed on a licensee's computer and with each subsequent download, a message appeared reminding the customer that the use of SelectPhone was subject to the shrinkwrap license. Zeidenberg and his company, bought ProCD's software package, and, despite the license agreement forbidding commercial use, made the SelectPhone listings available on the Internet.

ProCD filed suit for copyright infringement and violation of the shrinkwrap license. The lower court ruled that the data contained in SelectPhone was factual and, therefore, not protected by copyright. On appeal, the Circuit Court found the shrinkwrap license to be enforceable and held that by purchasing the software, the defendant agreed to be bound by the terms of the license (much like the terms on the back of an airplane ticket). Since the terms specifically prevented the licensee from using SelectPhone for commercial purposes, even though the listings were not protected by copyright, Zeidenberg was still prevented from making the listings generally available for sale on the Internet.

COMMON DATA RIGHTS ISSUES

This chapter will now review a number of common data right issues that software owners face on a daily basis: (i) idea vs. expression in copyright law; (ii) data rights of employees and independent contractors; (iii) government data right and (iv) internet related issues.

IDEA VS. EXPRESSION

Introduction. As described above, distinguishing between protectable expression and unprotectable ideas lies at the heart of determining the extent of copyright protection for software programs. Unfortunately, as the leading federal court decision in this area noted:

> "Drawing the line between idea and expression is a tricky business ... [no]body has ever been able to fix that boundary, and nobody ever can ... Decisions must therefore inevitably be ad hoc." (CAI v. Altai, 1992).

The extremes are clear. Copyright prevents making copies of an original software program, such as copying the 1-2-3 program from one floppy disk to another. The other extreme is equally clear. General ideas regarding the functionality of the program cannot be protected by copyright, such as the general idea of an electronic spreadsheet. In between these extremes, the line between idea and expression determines whether a similar competing software product violates the copyright rights of the original program.

Abstraction. Generally, courts apply a strict test to determine copyright infringement for software where there is no identical copying. This "abstractions" test was first formulated by Judge Learned Hand in the case of Nichols v. Universal Pictures:

> "Upon any work, and especially upon a play, a great number of patterns of increasing generalities will fit equally well, as more and more incident is left out. The last may be perhaps no more than the most general statement of what the play is about, and at times may consist only of its title; but there is a point in this series of abstractions where they are no longer protected since otherwise the playwright could prevent the use of his "ideas," to which, apart from their expression, his property is never extended." (Nichols v. Universal Pictures, 1930).

The Nichols case involved two literary works, the play "Abie's Irish Rose" and a subsequent motion picture "The Cohens and The Kellys." Both works included a quarrel between a Jewish and Irish father, the marriage of their children, the birth of grandchildren and a reconciliation. However the court held that these plot abstractions were not protectable and that the actual scripts of the play and movie were not substantially similar.

Applying the abstractions theory to software, like the script of a play, the complete version of the computer code serves as the starting point for the abstraction analysis. From there, more general abstractions regarding the code are reviewed: subroutines and modules, flow charts and structural designs, code and functional specifications and finally, general descriptions of the code's purpose. At some point along this compendium, the abstractions turn from copyrighted expressions to unprotectable ideas (Softel v. Dragon, 1998).

The second step in the abstractions test is determining which abstractions are ideas and which are expression. Referred to as the "filtration" step, to determine that line, generally courts review whether an abstraction can be expressed in more than one manner (Apple v. Microsoft, 1994; Mason v. Montgomery, 1992; Lotus v. Paperback Software, 1990). If there is only one practical means expressing the abstraction, the expression has merged with the idea and there is no protection. Other factors courts review include whether (i) the abstraction dictated by external factors such as the needs of the industry or requirements for compatibility (the so-called "scenes a faire" doctrine (Mitel v. Iqtel,1997)), and (ii) the abstraction contains non-protectable public information such as software code taken from an electronic bulletin board CAI v. Altai, 1992; Nimmer, 2000).

Once the abstractions are identified and classified as ideas or expressions, the final step in the abstractions test is to analyze the substantial similarities between the original work and the competing product. If areas of similarity include protectable abstractions, there is copyright infringement. On the other hand, where the similarity is limited solely to abstractions that are classified as ideas, there is no copyright violation.

