WESFACCA

INFORMATION TECHNOLOGY CONTRACTS WORKSHOP

October 3, 2002

Thomas M. Pitegoff
Leonard D. Andrew

Pitegoff Law Office
Ten Bank Street, Suite 540
White Plains, NY 10606
914-681-0100
www.pitlaw.com

CONTENTS

Introduction

A lawyer handling an Information Technology (IT) contract needs to have a basic understanding of the technology and the process by which it is implemented and maintained in the corporate setting. With this understanding, he or she can deal with questions such as: What business need is the client seeking to meet with the IT project? How well are the company's requirements articulated? Can the solution meet the need? How can the company measure whether the solution actually meets the requirements?

The second area of concern is ownership. What Intellectual Property (IP) is the client getting? Is it a license, or is it ownership of the intellectual property? What third party rights need to be considered? To what extent will the supplier maintain the software? To what extent can the client obtain access to the code for modifications if the supplier stops maintaining the software?

These two areas of concern, the IT and IP issues, both arise in the context of software development, warranties, infringement, indemnification, and license and maintenance agreements.

A third topic covered by this paper is confidentiality. Confidentiality includes trade secrecy as well as the protection of individually identifiable data of customers and employees. Confidentiality issues arise even before the IT project begins and can continue indefinitely thereafter. Both the corporate IT user and the supplier need the protection of a confidentiality agreement. For example, the supplier may have access to the corporation's sensitive data and the corporate customer may receive competitively sensitive confidential information about the supplier.

Finally, this paper introduces the emerging uniform law of computer contracts, the Uniform Computer Information Transactions Act or UCITA.

Reducing the High Rate of IT Project Failure

The causes of IT project failure

A surprising percentage of IT projects end in conflict or are never completed. The CPR Institute for Dispute Resolution (www.cpradr.org) cites statistics indicating that 50% of IT projects fail. Whether you're counsel for a corporate end user or for an IT company, your challenge is to reduce the risk of lengthy and expensive litigation. Project failures can be critical to the businesses of both end users and IT suppliers. Counsel's role in advising the client and in drafting and negotiating the IT agreement can be effective in substantially reducing these risks.

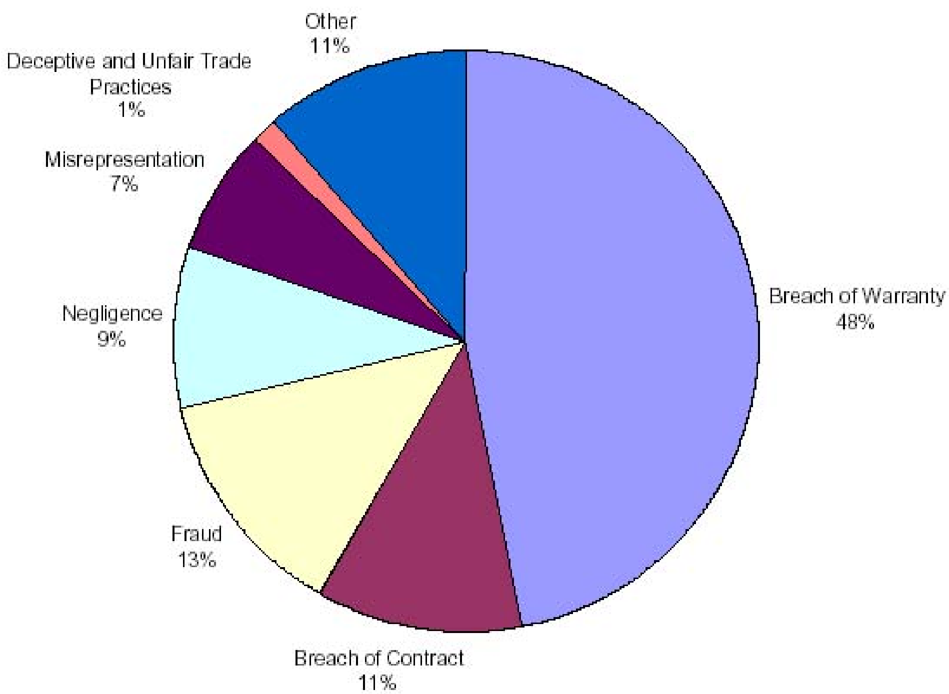
Failures can arise from many causes. In some cases, the company management has unrealistic expectations of what software can do. Sometimes this unrealistic expectation is created or reinforced by the unrealistic promises of the vendor. In other cases, the scope or requirements of the project may be inadequately defined by the corporation. Problems may arise because of a lack of communication between the customer and vendor during the course of the project. The project may be managed poorly by both parties. The vendor may make changes or perform other work outside of the original contract without signed work orders. The contract itself may not be signed or the work may be done without any contract at all.

PricewaterhouseCoopers LLP conducted an illuminating survey of reported litigation involving IT projects over a 25-year period (1976-2000). ("Patterns in Litigation: Systems Failure", available at www.pwcglobal.com) The survey identified 120 cases that dealt with some form of systems failure and found a small number of recurring patterns. The survey identified the following "core quality values" that need to be addressed with respect to IT projects:

- Reliability – minimal errors and downtime
- Performance – complete operations within acceptable timespans
- Functionality – offer the required features
- Compatibility – interact effectively with existing IT systems, including external systems
- Lifespan – must warrant the cost to the client, often with the ability to grow
- Deployment – within an acceptable timeframe
- Support – capability to be upgraded and repaired over time
- Cost – must be justified

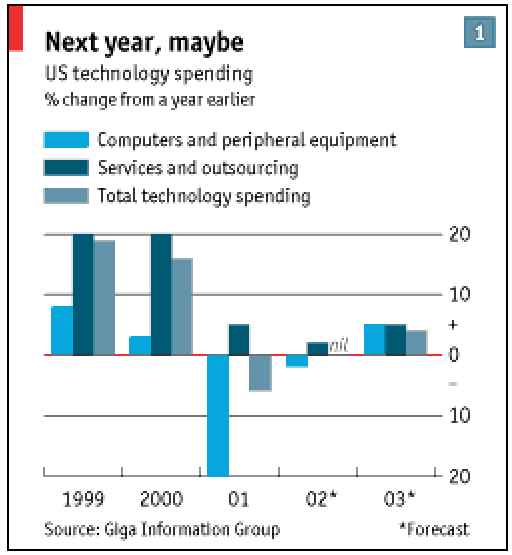
Failures occurred when one or more of these values were not acceptable to the client. In most of the cases, the computer systems either failed to work acceptably or never worked at all.

The following pie chart, taken from the PricewaterhouseCoopers survey, presents a breakdown of the legal claims made in these cases.



Breach of warranty and breach of contract together account for a majority of the claims. Other claims fall in the categories of misrepresentation and fraud. All of these types of disputes can be reduced by better communication between the parties at the time of contract drafting and throughout the course of the project.

The IT market: Bargaining power shifts from vendors to customers



Given the economic slowdown of the last two years, particularly in the IT sector, IT customers have more leverage now than ever. The balance of bargaining power is shifting to customers and away from vendors. This should make it easier for corporate user counsel to negotiate agreements more favorable to their clients. According to Giga Information Group, IT spending decreased by 6 percent last year and has remained flat this year, after rising almost 20 percent in 1999 and 16 percent in 2000. (*See* "Will the IT Doldrums Drag On?" B. Morrissey, Aug. 30, 2002, www.internetnews.com.) The chart to the right appeared in an article entitled "IT Grows Up", *The Economist*, August 24, 2002. The chart shows that spending on IT services and outsourcing is even lower this year than it was in 2001.

As *The Economist* reports, companies today are spending money on IT projects that help cut costs, improve security and integrate existing software applications. Rather than undertaking multi-million-dollar deals as in past years, companies today prefer small, short-term projects. IT investments today have to provide a clearly discernible return on investment. Companies are looking to link their various processes and data bases using open standards whenever possible. Gone are the days of inflated dot com promises. Today the emphasis is on integration and value, quality and reliability. Hopefully, this will result in more reliable software and fewer project failures.

The central role of the agreement

An IT project that entails software development, consulting or integration stands an excellent chance of failure, as noted above. It is very possible that the software will not perform as promised and that the customer's recourse may be limited. The PricewaterhouseCoopers survey notes that the most common type of IT dispute results from the purchase of systems that are defective. The vendor attempts to repair it, with limited and unsatisfactory success. In some cases, the client returns the system and acquires a new one from a different vendor.

