

## **Licenses and Agreements: Responding to Potential Pitfalls**

**Kerry Henson**  
**University of Central Missouri**  
**Adrian and Margaret Harmon College of Business Administration**  
**Computer Information Systems Department**  
**Warrensburg, Missouri 64093**  
**660.422.2705**  
**henson@ucmo.edu**

### **Introduction**

One upon a time there were three bears: vendor bear, developer bear and third-party bear. Vendor bear sold developer bear software which he used to create a porridge recipe application. Third-party bear used developer bear's application to make a big batch of porridge to sell at his Porridge Emporium, but alas the porridge was too hot, destroying his customer base and eventually forcing third-party bear to sell remaining assets at rock bottom prices. Whom does third-party bear sue? Who pays the blonde-headed defense attorney? Unfortunately, as in the fairy-tale world, the answers are not obvious in academia.

Use of software is fundamental to any computer-related field of study. Whether in the form of word processors and other productivity tools, software development platforms and compilers, operating systems and systems utilities, or other applications, academic institutions install numerous software packages and are subject to the related licenses. Some software is licensed to support administrative tasks while others are licensed for academic use. Fortunately, many vendors differentiate between academic and administrative use and thus price accordingly. Academic pricing programs vary from no cost to a discounted price structure, or some combination of the two.

As it is for most organizations, license management, that is, obtaining the appropriate type and adequate number of licenses in the most economical manner, is challenging for academic institutions, but the variety of titles and types of licenses complicates the task for schools. Also, the pervasive attitude toward academic freedom and individual exploration motivates academicians to deploy software as needed. Trial licenses, freeware, shareware and open-source software are easily obtained and, unless tightly controlled, can be installed on institutional hardware.

While the primary focus of license management is matching licenses held with installed seats, embedded within these licenses are commitments accepted by the institution as the software is obtained or deployed. Many clauses in the endless legal morass of a license appear to be irrelevant to academic deployment, yet indeed the clauses are included for a specific reason, to protect the interests and identify the contractual obligations of the parties. This paper will explore the purpose of the software license and some of these clauses included therein, specifically indemnity and jurisdiction, as it relates to academic users. In addition, the issue of signature authority will also be examined.

### **Licensing**

## ***2009 ASCUE Proceedings***

Software exists as intellectual property and thus the rights can be assigned or licensed. Assigning gives all rights to another party while licensing gives the licensor the ability to grant specific rights (Classen 2007). Although the phrase “buying software” is not uncommon, in most cases such a transaction does not result in acquiring the product. If sold as a good, the purchaser can duplicate and redistribute the product. Thus, software is “purchased” through a license to use agreement. The producer can then extend control over product distribution through copyright protection and preserve a source of income.

The license is a contract between the software provider, or licensor, and the purchaser, or licensee. It delineates the rights and obligations of each party which are negotiated by the parties involved, each of which, depending upon the amount of leverage in the relationship, seeks to buttress its own interests. Allowed actions, limitations, remedies and responses to breaches of the license or applicable law are identified.

Proprietary licenses tend to be very specific in restricting use, copying and conveying the license to others. In contrast, open-source or “copyleft” licenses promote redistribution of the software itself while maintaining control over the use of the software. Free software licensing conveys virtually all rights to the licensee including the ability for the licensee to incorporate the free software into its own proprietary offerings.

### **Pertinent Statutes**

All contracts have a foundation in the law, but the contract’s purpose is to modify and extend the law to the contractual relationship. The underlying law in essence is the default agreement from which the license may deviate. For software, the underlying state laws are based upon the Uniform Commercial Code-Sales, or UCC Article 2, which addresses sales of goods (Cornell University Law School). It predates software distribution but nonetheless has become foundational to judicial regard for software licensing addressing relevant issues such as “contract formation, interpretation, performance, warranties, and remedies” (Landy 2008, 194). State laws also provide protection for trade secrets and are based upon the Uniform Trade Secret Act (UTSA).