Utilitarian / Look and Feel. It should be noted that prior to the Altai case, some courts were employing two more expansive tests for copyright infringement. Under the first test, the utilitarian theory, a court first identified the basic idea contained in the copyrighted work. Next, the copyrighted work was examined to divide the expressions contained in the work into two groups: (i) utilitarian expressions which are the sole means of expressing the idea embodied in the original work; and (ii) protectable expressions that are one of many methods for expressing such idea. Finally, the court determines what portions of the copyrighted work are substantially similar to the competing work: the utilitarian ideas or the protectable expressions. The second test, "look and feel" was originally developed to protect comic book characters from infringement. This test looks at the copyrighted work as a whole rather than reviewing individual elements and analyzes whether the competing work as a whole is so substantially similar to the original work as to copy its look and feel (Krofft v. McDonalds, 1977; Broderbund Software v. Unison, 1986; Berkeley, 1988).

These methods of analysis was applied to software in Whelan v. Jaslow Dental Laboratories, which involved an accounting system for dental offices (Whelan v. Jaslow, 1986). In this case, the defendant converted a minicomputer program developed by the original owner to software for use on personal computers. Therefore, the defendant's program contained very few lines of code from the original work. On the other hand, the court found that the basic idea of a dental accounting package could be expressed in many different manners and that the defendant had copied the one particular "structure, sequence and organization" that was first developed by the plaintiff. Therefore, while there was no literal copying of the original computer code, the defendant's program was held to violate the plaintiffs copyright rights.

Generally, in the context of software, since Altai, courts have rejected these two theories in favor of the narrow abstraction test (Gates v. Bando, 1993; Kepner-Tregoe v. Leadership, 1993). Many courts and commentators have stated that the utilitarian method protects too broad a range of expressions since there are many ideas embodied in software that are not included in the overall purpose of the software (CAI v. Altai, 1992; Plains Cotton v. Goodpasture Computer, 1987; Gage, 1987). Similarly, "look and feel" analysis has been widely criticized as extending software copyright expression to unprotectable ideas (Lotus v. Paperback Software, 1990; Nimmer, 2000; Samuelson, 1989).

1-2-3. The opinion in Lotus v. Paperback Software analyzes the idea/expression dichotomy as reflected in the 1-2-3 program (Lotus v. Paperback Software, 1990; Antton, 1990). In Lotus, the court examined the extent of copyright protection for expressions contained in the 1-2-3 user interface. The court held that the inverted "L" whereby numerical, statistical and other data was organized with letters across the top and numbers down the left side was part of the unprotectable idea of an electronic spreadsheet. On the other hand, the court held that the menu structure, including the choice and structure of command terms, their screen presentation, and the design of command prompts was capable of being expressed in many if not unlimited ways and was subject to copyright protection. Specifically, the court found that Paperback Software had arranged the commands and menus of its VP Planner so that they matched the 1-2-3 program keystroke for keystroke and contained specific menu expressions that were virtually identical to 1-2-3.. Therefore, the court found that VP Planner violated Lotus' copyright data rights.

Applying the theories discussed above to the Lotus case illustrates how the court drew the idea/expression line. At the highest level of abstraction, Lotus was unable to protect the basic spreadsheet structure because it was the sole practical means of expressing the idea of relating columns and rows of numbers. On the other hand, at lower abstraction levels, the specific type of menu expression and command language could be expressed in many different ways.

EMPLOYEE DATA RIGHTS

Introduction. If all software was developed and sold by software engineers working independently on their home personal computers, ownership of software data rights would be simple. However most software in the United States is developed under an employment relationship or a development agreement. Therefore, principles of employment and contract law frequently determine who "owns" the data rights discussed in this chapter. Basically, the principles applicable to the software development can be summarized in three basic rules:

(i) Employers retain copyright rights for software authored by employees within the scope of their employment under the "work-for-hire" doctrine;

(ii) Independent contractors generally own copyright rights in software they develop outside of an employer/employee relationship;

(iii) Since the foregoing rights can be altered by contract, to provide clarity to data right ownership, most parties that pay for software development insist that ownership rights be settled in a written agreement.