According to a recent study by the National Institute of Standards and Technology, software bugs cost the U.S. economy nearly $60 billion each year. ("Software Errors Cost U.S. Economy $59.5 Billion Annually," June 28, 2002, www.nist.gov) In its press release, the NIST stated as follows:

> Software is error-ridden in part because of its growing complexity. The size of software products is no longer measured in thousands of lines of code, but in millions. Software developers already spend approximately 80 percent of development costs on identifying and correcting defects, and yet few products of any type other than software are shipped with such high levels of errors. Other factors contributing to quality problems include marketing strategies, limited liability by software vendors, and decreasing returns on test-

> ing and debugging, according to the study. At the core of these issues is difficulty in defining and measuring software quality.

Software that does not perform as promised can cost your company millions of dollars. While a company's counsel can do little, if anything, to reduce the incidence of software bugs, there are things that an attorney can do to assist his or her client in dealing with a process in which defects are the norm. Here are some recommendations contained in the PricewaterhouseCoopers survey:

> All parties to the IT project should agree ahead of time to specific expectations, promises, and contingencies regarding each of the areas of quality given above. For example, the system specifications should include not just the required functionality, but should also spell out any performance requirements or constraints, compatibility requirements, anticipated lifespan, and acceptable levels of defects.
>
> Both parties should also clearly and unambiguously define key terms, conditions, and activities such as the meaning of "beta testing" or the standards for judging whether the client has accepted the system. In the IT world, accepting a system can occur at many different times, such as when it has passed a series of agreed-upon tests ("acceptance testing") and has been in operation for a certain period of time with no serious defects appearing. If all parties are not willing or able to do so, that's a strong warning sign that a dispute may well emerge. However, chances are the exercise of creating such a document will in itself flush out potential problem areas well in advance of any signing, payment, or delivery.

In other words, in order to reduce the likelihood of failure, the contract should be as specific as possible. This requires the attorney working on the project to communicate in detail with the client and to endeavor to understand the project requirements.

The PricewaterhouseCoopers survey also states that IT disputes commonly arise when the vendor makes exaggerated promises or the client succumbs to wishful thinking, expecting more from a system than it can deliver. Miscommunication results in failure. According to PricewaterhouseCoopers, this type of issue

> can be avoided by making sure both sides agree upon a common, written set of definitions, specifications, and time tables with regards to the systems in question. As questions and issues arise, both sides can refer to and, if necessary, revise the document. This document should also go up and down the chain of command in both organizations as needed to make sure all relevant personnel understand what is promised and what is expected.

Here again, the attorney working on the project can do a great deal to facilitate a detailed meeting of the minds.

Put another way, the problem is one of mismatched expectations. The vendor looks to its defined contractual obligations while the customer seeks to solve business issues. To deal with this problem, the agreement should include conflict and change provisions. It might call for monthly meetings to review performance, problems and successes. This helps build personal relationships and helps the parties deal with issues before they become major problems.

Finally, although the balance of bargaining power may be shifting to customers, counsel for the customer should strive to reach a reasonable agreement, not an onerous one. Companies generally work with IT vendor for period of years. The customer usually needs the ongoing assistance of the vendor to maintain the software. Accordingly, customers will want to preserve good relationships with their vendors and not hamper their businesses. Counsel should endeavor to structure the client's IT projects in ways that benefit both parties.

Project management

As important as contract drafting is, good drafting alone will not prevent problems and disputes. The most perfectly written agreement can still result in litigation resulting from faulty administration. Legal counsel can play the additional role of asking management whether adequate administrative controls are in place to protect the company's interests. Legal counsel can work with management to create written guidelines for these administrative controls.

The larger the project, the more important it is to be sure that project management is handled properly. A typical project entails three interrelated considerations: requirements, cost and schedule. The key to success is setting and meeting customer expectations. This can be accomplished by joint staff meetings between vendor and customer; the designation of a joint steering committee; identifying the individuals on both sides who will have responsibility for the project; continual dialog and open discussion of problems; keeping the contract up-to-date with an effective change-control process; and requiring the vendor to provide early and frequent progress reports in order to minimize the potential for surprises.

According to project management experts, most project troubles are caused by:

- making dumb commitments (*e.g.,* performance guarantees, unachievable schedules, fixed-price contracts without the required analysis)
- not establishing a firm contractual baseline (*i.e.,* unclear requirements, terms and conditions and statements of work)
- not managing the customer relationship (*i.e.,* the working relationship must be continually enhanced and problems must not be hidden)
- underestimating (*e.g.,* labor time, costs, risks)
- having an unsupportive customer (changing personnel or stakeholders)
- not managing change (*i.e.,* no formal change management process)
- not staffing a project team adequately (inadequate skills, changes during the project)
- not managing the project (poor planning, poor communications, lack of discipline)
- starting development before the design is complete
- not managing subcontractors (poor selection of subcontractors or lack of coordination)

Everything should be documented in writing, from change requests to problems. ("IT Malpractice," Ann Bednarz, *Network World*, April 8, 2002, www.nwfusion.com.) The agreement should include procedures for quality reviews, testing, measurement of progress, defect management, upgrades and much more. The customer should consider its needs with respect to reliability, performance, functionality, compatibility, lifespan, support and cost. When problems arise, the customer should act quickly. Sometimes it's better to accept a loss and terminate a project rather than to allow it to escalate into a far more expensive failure.

Fees and the changing nature of IT agreements

While the obligations of the vendor to the customer may be complex, as discussed below, the primary obligation of the customer is simple. The customer pays the vendor fees for its services or its license.

Development agreements have traditionally been large projects, commonly calling for progress payments. A customer can manage the risk of these projects to some extent by paying in increments based on project milestones, or holding back a portion of the fee until the software is deemed operational. However, these approaches may pose a problem for the vendor. Current accounting rules do not permit a software company to record a sale until all contingencies have expired.

This may be one of the reasons that companies are increasingly breaking projects into smaller chunks, covering shorter periods of time. (*See* "Will the IT Doldrums Drag On?" B. Morrissey, Aug. 30, 2002, www.internetnews.com.) Another approach might be to pay a greater amount for the vendor's services in implementing the software, rather than for the software license itself. "That helps protect the customer against quality problems because more payment is tied to an ongoing business relationship, but it also lets the vendor recognize some revenue from the non-contingent sales part of the agreement." ("Quality Counts," *Information Week*, Aug. 12, 2002, www.informationweek.com.)

License agreements may call for one-time payments or recurring payments, depending on whether the software is viewed as "rented" or "purchased". Maintenance fees are recurring, and may include additional hourly charges. Licensors typically charge 18% or more of the product price each year for maintenance.

IT agreements commonly list the amount of fees and the manner of payment in a schedule to the agreement. The agreement might state that the amounts set forth in the schedule will be effective for the first year after the agreement is signed.

The customer may want better price protection than one year. For example, the agreement might provide that the prices and rates in the schedule will not increase for thirty-six months following the date that the customer implements the software.

Thereafter, the vendor will want the right to increase its prices and rates. The customer should at least ask for adequate notice of any price increase so that the customer can replace the vendor if it finds the price to be too high.

The customer may also want to include a general clause along the following lines:

> For all services that Vendor performs for Customer, Vendor will charge only such amounts as are reasonable and customary. Vendor will not charge Customer more for any such service than Vendor's standard charges for similar services for other customers.

The first sentence above calls for fair prices. The customer may want to obtain prices from other vendors to be sure that the prices are fair. The second sentence is a most favored nation pricing clause. This is very good from the customer's point of view. Practically speaking, only those customers that are in the best bargaining position will be able to obtain such a clause. In order to enforce this clause, a customer would want the right to inspect the vendor's books and records.

An easier clause to negotiate might be one that gives a customer the benefit of any better offer the vendor may make to an affiliate of the customer. Here is a clause along these lines:

> Licensor agrees that if any offer is made to or agreement entered into by Licensor with any Affiliate for substantially similar programs in substantially similar volume at a price less than the price to Licensee reflected in this Agreement, then the price reflected in this Agreement shall be reduced to such price offered to such Affiliate.

Models for IT contracts are continually evolving. Licensors, in particular, are looking for ways to increase their revenues and secure recurring revenue streams rather than one-time license fees. Microsoft, for example, has issued a major change in its licensing policies. The changes in the Microsoft approach affect most PC users. Microsoft's operating systems (Windows 95/98/2000/NT/XP) and applications (Word, Excel, Power Point, etc.) are used on 90% of PCs.

Since August 1, 2002, Microsoft no longer offers discounted prices for new versions to registered users of their products. Instead, users are required either to buy the complete new license at full price or to buy an "Upgrade License and Software Assurance" plan for a particular Microsoft product. (*See* www.microsoft.com/licensing/default.asp) The Software Assurance plan allows the user to install any new release of the licensed product that becomes available during the covered period. Microsoft offers a "Volume Licensing" program, generally applicable to five or more PCs. Microsoft Enterprise Agreement 6.0 is a software volume licensing program designed for corporate customers with 250 or more desktops. This three-year agreement provides discounts compared to full retail, with a fixed, annual price based on the number of eligible desktops in your enterprise for "Software Assurance" coverage for all enrolled products.