The Uniform Computer Information Transactions Act (UCITA), proposed in 1999, specifically addresses software and digital products. It has been heavily criticized as favoring large software vendors over consumers, and it has not been widely embraced by state legislatures (AFFECT; Landy 2007).

### **Licensing Intellectual Property Rights**

Licensing limits the licensee’s use of the software under intellectual property rights. The licensor’s rights will be considered violated if the licensee exceeds the limitations in which case the license will specify the remedy to follow, but this also implies that the licensor holds rights it can grant (Classen 2008). Thus, the license serves as a demarcation between appropriate and inappropriate use of the software. A well-structured license will protect the interests of both parties and gives each the freedom to exercise within the specified bounds.

Intellectual property rights take the form of trademarks, copyrights, patents and trade secrets. Each of

## *2009 ASCUE Proceedings*

these forms protect a different aspect of the software product and can be applied “because software can be both a work of authorship as well as a business process” (Classen 2007, 11). Software exists as source code that can be printed or displayed, or as executable modules containing procedures and algorithms. It can also enforce adoption of specific procedures, e.g., ERP systems. Multiple protections are necessary for the diverse aspects of intellectual property, that is, no single protection is applicable to all aspects of the intellectual property.

### **Trademarks**

Trademarks protect branding property such as product names and slogans. (Gordon 2006). Some licensors create logos specifically for academic institutions and related trademark clauses may be applicable to the school’s use of logos in its own advertising. Aside from branding of the software, trademarks are not an important intellectual property issue.

### **Copyrights**

Copyright protects the expression of the software, in a printed or digital form, but it does not address the underlying concepts within the programs. It protects code when printed on a page, but not the processes or procedures represented by the code (Classen 2008). As an expression in a digital form, licensed, proprietary software can be used, but not duplicated and redistributed.

One caveat for copyrighted material is that it must be available to the public so other parties can avoid infringement. Applied to embedded processes, this would require the vendor to publish the source code or to allow reverse engineering (Gordon 2006). Thus, copyright protection, while useful in limiting duplication of the digital expression, is not an attractive licensing option for protecting the underlying processes.

### **Patents**

Conversely, a patent applies to processes, equipment and physical compositions as well as related extensions and improvements (Classen 2008). Thus, it protects the underlying concepts but not the expression thereof. Also unlike copyrights, one must apply for a patent and ensure that the concept is adequately different from all other patented processes. Patents restrict others from using the processes for twenty years following the first application filing. Like the copyright a patent requires the holder to reveal the protected process so that other parties can avoid infringement.

### **Trade Secrets**

Trade secrets, like patents, offer protection to underlying processes, but trade secrets do not require an application review and are not limited to twenty years. Protection offered is also broader than that of patent as it extends to “computer code . . . program architecture . . . information content including order, structure and sequence [and] algorithms” (Classen 2007, 13). Trade secrets are at risk in that the intellectual property can be discovered or obtained through reverse engineering and the property right disappear (Amjad 2002). Violation of trade secrets is the easiest infringement of intellectual property rights as the protected concept is not visible (Classen 2007).

## ***2009 ASCUE Proceedings***

### **Presentation of the License**

Software licenses are as ubiquitous as software. The shrink-wrap license, where the licensee indicates acceptance of a printed agreement by opening product packaging, is common for off-the-shelf titles. Licenses for software that is downloaded are often presented prior to downloading, requiring the licensee to click an acceptance button to initiate downloading files. In both cases, required acceptance is often reiterated during software installation through so-called “clickwrap” or “clickthrough” action. A product’s paper license and clickwrap license may be inconsistent so one should carefully examine both. Membership agreements such as those for Microsoft® MSDN® Academic Alliance, Oracle® Academy, IBM® Academic Initiative and Altova® Education Partnership may place additional restrictions on software use.

Licenses may not be printable nor available for review after installation making later review most difficult. An inconvenient review during the installation process may prompt the licensee to forego a careful reading, but this neglect can lead to unintended obligations.