Copyright. Generally, employers retain the copyright to writings authored by employees under the work-for-hire doctrine. Prior to the adoption of the Copyright Act, the term "work-for-hire" was broadly interpreted to include not only employees, but also works produced by independent contractors that were specifically directed and financed by a third party. However, as a means of promoting the interests of authors in their writings, Congress limited the work-for-hire doctrine in the Copyright Act, to the employee/employer relationship and nine narrow exceptions (17 U.S.C. §101; H.R. Report No. 1476, 1976).

Despite this statutory language, a number of lower courts broadly read the term "employment relationship" to include-persons retaining independent contractors (Evans v. Chicago Software, 1986; Aldon v. Spiegel, 1984). However in Community for Creative Non-Violence v. Reid, the U.S. Supreme Court, based on the literal language of the Copyright Act, held that transfer of copyright ownership by the work-for-hire doctrine was strictly limited to the employee/employer relationship as determined under common law agency principles (CCNV v. Reid, 1989; Lee, 1990). The court noted that factors determining whether a common law agency relationship exists include the person controlling the manner and means of authorship, location where the work was written, duration of the relationship between the parties, right to produce additional products from the original work, author's right to hire other parties to complete his/her responsibilities, type of compensation (hourly vs. fixed price), payment of employee benefits and treatment of the author for tax reporting and withholding purposes. Based on these factors, in CCNV, the Court held that a sculptor performing a commissioned work was not an employee and retained the copyright rights to the sculpture absent a written assignment to the commissioning party.

Absent an agreement to the contrary, a programmer who is found to be an independent contractor retains the copyright in the software. The company that commissioned the work will most likely be found to be authorized to use a copy of the software and modify it as necessary, so long as it has duly paid the contractor (Aymes v. Bonnelli, 1995). The independent contractor still owns the rights to the software and could, unless contractually restrained, market it to the company's competitors. Spelling out the ownership issues in advance, therefore, becomes vital.

Work performed by employees specifically hired to create software is clearly a work for hire. The scope of employment may, however, become fuzzy when an employee works off hours at home, or performs a type of work different from their normal job description. If the employee develops software that helps them do their job, it still may be considered outside the scope of employment (Quinn v. Detroit, 1998; Roeslin v. DC, 1995). To avoid any such issues employment agreements should clarify the issues once any work has begun.

In addition to the employee/employer relationship, the Copyright Act lists nine specific categories where specially ordered or commissioned works will be considered works-for-hire. Many of these categories do not apply to software development but arguably three may be applicable: (i) collective works, (ii) translations, and (iii) compilations. To transfer copyrights in this manner, a work-for-hire must satisfy one of these categories and the independent contractor must expressly agree in a written instrument that the work shall be considered a work-for-hire (BPI v. Leith, 1981; Kreiss, 1991).

To alleviate any uncertainty regarding copyright ownership rights, most software development by independent contractors is governed by written development agreements that specifically set out copyright ownership rights. These agreements identify works covered by the agreement, highlight the work-for-hire provisions that transfer ownership rights and frequently contain an assignment by the independent contractor of all copyright rights he/she may acquire as author of the program (Whelan v. Jaslow, 1985; Roditti, 2000; Gordon, 1986).

Two final items should be noted about copyright ownership. First, joint copyright ownership only applies to cases where each of the owners actually writes a portion of the copyrighted work (Whelan v. Jaslow, 1985; Nimmer, 2000). In the typical software development situation where one party provides ideas regarding functionality and the other party writes the software code, only the author of the code retains copyright rights. Second, copyright assignments, like any contracts, can be oral or written. However, the copyright office only accepts written assignments of a copyrighted work (17 U.S.C. §§204-205; Forry v. Neundorfer, Inc., 1988). Since an assignee will be unable to bring a copyright infringement action until the copyright is registered, it is good practice to document any such assignments in a short written form that is acceptable to the copyright office.