If your company's use of Microsoft products is limited to operating systems, it may be unnecessary to obtain "Software Assurance." PCs always come with the latest operating systems, so upgrades will be obtained as new PCs are installed.

However, for companies that plan to upgrade a significant number of Microsoft applications, "Software Assurance" is now the only practical option. Unfortunately, this new approach has certain disadvantages for customers. First, there is no assurance that there will be upgrades to your particular applications during the "Software Assurance" period, or that what upgrades become available will significantly enhance the product. Second, there is no assurance that Microsoft will not announce "end of life" of your version of its software, providing no support or upgrades and raising the possibility of compatibility issues.

Microsoft is able to exercise a high degree of control because the founders had the foresight to retain ownership of the software from the outset. Obviously, if a development agreement calls for ownership of the code by the customer, the price will be far greater, and the agreement will not be a license.

Ownership: A Review of Basic IP Concepts Relating to IT

While software performance is one area in which disputes may arise, it is not the only area for potential disputes in IT contracts. Disputes can also arise over ownership issues. The ownership of the software should be clear from the language of the contract. In order to approach the ownership question properly, the drafting or reviewing attorney needs to understand the intellectual property aspects of IT.

Intellectual property law covers four distinct but overlapping subjects: trademarks, patents, copyrights and trade secrets. A patent provides a legally-protected monopoly in an invention. Copyrights are legal rights in creative works. A trade secret is confidential information that has business value. A trademark is a brand name or logo that identifies the source of goods or services. While trademarks are important to owners of proprietary software, we will not deal with them here because they do not affect the core concepts in IT contracts.

Patents

Patents protect rights in inventions. U.S. law grants the inventor a monopoly on the rights to the invention for a specified number of years. This period was once seventeen years in the U.S., but is now twenty years. This limited-term monopoly is intended to encourage inventors to file for patent protection, thereby publishing their inventions and promoting further invention. Patents are published once they are issued. The fact that patents are published distinguishes them from trade secrets, which also protect the underlying invention.

In order to be patentable, an invention must be non-obvious and it must have a useful purpose. Laws of nature, such as mathematical formulae, are not patentable. For this reason, it was once not clear whether computer source code could be patented. Today, it is settled that source code is patentable. Nevertheless, most software is not patented. When it is patented, it is often the business method rather than the software itself that is patented.

An invention is not protected by patent law in the U.S. unless the U.S. Patent and Trademark Office issues a patent. The application process is lengthy and expensive, especially if the inventor is seeking patent protection in multiple countries. Enforcement of patent rights is also expensive. In addition, once the twenty-year patent term expires, patent protection is ended and the invention is available for all to use. For these reasons, patents make sense for inventions with broad appeal that will be commercially viable for years.

Copyrights

Copyright law protects original, creative expression in the tangible form of books, films, music and other works of art. In the commercial area, copyrights protect advertising materials, trade publications, label designs, operations manuals and photographs. Copyrights also protect computer software and databases. Works are protected from the moment they are created in tangible form, regardless of whether the author files with the U.S. Copyright Office or places a copyright notice on the work. Although federal copyright registration is not a prerequisite to copyright protection in the U.S., registration is necessary before the copyright owner can bring a lawsuit for infringement. In practice, most software is seldom registered in the U.S. Copyright Office.

Under Section 106 of the Copyright Act, the owner of a copyright has the exclusive rights to do the following:

- Make copies of the work.
- Prepare derivative works based on the work.
- Distribute copies of the work publicly.
- Perform the work publicly.
- Display the work publicly.

Copyright law protects the expression, not the idea. In this sense, copyright law differs from patent and trade secret law.

Copyrights in the U.S. generally last for the life of the author plus seventy years. When a work is created by an employee in the course of his or her employment, it is generally considered to be a work made "for hire". Copyright protection for a work made for hire lasts ninety-five years from publication or one hundred twenty years from creation, whichever is shorter.

Trade secrets

Trade secrecy law protects confidential information that has commercial value. Unlike copyright law, trade secrecy law protects the ideas themselves. Protectable trade secrets may include inventions, whether patentable or not. It may also include sensitive competitive information, such as customer lists, methods of doing business, financial information and the like. The Uniform Trade Secrets Act, adopted in about 40 states, defines a "trade secret" (in Section 45) as:

> information, including a formula, pattern, compilation, program, device, method, technique, or process, that: (1) Derives independent economic value, actual or potential, from not being generally known to the public or other persons who can obtain economic value from its disclosure or use; and (2) Is the subject of efforts that are reasonable under the circumstances to maintain secrecy.

One important advantage of maintaining information as a trade secret is that a trade secret can last as long as the information remains a secret. The formula for Coca Cola, for example, has remained a secret for more than a century. Once the information goes into the public domain, however, trade secrecy protection is lost forever.

Trade secrets are protected under state laws in the U.S. as long as the owner of the secrets takes the appropriate steps to protect their secrecy. This means, for example, prominently marking all secret information as trade secrets. Companies that have confidential information they wish to protect must take steps to safeguard that information. This is an ongoing effort. The owner of the information should prominently mark it as secret or confidential, disclose it only to the extent necessary, and obtain confidentiality agreements where appropriate. Entering into appropriate confidentiality agreements is another step that owners of trades secrets can take to protect these secrets.

IP protection of software

Computer software consists of a set of instructions written by a human programmer. It is written in a computer "language", such as C++, Basic, Cobol or the like. (These languages are themselves computer programs.) The resulting set of instructions is called "source code". Once the source code is written, a programmer "compiles" it, using a compiler program, into binary or "object code" so that it can run on computers. Object code appears as strings of ones and zeros.

Computer software can be protected in the U.S. by trade secrecy, copyright and patent law. Each type of protection involves different considerations. Generally, source code is protected by trade secrecy and copyright law, while object code is protected by copyright law. Some software is also protected by patent.

Although computer software is patentable, most software is not patented. Because of the lengthy patent process, computer software may be obsolete by the time a patent issues. Moreover, the developer may prefer to preserve the code as a trade secret forever. Finally, most software is not sufficiently novel to rise to the level of a patentable invention.

Copyright law is particularly important in light of the Internet, where copying and distribution is *very* easy. However, copyright law is of limited use for computer source code. While computer code is protectable by copyright law, that law does not stop another from independently creating new code that functions in the same way. Copyright law does not protect the ideas and functional elements of software, as does patent law. It only protects expression. For this reason, although copyright protection is inexpensive and easy to obtain, copyright protection is far weaker than patent protection. Independent creation is a complete defense to copyright infringement.

Trade secrecy is the most common method of protecting computer source code. The source code of proprietary software distributed under a typical software license remains secret. Software companies strive to maintain their competitive edge by keeping the source code out of the hands of potential competitors. Even if the source code copyright is registered with the U.S. Copyright Office, only a small portion of the code need be filed upon registration, while most of it typically remains secret.

Appropriate contracts are also crucial to a successful program of developing, exploiting and protecting rights in computer software. This is true both in the development of the software or creative work and in its licensing or sale. The remainder of this paper deals with computer contract issues.

Drafting to protect ownership of IP rights

When one party creates software for another, one basic question is who will own the software. When the creator of the software is to own it, the agreement can take the form of a license agreement with development features. When the customer is to own it, the agreement takes on characteristics of a development agreement or a consulting agreement.

When a work is created, the copyright immediately becomes the property of the person who created it. A limited number of exceptions to this rule fall within the definition of "works made for hire" under Section 101 of the Copyright Act. One type of work made for hire is a work created by an employee within the scope of his or her employment. When a work falls within this category, the employer owns the copyright rather than the employee. The other type of work made for hire is work made by an independent contractor, but only if the parties expressly agree in a written instrument signed by them that the work will be considered a work made for hire, and only if the work falls within one of the following nine statutory categories:

- a contribution to a collective work
- part of a motion picture or other audiovisual work
- a translation
- a supplementary work
- a compilation
- an instructional text
- a test
- answer material for a test
- an atlas

Although computer programming is highly unlikely to fall within one of these nine categories, IT agreements commonly contain provisions stating that the software will be deemed to be work made for hire, and that all rights will be owned by the customer rather than the vendor.

For this reason, the same clause should continue to state that, to the extent the work is not a "work made for hire" under copyright law, the vendor assigns ownership to the customer and will sign any documents reasonably requested by the customer to evidence this ownership. In some cases, the customer will ask the creator of the work to sign a written assignment of the copyright after the work is created. In fact, the customer may ask for ownership of all of the working product leading up to the finished product. Each time the software is updated or a version is prepared, the customer should get a new assignment. Written assignments are also important in order to protect copyright ownership outside the U.S. because the work made for hire rule is a creature of U.S. law.