### **Goals of Licensing**

Like any contract negotiation, parties entering into a license agreement wish to protect their interests. The licensee wants assurance the software meets its needs and that the product does not violate any third-party intellectual property rights prompting consequences for the licensee. Unfortunately, unlike customized software, shrink-wrapped and click-wrapped licenses are not submitted for negotiation. The licensee is given the option to accept the license or forgo use of the software.

Obviously the licensor wishes to protect its own interest. “The approach used in most vendor form agreements is to warrant title, provide a narrow indemnity for intellectual property infringement, and limit their liability to, at best, a fraction of the fees paid under the agreement” (Overly and Kalyvas 2004, 51). While these terms may appear fair, they, at a minimum, create disadvantages for the licensee and can be used to unfairly distribute risk to the licensee. Generally, license terms are binding, even if later considered onerous; thus, the licensee should consider the entire license, especially indemnity and jurisdiction clauses, that can be quite problematic.

### **Warranties and Indemnity**

Warranty is a separate but related concept to indemnity. Warranty clauses appear as statements that the software product meets certain criteria while the indemnity describes the remedy for specific breaches. Both seemingly offer guarantees about the product, but Tollen (2006) suggests

that warranty and indemnity is not about guarantees but about allocating risk between parties; thus, the licensee must recognize risks to which it is obligated and risks the licensor is unwilling to accept.

### **Warranty**

Warranties basically state that a representation is true. They may be explicitly stated (express) or be ap-

## *2009 ASCUE Proceedings*

plied by default (implied). Implied warranties of the UCC include merchantability, addressing quality and performance; fitness, addressing appropriateness of use; title, addressing rights to sell to the customer; and non-infringement, addressing intellectual property rights (Landy 2008). The licensor may deny implied warranties, but such a statement must be conspicuous in the license. Usually such a clause is printed in bold, upper-case letters.

Merchantability and fitness are two problematic issues for licensors. The inherent complexity of many software applications opens merchantability to various interpretations. Likewise, the fitness of the software for a particular use is assessed by the licensee; thus, licensors often include specific language to exclude implied warranties (Tollen 2006).

Other warranties may ensure that the software complies with applicable law, the licensor faces no pending litigation related to the product, that any third-party software incorporated into the product is identified, and that the licensor has the right to extend a license that includes such third-party programs (Overly and Kalyvas 2004). Warranties may guarantee the product does not contain viruses nor any disabling mechanisms. Additionally, the license may warranty that dates are processed correctly (Gordon 2006).

### **Indemnity**

In a license between two parties, indemnity's frequent purpose is to protect one party from negligence of the second party that results in claims brought by a third party. It is often used to limit liability when intellectual property rights are violated, but it can include other types of damage as well (Classen 2007). It can be specifically limited by the type of intellectual property, be it copyright, trade secret or patent, geographic region, date or the vendor's knowledge (Landy 2008).

Indemnity can extend from the licensee to the licensor or from the licensor to the licensee. For instance, if a software development firm incorporates another vendor's application into its own product and a competitor of the vendor claims infringement by the vendor, an indemnity clause could require the vendor to protect the software development firm. Likewise, if a competitor of the software development firm claimed infringement by the firm, an indemnity clause could require the firm to protect the vendor. In either case, the obligated party may incur the cost of defending itself and the second party; however, the inconsistencies in the goals of the two parties may vary such that the second party chooses to take up its own defense. Responsibility and limitations of this separate defense may be specified in the indemnity clause. For example, a licensee required to indemnify and defend the licensor against a claim of infringement is not necessarily interested in maintaining the licensor's claim to the intellectual property and may be willing to cede such in a settlement. It would be to the licensor's advantage to obtain its own counsel to defend its position (Classen 2007).

Indemnity can also extend obligation beyond what is otherwise applicable by law. It can be used to reverse responsibility or obligation to protect the other party against a claim (Adoranti 2006). This is especially dangerous for the negligent licensee. "Increasingly, licensors are seeking to limit their identity and other liability to third party claims arising from the licensor's gross negligence and even have the licensee indemnify the licensor for the licensor's own negligence" (Classen 2007, 56). This poses a potentially devastating scenario as the licensee's obligation is tied to actions and events outside the licensee's control.