Trade Secrets and Patents. Trade secret law attempts to balance between the rights of employers to protect confidential information and the rights of employees to freely use their general skills and knowledge to earn a livelihood. To draw this line, courts generally look at the relation between the confidential information and the employer's business. If the trade secrets are developed at the direction of the employer and/or contain specific confidential information related to a proprietary aspect of the employer's business, the information will be owned by the employer and protected by trade secret law (Q-Co. v. Hoffman, 1985; Telex v. IBM, 1975). On the other hand, if such information is not specifically related to the employer's business and/or is part of the general art of a specialist skilled in the employer's industry, the employer will receive no ownership rights (Plains Cotton v. Goodpasture Computer, 1987; Jostens v. NCS, 1982).

In a similar manner, the law governing ownership of patent rights balances competing interests between employers and employees. If an employee is hired to specifically develop a particular patented invention pursuant to an employer's directives, the employer owns. the resulting data rights (U.S. v. Dubilier, 1991; Mislow, 1985). However, if the employee is not hired to develop a specified invention, the employee retains the original patent ownership rights, subject to the doctrine of shop rights. The shop

rights doctrine provides that where an employee uses his/her employer's time and materials in developing an invention, the employer receives a royalty free license to use the patented invention. However, if the invention was completed by the employee during off-hours using his/her own resources, the employer will have no ownership or license rights in the invention.

Based on the ambiguity of the rights to employee inventions and know-how, one study found that in excess of 80% of major employers required their engineers to sign confidentiality / invention assignment agreements to clarify trade secret and patent rights (Lipscomb, 2000; Epstein, 2000). In addition to clarifying ownership of inventions and confidential information developed during company time, such agreements frequently provide that inventions developed on off-hours that are related to an employee's work for the employer will be disclosed to and/or owned by the employer. As a prophylactic measure, such agreements may also contain covenants-not-to-compete preventing employees from working for competitors for a period of time after termination of their employment. Although the law of such covenants is beyond the scope of this chapter, it should be noted that covenants containing excessive time or subject matter restrictions may not be enforceable (Trilog v. Famularo, 1974; Milgram, 2000).

GOVERNMENTAL DATA RIGHTS

Introduction. The previous discussions in this chapter were limited to data rights in the private sector. Predictably, the U.S. Government does not operate under the same set of guidelines. Software development and ownership involving the U.S. Government is complex and generally requires the expertise of a government law specialist (Geib, 1992; Samuelson, 1986). However, this chapter will attempt to introduce software engineers to the basics of governmental data rights with regard to software.

Government Employees. The rights to software developed by a government employee are very similar to the rights of an employee in the private sector. Unless the government specifically directs the production of an invention, a government employee may file a patent application for software invented while he was an employee. By statute, if the employee's department head certifies that the invention is likely to be used in the public interest, the government retains the right to manufacture and use the software for governmental purposes without payment of any royalty (35 U.S.C. §266, Lipscomb, .1982).

Copyright data rights in software developed by government employees are governed by the work-for-hire doctrine. If the software is developed by an employee within the scope of his/her employment, the employee has no individual copyright rights in the work (H. R. Report No. 1476, 1976; Nimmer, 2000). However, since the government cannot own copyrights, there is nothing that prevents the employee from independently reproducing, distributing or modifying such software.

It should be noted that employee ownership of patents, copyrights and trade secrets does not apply to matters created outside of governmental time (even if they relate to governmental matters) and

that all employee rights are subject to espionage laws, security clearances and other confidentiality restrictions placed on governmental employees for sensitive information.

Independent Contractors. Obtaining government funding for software development as an independent contractor is a mixed blessing. Generally, the government has unlimited rights to anything developed at government expense, including the right to transfer any software developed for the government to competitors. In fact, if not properly protected, merely licensing to the government software developed at private expense can open the software up to the full range of government data rights (CCH, 2000).