Consulting companies that develop software for their customers frequently use independent contractors themselves. The consulting company has the same concerns with respect to the ownership of the software that the customer has. In other words, the consulting company should have agreements and appropriate assignments from its own independent contractors. The customer will want assurance from the consultant that the consultant has taken the appropriate steps to protect its own ownership rights, regardless of whether the customer ultimately owns or licenses the software.

In some cases, third party software is required as part of the project. In such cases, the customer needs a license from the third party. Assignment of such software is unlikely. The contract should require one party or the other to obtain a license to this software and install it. Your client should also know the cost of such third party software in advance.

While a clause specifying who owns the software is usually not the first paragraph in a consulting or development agreement, it is a starting point from a conceptual point of view. Here is a sample IP ownership section that might appear in a development agreement:

> (a) The parties intend that Developer's work product shall be deemed a "work made for hire" within the meaning of the copyright laws of the United States and any similar laws of other jurisdictions. Accordingly, all products of Developer's work hereunder will become the property of Customer alone, without further compensation to Developer, and Customer will be the sole owner of all copyrights in such work, even if this Agreement is terminated pursuant to Section __. To the extent that the product of any of Developer's work hereunder is not work made for hire, Developer hereby irrevocably assigns to Customer, and agrees that Customer shall be the sole and exclusive owner of, all right, title and interest in and to such work product, including all patent, copyright, trade secret and other proprietary rights therein, free and clear of any liens, claims or other encumbrances, to the fullest extent permitted by law.
>
> (b) Developer agrees to cooperate with Customer and to sign all papers and documents that Customer may reasonably request in order to evidence the ownership of Customer described in Section __ (a) or effect and evidence the assignment described in the last sentence thereof. Customer may withhold any payment until Customer receives any such requested signed paper or document.

When ownership is especially important to the customer and several programmers are working for the consultant, especially independent contractors, the customer may want more protection than that afforded by paragraph (b) above. Here is another clause that performs the same function as paragraph (b) above in a more far-reaching way:

> Consultant will furnish to Client, upon Client's request and at no additional cost to Client, an agreement, signed by each contributor to the work product stating that (a) each contributor has read the terms of the grant and confirmation of proprietary rights in the Works in this Agreement; (b) each contributor acknowledges that Client was induced to enter into this Agreement based upon Consultant's representations that it possesses the entitlement to convey to Client or to confirm, fully and effectively, the entire and exclusive copyright ownership of the Works in Client, and (c) each contributor has or will grant or confirm by additional documents such rights in favor of Client in the event Consultant fails to do so, without first or concurrently proceeding against Consultant. Consultant will execute all documents

> and perform all acts that Client may reasonably request in order to assist Client in perfecting its rights in and to the Works anywhere in the world, and Client will reimburse Consultant for any expenses reasonably incurred by Consultant in so doing.

Consultants and developers often have preexisting rights to software that they install for your client. The consultant might use software that the consultant created earlier for other customers. In most cases, the parties will have no intention of assigning such preexisting rights to the customer. Such software should be carved out from any assignment of software ownership. Here is a sample paragraph covering this subject:

> To the extent that Developer has any patent, copyright, trademark, trade secret or other intellectual property right which pre-existed the date of this Agreement and which relates or pertains to any part of the Software, Developer hereby grants to Customer an irrevocable, non-exclusive, worldwide, royalty-free license: (i) to make, use, market, sell, sublicense, execute, reproduce, display and distribute copies of and prepare derivative works based upon any such rights and (ii) to authorize others to do all of the foregoing.

Depending on the context, the consultant may not want to grant such a broad license to preexisting software, but may prefer to limit the license to use only internally by the company as a component of the developed software, while the consultant retains all other rights.

Who owns the software if the customer terminates the agreement before the software is completed? This can be a very important question. If the customer is to receive a full refund, it may be difficult to say that the rights to the software remain with the customer who paid nothing. Typically, the vendor receives some amount of compensation for work completed. However, even where some compensation is paid, without specific language to the effect that the ownership remains with the customer, the vendor may be able to argue that ownership did not vest in the customer if the customer rejected the software. Here is a sample clause that would ensure ownership by the customer:

> If Customer rejects the completed Software and terminates this Agreement pursuant to Section __, Vendor shall refund one half of all payments that Customer previously made to Vendor under this Agreement. Notwithstanding such termination, Customer shall own all rights in the Software pursuant to Section __.

In other cases, the vendor's contract may specify that the customer does not own the software until the customer pays for it in full.

Confidentiality Agreements

Parties to IT contracts frequently enter into confidentiality agreements before a project agreement is signed. The agreement may be in the form of a negotiation letter agreement which states that all discussions are preliminary and the neither party is obligated to proceed with any project unless and until the parties sign a written agreement to that effect. It would be appropriate to include a confidentiality obligation in such letter agreement.

Any consultant who has access to sensitive corporate data must agree to treat that data as confidential, whether it is personally identifiable employee or customer data, customer lists, marketing plans, nonpublic financial information, trade secrets or the like.

In many cases, a purchaser of IT services may need to disclose some of this information to an IT vendor before the contract for the project has been signed. Likewise, the IT vendor may have proprietary information that it seeks to protect through such a preliminary agreement. In these cases, it makes sense to sign a confidentiality or nondisclosure agreement before serious discussions begin.

Later on, the project agreement itself typically also contains a confidentiality obligation. A better way to approach confidentiality, however, might be to refer in the project agreement to a separate confidentiality agreement, which will remain in effect regardless of the number of project agreements signed and regardless of their expiration or termination.

Confidentiality protection requires more than just a good written agreement. When the company needs to protect trade secrets, employees need to be educated not to make unnecessary disclosures. Where the agreement calls for marking or otherwise identifying information as confidential, employees must be sure to so identify the information as confidential. It is also important to keep complete and accurate records of who has access to the information and how it is used.

Confidentiality agreements are so common that companies frequently use standard agreements and boilerplate language without thought. This can result in an overly restrictive agreement or one that lacks necessary protections.

One way or mutual?

Many confidentiality agreements are intended to protect the IT purchaser or licensee. In other cases, the IT vendor may want to maintain the confidentiality of its prices or the details of its products. The first question to consider is whether your client needs a confidentiality agreement before a project agreement is signed. If only one party requires confidentiality, then the other party should review that agreement very carefully. Obviously, if the agreement is to mutual, neither side can ask for more restrictions than it is willing to accept for itself.

Sometimes, the best approach is to advise your client not to disclose confidential information. That way, the confidentiality obligations can be one-sided, making them easier to negotiate, while your client need not worry about the disclosure of information by the other party who has not received the information. In many cases, however, it is necessary to disclose confidential information in order for the other party to evaluate the potential project.

In other cases, your client may not want to receive any confidential information from the other side, and the preliminary agreement may specifically state that the other party will not provide any confidential information. This protects your own company from liability for disclosure, because the contract establishes the fact that no information received is confidential. Aside from the risk that your client may be in breach of contract for disclosing trade secrets wrongfully, such disclosure carries potential criminal penalties under the Economic Espionage Act of 1996 (18 U.S.C. Sec. 1831 et seq.)

A corporate recipient of confidential information from a consultant may want to include an obligation to the effect that the consultant will not use or disclose the trade secrets of any other party. The risk is that by doing so, the consultant may breach its obligation to others.

What's covered?

What is the scope of the information that the agreement should protect against unauthorized disclosure? If only one party is seeking to protect confidential information, then it may draft a contract with a very broad definition of what is confidential. On the other hand, it may be in the interest of one or both of the parties to limit the definition of "confidential information".

What type of information might the agreement cover? One type of information is trade secrets. In order for information to qualify as a trade secret, a company must take reasonable efforts to maintain secrecy. Confidentiality agreements constitute one method of maintaining that secrecy.

Confidentiality agreements can also cover information that may not qualify as a "trade secret" because it may not have competitive economic value. One example might be personally identifiable employee data.

Here is one approach to a contractual definition of "confidential information":

> As used herein, the term "Confidential Information" of Customer means all information that Vendor may receive from the Customer, its employees, agents or representatives, prior to or on or after the date hereof, which is not generally available to the public, including but not limited to customer lists, proposed or planned products or services, marketing plans, financial and accounting records, cost and profit figures, forecasts and projections.

The list of examples can be much longer.

Typically, the recipient acknowledges that the defined information is owned by the discloser. A better approach from the recipient's point of view is to limit such a clause by simply acknowledging the discloser's claim of ownership rather than acknowledging in advance that it is proprietary information of the discloser. The recipient may want to argue, in the event of an alleged breach, the information was not proprietary to the discloser.

What's the disclosure period?