## ***2009 ASCUE Proceedings***

Vendors are increasingly incorporating third-party software into their products, thus the third-party warranties and indemnities affecting the vendor can extend through the license to the licensee (Overly and Kalyvas 2004). The licensee should look for a warranty addressing third-party licenses to assess the associated obligations and risk.

While indemnity against an infringement claim is a major focus, consequences of software errors pose significant risk as well. Suppose a software consultant uses a major vendor's development platform to create a system for a business. Managers at the business use the software to support making a strategic decision only later to discover a program flaw. Although the error may be correctly attributed to the vendor or consultant, indemnity clauses may dictate who bears the risk.

### **Remedy**

The indemnity clause also specifies the extent of liability born by the responsible party. Generally it is the total amount suffered unless limited within the indemnity clause. This could include direct costs due to injury, lost opportunity costs, loss of intellectual property, loss of data and impact upon the reputation of the company (Adoranti 2006). Indirect damages can expand and seriously erode the responsible party's financial standing.

The licensor will seek to limit its monetary obligation to the licensee prompted by the licensor's own infringement. Remedy may be offered in various forms: the vendor modifying the software to make it compliant, the vendor obtaining rights from the third party bringing the claim enabling the licensee's continued use of the product, or the vendor refunding all or a portion of paid fees and terminating the agreement (Gordon 2006). In each case the licensee may face a work disruption as the vendor and third party seek resolution. Obviously terminating the agreement may pose serious consequences for the licensee, especially if business processes are tailored to the software. Indemnity clauses may omit a deadline for vendor action which again could disrupt the licensee's business. That said, it generally is to the third party's advantage to negotiate with the vendor in order to gain monetary benefit as a result of the breach and to expand its product's use (Gordon 2006). Still, the licensee accepts risk when the licensor's indemnity is limited.

Indemnification can dictate whether contributory or comparative negligence applies. Contributory negligence is a defense where, if the plaintiff is found to have contributed to the cause of injury, the defendant is free of obligation to pay any damages. In contrast, comparative negligence assesses damages based upon of the defendant's degree of participation in the cause. In common terms, contributory negligence is all-or-nothing while comparative negligence is partial credit.

### **Examples of Indemnity**

Indemnity clauses can be fairly simple as reflected in the following from a Macromedia® Dreamweaver® EULA:

*You agree to indemnify, hold harmless and defend Macromedia from and against*

## 2009 ASCUE Proceedings

*any loss, damage, claims or lawsuits, including attorneys' fees, that arise or result from the use or distribution of your application.*

In this case Macromedia is requiring that the licensee defend Macromedia against claims stemming from the licensee's use of the program. This may be considered fair. A second example from the Apple® QuickTime® 7 EULA is a bit more complex:

- 6.1 *Apple has no obligation to indemnify, defend or hold Licensee harmless from and against any claim that the Software licensed hereunder infringes any third party patent, copyright, trademark or other intellectual property right. Licensee will promptly notify Apple of any such claim.*
- 6.2 *To the extent permitted by applicable law, Licensee will indemnify, defend and hold Apple harmless from any and all claims, damages, losses liabilities, costs and expenses (including reasonable attorneys and other professionals) arising out of or in connection with Licensee's and its distributors' distribution of the Software, unless the claim arises solely out of the Software as originally provided by Apple to Licensee. The foregoing exception will not apply to a claim arising out of the combination of the Software with any other software or hardware. Apple will promptly notify Licensee of any such claim and will provide reasonable cooperation and assistance in connection with such claims.*

Here Apple specifically declines to shield the licensee in case of its own infringement but requires indemnification from the licensee related to the licensee's actions.

Indemnity clauses are frequent, but not universal across EULAs. A review of a variety of applications found indemnity clauses in licenses for Microsoft SQL Server®, Visual Studio® and Office, Mozilla®, MySQL®, NetBeans®, and Corel® WordPerfect®. They also appear during installation of seemingly innocuous software such as print drivers and media players that may be installed by students in labs.