Although frequently criticized by commentators, there are legitimate purposes behind these broad data rights. The government is (i) seeking the maximum return from the taxpayer's investment in software development, (ii) promoting competition in the software industry to provide the government with second source opportunities, and (iii) encouraging the overall national development of software to foster international competitiveness and defense readiness. Obviously, these governmental interests conflict with commercial interests in protecting proprietary data rights and the government has encountered difficulties purchasing commercial software and locating software developers willing to work under such restrictions (Martin, 1987; Samuelson, 1986). Therefore during the past decade, the government has reexamined its position regarding data rights for software owners and developers and modified the Federal Acquisition Regulations and the Defense Acquisition Regulations to respond to these concerns (FARS, 2000; DFARS, 2000).

Developed at Private Expense. Data rights granted to the government under FARS or DFARS depend on whether software is developed at governmental or private expense. Software that is a commercial product (generally offered to the public prior to the time of the government contract), will be considered to be developed at private expense (48 CFR §227.472-3; Dowty Decoto v. Navy, 1989). Moreover, even if government funding is. used to debug a final program, a government contractor can still claim that the program was developed at private expense if it can prove that a "workable" version of the software was produced prior to the expenditure of governmental monies.

If it is determined that the software is commercially available, a software owner can unilaterally impose the following restrictions on the government: (i) full ownership rights in the software remain with the licensor; (ii) the government can only use the software on computers and at facilities specified in the license (the government has the right to use backup computers); (iii) the government can make copies solely for backup purposes; and (iv) the government has the right to modify the software solely for its own use (48 CFR §252.227). For noncommercial software that is developed at private expense (such as software that can only be used by the government), at a minimum the government retains the rights it receives for commercial software and the government will specify any additional government data rights in the final contract. In addition to software, government data rights extend to trade secrets and other technical data. If trade secrets and technical data are developed at private expense, as long as the trade

secrets are properly legended, the government will agree not to release the trade secret information outside the government.

Government Expense. On the other hand, if software or technical data is developed at government expense, generally the government will have the full ownership rights in the deliverables including rights to use them for any purpose, to copy, modify and incorporate them into other products and the right to disclose and transfer them to third parties (48 CFR §227.471). Therefore, it is important for private contractors to understand and carefully document the ownership rights in any software produced at government expense. Moreover, private contractors should carefully identify the deliverables subject to the contract. Anything not included in the deliverables, such as software developed before or after work on the contract, will fall outside of the reach of governmental data rights.

Procedures. In addition to reviewing potential governmental data rights, it is important to understand the procedures involved in implementing those data rights. If the specific procedures are not followed, the government may gain unlimited rights in software it acquires. All procurement proposals must include a description of data right restrictions that the contractor would like for the software and related trade secrets and technical data. If the government agrees to such restrictions, they must be documented in the final government contract with the developer (48 CFR §227.481). In addition, all software and technical data delivered to the government must include the legends described in FARS and DFARS regarding limited data rights. For commercial software, the legend must read:

> "Use or duplication by the government is subject to restrictions set forth in subparagraph (c)(1)(ii) of rights and technical data and computer software clause of FARS 52.227-7013." (DFARS 252.227-703)

For other items where the government agrees to retain the data in confidence, the following legend should be used:

> "Use and disclosure by the government is subject to the restrictions contained in contract number _____ with _____." (48 CFR §227.7013)

It should be noted that the best protection against unlimited governmental data rights is patent protection. As described above, except as set forth in an express government contract, nothing prevents a contractor from filing a patent application, although such patent may be subject to a royalty free government license.

INTERNET ISSUES

The Internet has raised many interesting data rights issues for software owners. For example , when Congress passed the Digital Millennium Copyright Act (DMCA) in 1998 to implement the international WIPO Copyright Treaties, they specifically addressed copyright issues raised by the Internet and modified the copyright law in a number of ways:

First, Congress addressed the question of whether carriers of Internet traffic could be liable for contributory copyright infringement. The DMCA limits the liability of "service providers" for transmitting third party messages, storing messages posted to their systems (e.g. bulletin boards), providing system caching (temporary storage of content to improve network performance), and providing links to other Internet sites. Service Providers are broadly defined to include any entity which provides network access, E-mail and webpage hosting (17 U.S.C. §512 (k)(1)). To qualify for the exemption Service Providers must agree to terminate users who repeatedly infringe copyright rights on the system. However the Service Providers need not actively monitor messages on its systems in an attempt to identify infringers. For messages posted to a Service Provider system or links provided to infringing material, the exemption provides that it will generally not be liable for infringement if they did not know and could not reasonably have known that the material was infringing. However, Service Providers that learn of infringement must act expeditiously to remove or block access to the infringing material.