Another point that should be considered in drafting or reviewing a confidentiality agreement is the time period covered by the agreement. The question of how long the recipient of the information is required to hold the information in confidence is a separate question, discussed below. Here, we refer to the period of time for the making of disclosures that are within the scope of the confidentiality agreement. Here's an example of an expansive clause in a mutual confidentiality agreement, which contains a specific definition of "Disclosure Period":

> As used herein, the term "Confidential Information" means all information that either party or its affiliate (the "Recipient") may receive from the other party or its affiliate, or the employee, agent or representative of the other party or its affiliate (the "Discloser"), during the Disclosure Period (as defined below), that is not generally available to the public, including but not limited to prices, royalties and fees and their structures, the technical specifications of any software, computer source code and object code, inventions, customer lists, proposed or planned products or services, marketing plans, financial and accounting records, cost and profit figures, forecasts and projections and credit information. The term "Disclosure Period" means the period beginning on the effective date noted above (the "Effective Date") and continuing for one year and so long thereafter as the parties continue to discuss a Possible Agreement.

IBM customarily enters into two confidentiality agreements at the outset of discussions with a potential customer. The first is a framework agreement, and the second is a supplement describing the particular information to be disclosed. Here is an except from an IBM framework agreement, which also refers to a disclosure period:

> Each time Discloser wishes to disclose specific Information to Recipient, or wishes to engage in multiple disclosures relating to a specific subject matter, Discloser will issue a supplement to this Agreement ("Supplement") before disclosure. The Supplement will contain initial and final disclosure dates, a non-confidential description of the Information to be disclosed and any additional or different terms and conditions. The Supplement must be signed by the Discloser and the Recipient.

Identifying confidential information

If you want the agreement to be more restrictive, you can require that each item that is disclosed be specifically identified as being "confidential" in order to be within the scope of the agreement. Here is one example of such a clause:

> If the Confidential Information is embodied in tangible material (including without limitation, software, hardware, drawings, graphs, charts, disks, tapes, prototypes and samples), it shall be labeled as "Confidential" or bear a similar legend. If the Confidential Information is disclosed orally or visually, it shall be identified as such at the time of disclosure, and be confirmed in a writing within thirty days of such disclosure, referencing the place and date of oral or visual disclosure and the names of the employees of the receiving party to whom such oral or visual disclosure was made, and including therein a brief description of the Confidential Information disclosed.

In some cases, it makes sense specifically to include as confidential information oral and written information that an objective observer would consider confidential taking into account the surrounding circumstances. In other words, did the recipient have reason to believe the information he or she sees (rather than information actively supplied to him or her) might be confidential?

What's the extent of the nondisclosure obligation?

The core of any confidentiality agreement is the clause that obligates the receiving party to treat the received information as confidential. This clause can be drafted in any number of ways. Here's an example of an expansive confidentiality provision:

> Except as set forth herein or as otherwise agreed by the parties in writing, each Recipient shall at all times, both during and after the Disclosure Period, (i) not disclose any Confidential Information of the other party or its affiliate to any person other than the Recipient's employees or representatives who need to know such information; (ii) use the same care and discretion to avoid disclosure as the Recipient uses with respect to its own confidential information; (iii) not use any Confidential Information in the Recipient's business, nor develop, market, license or sell any product, process or service based on any Confidential Information; and (iv) not modify, reverse engineer or create derivative works based on any computer code owned by the other party or its affiliate.

This clause includes both a nondisclosure and nonuse obligation. It specifies a level of care that the recipient uses with respect to its own confidential information, and it restricts the recipient from reverse engineering the disclosing company's software or creating derivative works. Some confidentiality agreements go as far as to include noncompete requirements, although their enforceability may be questionable.

A variation on the above clause might include, for example, an absolute obligation not to disclose, rather than an obligation to exercise a defined level of care to avoid disclosure. Depending on the importance of the information, you might consider including a detailed security requirements addendum. Another way of limiting the use of the information would be to say that the recipient may "use the information only for the purpose for which it was disclosed, or otherwise solely for the benefit of the Discloser."

The clause above also restricts the range of parties to whom the recipient may disclose to the recipient's employees or representatives who need to know such information. Some agreements add that any such employees or representatives must be under a similar obligation of confidentiality. At a minimum, the recipient should be obligated to inform such recipients of their obligation to retain the information in confidence. Here is a sample clause:

> "Each Recipient will ensure that its employees, agents and representatives also comply with the Recipient's obligations of confidentiality and non-use under this Agreement."

Consider also whether it is appropriate to refer specifically to subsidiaries and other third parties in the context of a particular agreement.

Exceptions to coverage

Certain information is typically excepted from the coverage of a confidentiality agreement. Here is a typical "exception" clause:

> The obligations of confidentiality and non-use described above will not apply to information that (i) was already rightfully known to the Recipient on a non-confidential basis before the Effective Date; (ii) was independently developed by the Recipient; or (iii) is publicly available when received, or thereafter becomes publicly available through no fault of the Recipient or its employees, agents or representatives.

Other exceptions might include information obtained from a third party without obligation of confidentiality, or information disclosed by the discloser without obligation of confidentiality. The following "exception" language, taken from an IBM agreement, excepts information disclosed by the discloser without a nondisclosure obligation:

> Notwithstanding the foregoing, no obligation will apply to Information that is: (i) already rightfully in the Recipient's possession or rightfully received by the Recipient without a nondisclosure obligation; (ii) developed independently by the Recipient; (iii) publicly available when received, or thereafter becomes publicly available through no fault of the Recipient; (iv) disclosed by the Discloser without a signed Supplement as required by Section 1; or (v) disclosed by Discloser to a third party without a nondisclosure obligation.

A service provider or vendor might also want an exception for "residual information." The vendor's programmers will inevitably learn through the work they perform for the customer, and it would be impossible to prevent them from using the skills they learned. The vendor will not want to be liable for breach of contract or violation of the Economic Espionage Act as a result of the vendor's use of such residual information. A clause along these lines might read as follows:

> "The recipient may disclose, publish, disseminate, and use the ideas, concepts, know-how and techniques, related to the Recipient's business activities, which are contained in the Discloser's information and retained in the memories of Recipient's employees who have had access to the information pursuant to this Agreement ("Residual Information"). Nothing contained in this Section gives Recipient the right to disclose, publish or disseminate, except as set forth elsewhere in this Agreement: 1) the source of Residual Information; 2) any financial, statistical or personnel data of the Discloser; or 3) the business plans of Discloser."

Such a clause allows the employees of the vendor to work with competitors of the customer. It is also not a bad idea as a general rule from the customer's point of view, because the customer potentially receives the benefit of residual information that the vendor received from other customers. Here is another sample "residual information" clause:

> "Recipient understands and agrees that the employees of Discloser, during the course of performing services for Recipient, may further develop their experience, knowledge and skills as it relates to the business of Discloser. The subsequent use by such employees of such experience, knowledge and skills in the ordinary course of business of the Discloser does not constitute breach of this Agreement. Further, Recipient recognizes that receipt of Confidential Information under this Agreement shall not create any obligation in any way limiting or restricting the assignment of Discloser's employees."

Disclosures required by law

A party that is bound by a confidentiality agreement may find itself subject to a court order or a subpoena to disclose information that such party is contractually obligated not to disclose. Many confidentiality agreements specifically deal with this situation by requiring notice to the discloser and an opportunity to object or seek a protective order. Here is a sample clause with a slightly different approach to this issue:

> In the event that Recipient becomes legally compelled to disclose any Confidential Information, Recipient shall (i) promptly notify the Discloser that such information is required to be disclosed, (ii) use Recipient's best efforts to obtain legally binding assurance that all those who receive disclosure of such information are bound by an obligation of confidentiality, and (iii) disclose only that portion of the Confidential Information that Recipient's legal counsel advises is legally required to be disclosed.

How long does the obligation of confidentiality last?

Many corporate recipients of confidential information from suppliers seek to limit their obligation not to disclose such information. One way to do this is to limit the term of the confidentiality obligation. Some agreements, for example, require the recipient of confidential information to regard the information as confidential for a period of one, two or three years.

On the other hand, it may be very important to the discloser to preserve the confidentiality of the disclosed information indefinitely. This is an issue that the parties should consider based on the specific facts and needs of the parties.

When representing the disclosing party, as a general rule, it is desirable to preserve the information in confidence indefinitely. The discloser always has the ability to end the confidentiality obligation by publicly disclosing the information. This can be accomplished by including language in the basic confidentiality provision to the effect that the recipient shall maintain such information in confidence "both during and after the Disclosure Period".

Return of materials

The discloser of confidential information will want to include a clause requiring the recipient to return the confidential information to the discloser upon request. Here is an example of such a provision:

> Upon the request of Discloser, Recipient will promptly return to Discloser all Confidential Information and all copies thereof in Recipient's possession or under Recipient's control, and Recipient will destroy all copies thereof on Recipient's computers, disks and other digital storage devices.