### **Potential Impact of Indemnity**

Intellectual property indemnity claims are infrequent, but their effect can be severe. Overly and Kalyvas (2004) compare it to an earthquake, a very sporadic event with a devastating impact. "Added to the risk of infringement damages is the fact that typical, legal, and expert fees for patent litigation defense - from start of the lawsuit to the end of the trial - is two to three million dollars (not including appeals)" (Landy 2008).

## ***2009 ASCUE Proceedings***

For academic licensing, indemnity risk is not so evident. Obviously one should avoid taking on the licensor's risks unnecessarily. But an argument can be made that indemnity extends to student use. As part of a class project, a student, working with an outside organization, may create a "real-world" application using the institution's software then provide the application to the organization. If the application later fails, causing direct and consequential damage to the organization, who is at fault? While the proper use of academically licensed software may be questioned, legal action by the organization requires a defense and that cost is incurred, in accordance with the indemnity clause, no matter the ultimate findings. Once a suit is filed, expenses mount, no matter the veracity or resolution of the case.

Academic programs at state institutions may find state policies disallow indemnity clauses in agreements. Private schools may have similar policies. Thus, an indemnity clause can become a "deal breaker" especially given clickwrap licenses are not negotiated.

### **Jurisdiction**

Another clause that poses difficulty for academic institutions is jurisdiction. Licensors often limit software use to certain nations to avoid export restrictions and foreign court jurisdiction. Within the United States, licensors will also seek to restrict litigation to states and regions. This in turn, can limit indemnification to certain localities.

Generally both parties prefer to conduct court action in a state where they are headquartered or have a substantial presence. Travel to another state and obtaining counsel familiar with laws of that state may substantially increase litigation costs. This possibility grows in magnitude as the licensee obtains multiple license holdings tied to a variety of other states, as it does for the licensor as its customer base grows.

If no jurisdiction is specified, each party may well seek to file first to obtain leverage in the litigation. The case history of that state then influences the outcome. If neither party concedes this clause, a compromise is selecting neutral jurisdiction such as New York (Gordon 2006). In addition to increased expenses due to litigation at a distance, Classen (2007) and Landy (2008) cite differences in state laws that can affect the legal proceedings:

- Virginia and Maryland are the only states that adopted the aforementioned UCI-TA, considered favoring vendors.
- UCC is basis for state law in all states except Louisiana.
- Some, but not all, states recognize "exemplary" or "punitive" damages.
- In several states, under the UCC, a basis for implied indemnity exists when a third party incurs damages due to a product failing to meet warranties.
- Some states, including New York, New Jersey and Texas have not adopted the UTSA.
- Certain states uphold indemnity clauses, even when it appears the agreement unduly favors one party, if the indemnity is obvious and clearly stated in the license.
- States vary in interpretation of non disclosure agreements under trade secret law.
- Sovereign immunity, that is, the ability to sue the state, is embraced to varying degrees across states, an issue potentially affecting litigation against state institutions.

As described before, states also vary in guidelines for awarding the plaintiff's claim in full or in

part. While most states embrace comparative negligence in its pure or modified forms, Alabama, Maryland, North Carolina, Virginia and the District of Columbia maintain pure contributory negligence (Matthiesen, Wickert and Lehrer 2009).

### **Signature Authority**

Many institutions adopt a signature authority policy that restricts employees from committing the school to financial and legal agreements. In some cases, the individual can be held personally responsible for unauthorized commitments. Institutions may require all software licenses be reviewed by a central authority, e.g., Johns Hopkins University (2006). For a school of any size, such a policy, while strategically sound, appears operationally problematic. Substantial resources are required to examine clickthrough licenses attached to all software, including print drivers, presented only during installation.

### **An Academician's Response**

In conclusion, academicians should follow certain measures. The academic licensee has basically no leverage in negotiating an academic license as vendors often have no financial incentive to allocate attorneys to revising license agreements for nonpaying customers. Thus, balancing institutional protection with academic demands can pose a special challenge for faculty in a computing-related discipline and for lab coordinators. Ultimately, it is a decision assessing liability risk in comparison to academic need. There are several steps a faculty member can take toward a proper balance.