Second, the DMCA makes it illegal to circumvent a technological measure that controls access to a copyrighted work without the permission of the copyright holder (17 U.S.C. §1201(a)), and also makes it illegal to manufacture, import, offer, or distribute a device or service that is primarily designed to circumvent copyright protection (17 U.S.C. §1201 (b)). Exceptions are provided for software developers to allow for reverse engineering of a program for the purpose of inter-operability, so long as copyright law would permit it and there is no other readily available means to achieve it otherwise. Research fines that test security programs for flaws and design encryption software also are provided a limited exemption.

Third, Section 1202 of the Copyright Act was added to make it illegal to remove or alter copyright management information conveyed along with a copyrighted work. Copyright management information includes the authors, owners, or performers in a work and the terms and conditions under which the work may be used. The intent of this section is to punish those who facilitate counterfeiting by stripping identifying information from the work. The section would also ensure that a shrinkwrap license remain would embedded in a computer program.

In addition to the protection provided to Service Providers under the DCMA for linking, courts have held that linking from one website to another, without permission, involves a violation of the Copyright Act since no copying is involved, and any information reprinted was factual and not copyrightable (Ticketmaster v. Tickets.Com, 2000). However, it should be noted that the court did find that the practice of "deep linking" to another parties' internal events pages (bypassing the original home page) could be a tortuous interference with prospective business advantage. The claim would be based on a disruption of advertising revenue if advertisers only paid for hits on the home page and not to an interior linked page. The court noted that if the site deliberately deep linked for the purpose of interfering with advertising revenue, then the linked site could recover damages.

Finally, the Internet raises interesting issues of regarding the enforceability of contract rights or other terms and conditions. By upholding the enforceability of shrinkwrap licenses, the ProCD decision

discussed above removed many of the ambiguities that surrounded the effectiveness of such shrinkwrap licenses, while providing guideposts useful to practitioners involved in the license drafting process. As noted in ProCD, it is essential that software programs and display screens provide notice that use of the software is subject to the restrictions contained within the software a shrinkwrap license agreement (ProCD v. Zeidenberg, 1996). Practically, the Internet allows a software owner to verify the assent of a licensee to the terms of a software license. Under such a system, downloading of a software program would only be permitted after a purchaser explicitly agrees to the terms of the license (i.e, select the "OK" prompt following a query "Do you accept the terms and conditions of the foregoing license?").

On the other hand, without such consent, the developer of a web site should not assume that all terms and conditions will be enforceable. One court has found that terms and conditions set forth on the bottom of a webpage, where a user would have to scroll down to read the terms, or where the user could access information without ever being aware of the terms, will not support a successful breach of contract claim (Ticketmaster v. Tickets.Com, 2000).

CONCLUSIONS

Software data rights are complex. In addition to the general complexities of intellectual property law, the application of data right concepts that were developed in the 18th and 19th centuries to software developed in the 21st century leads to confusion and ambiguity. Moreover, because of the growing importance of software, such data rights are constantly being adjusted both legislatively and judicially. Therefore, it is important that software engineers and owners consult legal counsel regarding data rights issues as they arise.

However, as this chapter demonstrates, data rights are too important to be left solely to attorneys. By integrating the data right concepts of copyrights, trade secrets, patents and trademarks into the development cycle for software, software can be created that contains unique expressions, confidential materials, patentable processes and valuable trademarks. Moreover, software engineers must understand the importance and techniques of corporate copyright and secrecy programs to fully protect software data rights, including data rights that are granted to employers, the government and the general public.

While this chapter will not provide all of the answers to questions that arise regarding such data rights, it is hoped that it has provided a thorough introduction to the field of data rights and a source, through both the text and the bibliography, for future research efforts in the software data rights.