When the confidentiality clause is part of a larger agreement, the agreement should provide that the confidential information will be returned to the discloser upon the expiration or termination of the agreement.

Other provisions

In addition to the clauses described above, a confidentiality agreement might contain provisions to the effect that:

- the discloser may obtain both injunctive relief and monetary damages in the event that the recipient fails to comply, and that the recipient will pay for the discloser's attorneys' fees
- a recipient in breach must indemnify if a third party sues the discloser due to the breach
- monetary liability is limited to a specified dollar amount
- neither party will be obligated to enter into any other agreement
- the disclosed information remains the property of the disclosing party
- the recipient shall immediately notify the discloser upon discovering any loss or unauthorized disclosure by any of the recipient's personnel of any confidential information
- each party and its affiliates shall comply with all applicable laws, rules and regulations, including those relating to technology export or transfer
- neither party is obligated to provide any particular information
- the discloser is disclosing the information "as is" or with implied or express warranties
- the discloser is granting no license
- the recipient is not restricted from providing competitive products or services to others
- the recipient may not reverse engineer, decompile or disassemble any disclosed software
- the recipient will not export any disclosed software in violation of any export laws
- the agreement is governed by the laws of a specified state
- the parties will mediate and then arbitrate any dispute (with a carve-out for injunctions)

Software Development Agreements

After the parties have discussed their IT project in detail and have reached a general understanding of the transaction, they will come to you to draft or review the contract. Your job will be to ensure, among other things, that the contract accurately reflects the understanding of the parties. In order to do this, you must discuss the transaction with your client, raise questions and read the documentation so that you have a clear understanding of the transaction.

One basic consideration is the technology itself. In order to draft or review the contract, you should understand the technology in terms that you can articulate. Ask your client to explain the transaction in his or her own words, because this explanation may differ significantly from the vendor's literature or proposed contract. Don't be embarrassed to ask your client questions that may reflect your lack of understanding of the technology. In most cases, the client will be pleased to educate you. The better you understand the technology, the better chance you have of preparing and negotiating an excellent contract.

In addition to understanding the technology, you will need to understand the sources of the technology and the roles of various parties. Are you dealing with a license agreement from the company that will install it and integrate it into your client's computer system? The vendor may be an integrator of another party's software, in which case your client may need a separate license agreement from the software owner. In some cases, the system may require more than one third party license.

Once you have an understanding of the client's goals, you will also be in a position to determine what types of agreements are required. While the client may enthusiastically want to proceed with the transaction, you can play a quality control function, ensuring that the transaction generally makes good business sense and that client has considered all of the relevant issues.

IT projects entail many types of agreements, including confidentiality agreements, software license agreements, maintenance agreements and escrow agreements. While these agreements are often separate and distinct, many transactions combine elements of two or more of these types of agreements in the same contract.

You should consider breaking the project into small, well-defined phases with clearly articulated allocations of responsibility for each phase. Such an approach can help to minimize the likelihood of disputes.

Defining the Deliverables

In any software development agreement, it is in the interest of both parties to define in detail the product that the vendor is agreeing to deliver. This is true regardless of who will own the software. The written agreement should state that the software will be developed in accordance with a detailed description attached as an addendum to the agreement. A software agreement should include a detailed description of the functionality of the software. If the functionality is not yet defined, the agreement should establish milestones for developing goals.

Specifications include integration with existing data and software, the delivery of operator and user manuals, training of employees, and much more. In many cases, it is desirable to have an initial contract in which the vendor does nothing more than design the specifications for the project. The project could then be put out to bid with those specifications. Of course, the vendor who prepared the specifications would have an advantage in the bidding, but the customer would have the opportunity to negotiate a better deal going forward.

The specifications should also include delivery of detailed user documentation.

A detailed description of the intended work product reduces the likelihood of a dispute over whether the delivered software is acceptable. If your client, the customer, prepared a detailed request for proposal (RFP), this would be a good starting place for the preparation of the software description addendum. You should beware, however, that there are good and bad RFPs. An RFP that is too broadly written can elicit too broad a range of proposals. On the other hand, an RFP that is too restrictive can result in a project that may not be the ideal solution. Getting the intended users involved in preparing the specifications for any request for proposal (RFP) can help enormously in preparing good specifications and avoiding future misunderstandings.

If your client did not prepare an RFP, the vendor's written proposal would be a logical place to start preparing the software description addendum. Many vendors call their proposal a "statement of work". Another place to look is the vendor's advertising and sales materials, whether printed or online. These materials will help you understand the product and its functions so that you can draft a better contract, possibly using some of the vendor's language in the software description addendum. In addition, you may find product claims and other representations that would be helpful to the customer if they are included in the contract. Such claims can be included in the software description addendum and become express warranties.

Software that is promised but never delivered is commonly called "vaporware". Some vendors make claims about software that does not actually exist. It is a good idea to clarify in the contract whether the promised software is completed and is in operation at other locations. If not, at what stage of development or testing is the software? The agreement should also include contingency plans if the software fails to satisfy the customer's requirements.

Software modifications during development

From the vendor's point of view, defining the deliverables in detail is important in order allow the vendor to price the project. However, a large project can be a moving target. Because of the need to cooperate and exchange information, the agreement should specify the customer's obligations to provide information and access to its systems and data. The agreement should provide for progress reports and meetings of personnel at specified levels on both the customer and vendor side. It should provide for payments at specified milestones that bear some relation to the amount of work performed.

Many customers ask for frequent changes in specifications as the software is being developed. Each change results in additional work for the vendor and a delay in the completion date.

A good development agreement will include a clause allowing for changes by means of written addenda to the agreement, just as construction contracts allow for work change notices. The agreement should contain another addendum with a form of work change notice. When the customer desires a change, the customer submits the notice to the vendor, who comes back to the customer with a proposal setting forth the added price and new delivery dates. If that proposal is acceptable, the parties then sign an amendment to the agreement. The customer should acknowledge in the agreement that modifications made pursuant to work change notices may affect agreed delivery dates and that any expected delay will be indicated in the relevant work change notice.

The use of a separate agreement for software modifications is a good format for other situations that may arise in software contracting. The parties may have a master consulting agreement or a maintenance agreement that sets forth certain defined work, but allows for other projects to be agreed pursuant to short-form agreements that refer back to the master agreement. This avoids the necessity of inventing the wheel each time a new project arises with the same vendor. You might consider adding a clause to a maintenance agreement that states that any additional work by the vendor during the term of the agreement is covered by the agreement even if the work is not described in an addendum to the agreement and even if that additional work is not covered by a separate contract. Here is a clause that accomplishes this purpose:

> As used herein, the term "Services" also includes services performed by Vendor during the term of this Agreement in addition to those services described in Schedule A. For all work beyond the scope described in Schedule A that Vendor agrees to undertake for Customer, including all annual renewals, the parties intend to sign a work order containing a written description of the services to be performed by Vendor, as outlined in Schedule B ("Work Order"). This Agreement will apply to all services that Vendor performs for Customer, regardless of whether the parties sign a Work Order covering such services.

Testing and Acceptance

Testing is an essential aspect of any software project. The IT department will not roll out a new software application or upgrade without first testing it to be sure that it works properly and integrates properly into the company's systems. When a consultant or developer is creating software code for a customer, the customer must have the right to test the software and then to accept or reject it. The agreement should specifically define the method of acceptance. Here is a sample acceptance clause:

> Customer will have the right to accept the completed Software as follows:
>
> (i) After its receipt of the completed Software, Customer will have thirty days to determine whether the completed Software conforms to the requirements of Section ___.
>
> (ii) If Customer does not notify Vendor within such thirty-day period that the completed Software does not conform to such requirements, then Customer will be deemed to have accepted the completed Software.
>
> (iii) If Customer discovers any nonconformities from the specifications agreed to by the parties in writing and notifies Vendor in writing (including by fax or e-mail), Vendor promptly shall correct such nonconformities at no additional charge to Customer.
>
> (iv) Once Vendor believes that it has corrected all such nonconformities, Customer again may determine whether the corrected Software conforms to all agreed-upon requirements. If Customer still finds nonconformities, Customer may

> reject the completed Software and terminate this Agreement upon notice to Vendor.

The thirty-day period may be three months or a different length of time, depending on the project or the phase of the project.

Acceptance can also occur at different levels. First, the software can be accepted in a test environment. The customer might then want a second acceptance period after the software is deployed in an operational setting. The second acceptance date may be the date that the warranty period begins.

Vendors will want to make it clear in the agreement that the customer may only reject a deliverable when the deliverable does not substantially conform to the written requirements agreed by the parties. The vendor will want any rejection notice to identify in detail how the deliverable fails to conform to the requirements.