1. Read the license.
2. Cooperate with the signature authority to avoid personal risk.
3. Preemptively examine software licenses before related book adoptions or other commitments.
4. Document the academic need for each title installed.
5. Maintain a file of all licenses.

In addition, lab coordinators can take additional steps:

6. Enforce tight controls over software installations in labs.
7. Use a license management tool that maintains seat counts as well as jurisdiction information (Overly and Kalyvas 2004).
8. Communicate the risks of indemnity and jurisdiction to faculty and students.
9. Enlighten those who have signature authority concerning the proliferation of clickthrough licenses. If possible, obtain parameters within which one may install without further license review. If not possible, explore creating a fast-track review process for less-complex licenses.
10. Create and publish lab policies for student labs restricting use to academic purposes. Reenforce the policy with a clickthrough at logon.

If these steps are followed, faculty, students and administrators, working cooperatively, can gain a proper perspective of licensing risks and make wise decisions in weighing risk with academic need.

## **2009 ASCUE Proceedings**

### **Note**

*The author does not hold a license to practice law and has received no legal training. The reader should not consider this paper sound legal advice, but should instead consult legal counsel in making decisions regarding topics discussed here. In other words, NO WARRANTY NOR INDEMNITY IS EXPRESS OR IMPLIED.*

### **References**

Adoranti, Frank. 2006. *The Manager's Guide to Understanding Indemnity Clauses*. London: LES50NS Professional Publishing.

AFFECT. *What's Wrong With UCITA?* [http://www.ucita.com/what\\_problems.html](http://www.ucita.com/what_problems.html) (accessed March 10, 2009).

Amjad, Holly M. 2002. *Patent vs. Trade Secret: Look at Costs, Industry, Returns*. Kansas City Business Journal.  
<http://www.bizjournals.com/kansascity/stories/2002/02/04/smallb3.html> (accessed March 11, 2009).

Classen, H. Ward. 2007. *A Practical Guide to Software Licensing for Licensees and Licensors*, 2<sup>nd</sup> Ed. Chicago: American Bar Association.

Cornell University Law School. Uniform Commercial Code - Article 2 Sales. 2005. The American Law Institute and the National Conference of Commissioners on Uniform State Laws. <http://www.law.cornell.edu/ucc/2/2-102.html> (accessed March 10, 2009).

Gordon, Jeffrey I. 2006. *Software Licensing Handbook*. Raleigh, NC: Lulu.com.

Johns Hopkins University. 2006. *JHU Signature Authority Policy*. Johns Hopkins Medical Institutions. <http://ssc.jhmi.edu/supplychain/signature.html> (accessed March 12, 2009).

Landy, Gene K. 2008. *The IT/Digital Legal Companion: A Comprehensive Business Guide to Software, Internet, and IP Law*. Burlington, MA: Elsevier, Inc.

Matthiesen, Wickert and Lehrer. 2009. *Contributory Negligence / Comparative Fault Chart*. Matthiesen, Wickert & Lehrer, S.C.  
<http://www.mwl-law.com/PracticeAreas/Contributory-Neglegence.asp> (accessed March 10, 2009).

Overly, Michael and James R. Kalyvas. 2004. *Software Agreements Line by Line*. Aspatore, Inc.

Tollen, David W., Esq. 2006. *The Tech Contracts Pocket Guide*. Lincoln, NE: iUniverse.

### **Attributions**

Altova is a trademark of Altova GmbH.

Apple and QuickTime are registered trademarks of Apple Inc.

Corel and WordPerfect are trademarks or registered trademarks of Corel Corporation and/or its subsidiaries in Canada, the United States and/or other countries.

IBM is a trademark of International Business Machines Corporation in the United States, other countries, or both.

Macromedia and Dreamweaver are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries.

Microsoft, SQL Server, VisualStudio and MSDN are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Mozilla is a registered trademark of the Mozilla Foundation.

MySQL is a registered trademark of MySQL AB in the United States, the European Union and other countries.

NetBeans is a registered trademark of NetBeans Corporation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.