Completion Date

Software projects often take far longer to complete than the vendor anticipates. In fact, many software projects are never completed at all. The contract should contain a timetable and should contain consequences for the vendor's failure to perform in a timely manner. At some point, the customer should have the right to terminate the contract by reason of delay.

The software vendor, of course, will want to be protected against delays caused in whole or in part by the customer. For example, the vendor may need access to equipment or data that the customer controls. Likewise, the vendor may be concerned about the need for cooperation by one or more third party licensors that are under contract with the customer. If the vendor's completion depends on such cooperation, the vendor may want a provision stating that the customer will ensure that the third party licensor cooperates as necessary with the vendor. In addition, the vendor may specifically request the right to delay based on the failure of the customer to pay any invoice within the required time period.

Another cause of delay may be forces beyond the control of either party. A force majeure clause may simply relieve a party of its contractual obligations during the time that such party is unable to perform due to a cause beyond its control. If the period lasts longer than a defined time, the parties might have the right to terminate the contract. Some force majeure clauses require the disabled party to give notice of the occurrence of an event that requires the suspension of its performance.

What should be the consequence of the vendor's failure to deliver an acceptable work product in a timely manner? Typically, a software agreement will include a provision allowing the customer to terminate the agreement. Here is an example:

> In the event the Software does not function substantially in all material respects, in accordance with the specifications supplied by Licensor, notwithstanding Licen-

sor's attempt to replace the Software or correct the non-conformances within a reasonable time frame using best efforts,

> *[alternative 1]* Licensee's sole remedy shall be to terminate this Agreement in accordance with Section ___ and receive a refund in accordance with Section ___.
>
> *[alternative 2]* Licensee shall either return the Software to Licensor, or destroy and certify the destruction of same in writing to Licensor, whereupon Licensor shall refund to Licensee as follows: (a) during the first year, Licensor shall refund the entire License Fee and (b) for every year thereafter the applicable pro rata portion of the License Fees and Maintenance Fees, and the license for such Software shall terminate. The pro rata amount for the License Fees shall be calculated based on a five year life expectancy for the Licensed Program(s) and Materials. The pro rata amount for the Maintenance Fees is based on the portion remaining in the then current year's Maintenance term. The foregoing remedy is in addition to all other remedies available to Licensee in equity or at law.

Warranties and Limitations of Liability

Warranty Period

After the customer accepts the software, the customer usually has a warranty period during which the vendor is required to fix problems and provide some level of support. Warranty periods vary in length. They are frequently twelve months, although they may be as short as three months. After the warranty period expires, customers commonly receive ongoing service through a maintenance agreement. Here is a sample warranty clause:

> For a period of ____ months from Customer's acceptance of the completed Software (the "Warranty Period"), Vendor represents and warrants that such Software will conform to all agreed-upon requirements. If, during that period, Customer notifies Vendor that the Software does not conform to agreed-upon requirements, then Vendor promptly shall correct such nonconformities at no charge to Customer. If Vendor fails to correct any problem, programming error or bug reported during the Warranty Period within thirty days after receipt of notice, Customer may contract for such work to be done by any third party and Vendor shall reimburse Customer for the reasonable cost of such work.

Express Warranties

Most customers will want far more extensive warranties than merely a warranty that the software will conform to all agreed-upon requirements, and they will want warranties that last beyond the "warranty period". This is true especially in vendor contracts in which the vendor prominently states that it makes no warranties other than those expressly set forth in the agreement.

The ownership issue appears again as a central subject in the warranty clause. A customer who licenses software or engages a company to develop software should obtain a warranty from the vendor or licensor that the software will not infringe on the rights of any third party.

In addition to warranties that the software and all fixes and enhancements and new versions will conform to agreed-upon requirements and will not infringe the rights of any third party, a customer might require warranties from the vendor that:

- The software and all enhancements and new versions will contain no known defects.
- Vendor has the right to enter into the agreement and to perform its obligations under the agreement.
- The agreement is its legal, valid and binding obligation.
- Neither vendor nor its employees have been or are the subject, directly or indirectly, of any governmental order, investigation or action of any kind, including without limitation any order or action to revoke or deny any export privileges, and vendor will notify customer immediately in the event vendor or any of its employees become subject to any such order, investigation or action.
- Vendor is under no obligation or restriction, nor will it assume any such obligation or restriction, which would in any way interfere or be inconsistent with, or present a conflict of interest concerning, the services which are the subject of the agreement.
- Vendor's performance will not breach or conflict with any prior obligation of vendor to any other party, including any obligation to keep confidential any information acquired by vendor before the date of the agreement.
- Unless approved in advance by customer, no information vendor discloses to customer in providing the services that are the subject matter of the agreement will be confidential to vendor or any third party.
- The vendor, if a licensor, has the right to grant a license to the software free and clear of any liens and encumbrances.
- The vendor is not currently the subject of any litigation or pending claim that would materially affect the vendor's ability to perform.
- The fees and hourly rates set forth in Schedule __ are the best rates vendor offers to its customers.
- The software and all enhancements and new versions will contain no known computer virus or other "contaminants", including any codes or instructions that may be used to access, modify, delete, damage or disable customer's computer system, which shall include, but not be limited to, security or expiration codes.

Another way of covering the last point is to include the following provision:

> Licensor expressly waives and disclaims any right or remedy it may have at law or in equity to de-install, disable or repossess the Software should Licensee fail to perform any of its obligations under this Agreement.

Here is another provision to accomplish the same purpose:

> In no event shall Licensor have the right to purposefully or accidentally electronically repossess the Software. For purposes of this Agreement, "repossess" shall include, but not be limited to, electronic lock-outs or booby traps.

When the vendor incorporates third party software into the software it is licensing or selling, the customer may want to include an obligation on the part of the vendor to obtain comparable warranties from such third parties and shall assign such warranties to the customer and cooperate with the customer in the enforcement of any such warranties.

What about the term or survival of these warranties? While the warranty regarding the conformity of the software to the agreed-upon requirements may have a fixed term, for example, of twelve months, you may want the warranty against infringement to last indefinitely, or at least for several years.

Y2K warranties are rare in current contracts for obvious reasons. Before January 1, 2000, lawyers working on IT contracts were very concerned about whether the software or hardware they were using and acquiring would work properly in dealing with dates in the 21st century. Today, if there is a date problem in the software, it will come to light during testing. There will be no uncertainty about a different result based on a date following December 31, 1999.

Some IT contracts define levels of software errors and deal with each level in a different manner. For example, the contract might define a "fatal error" as one that results in the inability of the software to perform a vital business function of the customer (as further defined in the agreement). The contract might provide, for example, that if the customer discovers a fatal error within six months, then the vendor will handle the error in the same manner as it would handle infringement. In other words, the vendor would either modify or replace the software, offer some work-around, or terminate the license and pay the customer the depreciated book value of the software. The vendor might want to provide that in order for the customer to be eligible to obtain any such remedy, the customer must have installed all the latest fixes issued by the licensor. The agreement might provide that the licensor or vendor will use its best efforts to fix any error other than a fatal error.

Liability Limitations and Exclusions

Aside from potential liability for breach of an express warranty, the vendor will typically want to limit its liability. One approach a vendor might use is to state that, except for the specific warranties set forth in the contract, the software is licensed "as is". Vendors commonly disclaim the following in bold font in the agreement:

- all implied warranties and warranties of fitness for a particular purpose
- any warranty that operation of the software will not be uninterrupted or error free
- any warranty regarding the results of use of the software.

Under Section 401 of the Uniform Computer Information Transactions Act (UCITA), discussed below, any disclaimer of a warranty noninfringement would have to be explicit. A general disclaimer of all implied warranties would not suffice. Of course, this is relevant only in when UCITA governs the contract and only when the contract contains no express warranty of noninfringement. Many contract specifically disclaim the application of UCITA. In most other cases, the laws of a UCITA state will not apply. Although only Virginia and Maryland have adopted UCITA courts in other states look to UCITA for authority.

Vendors frequently write into their contracts in bold text that they will not be liable for:

- consequential damages
- defects relating to external factors, *i.e.,* integration between the licensed software and the licensee's hardware or other programs.
- software modifications done by licensee

In addition, vendors may go as far as to require the customer to hold the vendor harmless against infringement or misappropriation claims arising out of the vendor's compliance with specifications when the customer supplies those detailed specifications and the method required for meeting those specifications. This obligation appears in Section 401(a) of UCITA, which also makes an exception for a claim that results from the failure of the vendor to adopt or notify the customer of a noninfringing alternative of which the vendor had reason to know.

Licensors and other vendors commonly seek to limit their total liability, often to the total license fees paid within the last 12 months, or the total amount that the customer paid the vendor for the project.

Business customers should be careful to carve certain exceptions out of the vendors' limitation of liability. It is common, for example, to exclude from the vendor's disclaimer any liability arising from:

- viruses or other contaminants
- third party claims (under the indemnity provision)
- the vendor's breach of its confidentiality obligations

A vendor might also disclaim retroactive compatibility of upgrades or fixes. In other words, the vendor might come out with an upgrade or fix that suddenly makes the software incompatible with the customer's existing system. It is for this very reason that customers must test all upgrades and fixes before they are rolled out into operation. Here is a sample clause dealing with this issue from the customer's point of view:

> Except as expressly provided in writing signed by Licensee, the Software shall be compatible with and not have its performance degraded by, all software platforms and operating systems which are in use or soon to be used by Licensee, as specified in the Schedule.

Indemnification

A company that licenses or acquires software from an IT vendor should include a provision in its IT agreements pursuant to which the vendor will indemnify the customer for liability to third parties arising out of the failure of any warranties or the vendor's breach of the agreement. Here is a sample indemnity clause:

> (a) Both during the Term and after the expiration or termination of this Agreement, Vendor will indemnify and hold Customer and its stockholders, directors, officers and employees harmless from and against any and all liabilities, suits, claims, losses, liabilities, damages and expenses, including reasonable attorneys' fees and costs, arising out of or incurred for the purpose of avoiding, any suit, proceeding, claim or demand or the settlement thereof, by reason of the failure of any of the representations or warranties set forth in Section ___, or relating to or arising out of any acts or omissions of, or any breach or alleged breach of this Agreement by, Vendor, Vendor's employees, agents and others acting for Vendor or on Vendor' behalf.
>
> (b) Customer shall promptly notify Vendor of any suit, proceeding, claim or demand brought or made against Customer for which Customer may be entitled to indemnity hereunder.
>
> (c) If any such suit, claim or demand is brought or made, Customer may elect to undertake the defense thereof or to request Vendor to undertake such defense. In either event, the cost and expense of the defense of any such suit, claim or demand will be borne by Vendor.
>
> (d) Whenever any such suit, claim or demand is instituted, Customer may withhold payments due to Vendor, subject to Vendor's right to draw on such sums to defray Vendor's expenses in connection with such suit, claim or demand. If a final adverse judgment is rendered in such a suit and is not discharged by Customer, Customer may apply the payments so withheld to the satisfaction and discharge of such judgment.

A vendor will want to condition any such indemnity upon the customer's giving of prompt notice and will want to have the right to defend the case. Here is a sample clause that is more favorable to the vendor:

> Licensor shall indemnify, defend and hold Licensee, its officers, directors, agents, consultants, employees and Affiliates harmless from all costs and damages incurred by Licensee solely to the extent any claim or action based on infringement or misappropriation of any United States copyright, trade secret or other proprietary right based upon Licensee's use of the Software in unmodified form as delivered by Licensor; provided that Licensor is promptly notified in writing of any such suit or claim against Licensee and further provided that Licensee permits Li-

> censor to defend, compromise, or settle such claim and gives Licensor all available information, reasonable assistance, and authority to enable Licensor to do so.
>
> If the Software or any part thereof is held to constitute an infringement or a misappropriation, and use of the Software by Licensee is enjoined, then Licensor will endeavor at its own expense, to either, at Licensor's option: (i) procure for Licensee the right to continue using the Software or part thereof; or (ii) replace the Software with a functionally equivalent non-infringing product; or (iii) modify the same so as to make it non-infringing while remaining functionally equivalent; or (iv) terminate this Agreement as to the infringing Software and refund to the custom the depreciated book value of the Software, based on a 5-year useful life, upon Licensor's receipt of the Software from Licensee or signed written certification that the Software and any copies thereof in Licensee's possession have been destroyed.

The above clause is favorable to the licensor because, among other things, it calls for a holding of infringement and an injunction against use of the software before it applies. From the customer's point of view, the following clause would be a better starting point on the subject of a remedy for infringement:

> If a third party objects to Licensee's use of the Software, Licensee will notify Licensor immediately. Licensor shall assume the defense of any infringement litigation, with Licensee's cooperation, at Licensor's expense. In the event of any such infringement, Licensor will either (a) obtain a license enabling the Licensee to continue using the software or (b) bring the infringement to an end by modifying the software or replacing it with other software that performs the same functions or (c) terminate the license upon notice to the Licensee, in which event Licensor shall pay Licensee the depreciated book value of license, based on a 5-year useful life, and Licensee shall return the Software and documentation and destroy all copies.

The vendor may also want to include language along the following lines:

> Licensor shall not be liable to Licensee under any provision of this Section __ if any patent or copyright infringement or claim thereof is based upon (i) the use of the Software, or any part thereof, in connection with any product, software or system not developed by Licensor, (ii) use of the Software, or any part thereof, in any manner other than in accordance with specifications contained in the Materials, (iii) modification of the Software, or any part thereof without Licensor's approval, by Licensee or its agent in a manner causing it to become infringing, or (iv) Licensee's use of any version of the Software other than the then most current version. The foregoing states the entire liability of Licensor to Licensee with respect to any copyright, trade secret, or patent infringement by the Software or any part thereof.

A provision placing liability on the vendor will not help the customer if the vendor does not have adequate resources to pay. For this reason, the customer may want the vendor to carry insurance covering the indemnified risks. Some companies will work only with vendors that provide proof of both general liability insurance and professional liability coverage, naming the customer as additional insured. The contract should state the required limit of the policies (usually at least $1 million for each occurrence) and should require the vendor to produce certificates of insurance annually to show that the required policies are in effect. The contract should also require the vendor to carry workers' compensation insurance as required by law.

License and Maintenance Agreements

Source Code Escrow and Bankruptcy

If your company intends to own the rights in the software, you will want the developer to deliver the source code along with adequate documentation so that another programmer can readily understand it and modify and improve it. You will also want the developer to agree to maintain the confidentiality of the source code.

What about a software licensee? Software license agreements typically do not allow the licensee access to the source code. In fact, software license agreements commonly prohibit reverse engineering that would enable a programmer to decipher the source code. Does your company need access to the source code if your company is a licensee? The answer to this question is likely to depend on factors including (a) the degree of pain that a loss of the ability to fix, modify and upgrade the software would cause to your company and (b) whether the licensor is a large and stable company. A licensee will be more motivated to seek access to the source code if the software is unique, central to the business and developed by an individual or a small company. The need for access to the source code is less pressing if other available software will serve the same function and the licensor appears likely to endure as a going concern so that it can fix bugs in the software and enhance and upgrade it.

As a general rule, the owner of rights in software is reluctant to disclose the source code. In fact, software licensors jealously guard access to the source code in most cases. Given this reality, what types of protection can a licensee seek?

One approach is a source code escrow agreement. A licensee with adequate bargaining power may succeed in having the source code placed in escrow with an independent software escrow company for a fee, thereby providing some assurance that the source code will be available to the licensee in the event that the licensor goes out of business, is acquired or decides to stop supporting the software. DSI (www.ironmountain.com), for example, has standard forms of escrow agreements that are available on the web. You will need to customize these standard agreements to your individual transactions.

In the event that the licensor becomes the subject of a bankruptcy proceeding, Section 365(n) of the Bankruptcy Code protects the enforceability of an escrow agreement. In addition, the source code held in escrow is likely to be outside of the property of the bankruptcy estate within the meaning of Section 541 of the Bankruptcy Code. *See, e.g., In re U.S. Interatctive, Inc.,* 285 F.3d 1078 (3d Cir. 2002). Accordingly, the bankruptcy filing should not prevent the licensee from obtaining the source code from escrow.

In practice, a source code escrow must be actively monitored in order to be effective. The protection of an escrow arrangement is often illusory. In order for the escrow to be effective, the source code must be adequately documented and the source code of each and every update to the software should be delivered into escrow. Few companies enforce this type of ongoing obligation. Nevertheless, a source code escrow is a meaningful protection in many cases.

Even without a source code escrow, a software licensee may have a degree of protection under the Bankruptcy Code.

License agreements are executory contracts for bankruptcy purposes because they contain ongoing obligations. (11 U.S.C. § 365) The licensor is obligated not to sue the licensee for infringement; the licensee is obligated to use the software in accordance with the terms of the license. Other "executory" obligations may include indemnity, support and confidentiality.

In bankruptcy, the debtor-in-possession or trustee has a choice between keeping the executory license in effect ("assuming" it) or terminating it ("rejecting" it). Under section 365(n), if a debtor or trustee rejects an executory contract under which the debtor is a licensor of a right to intellectual property, the licensee may treat the contract as terminated or retain its rights under the contract. Section 365(n) covers licenses of trade secrets, patents and copyrights, but not trademarks.

Section 365(n) protects the rights of licensees of intellectual property when the debtor-in-possession or trustee rejects the license. If the licensee seeks to retain its rights under the license agreement, the licensee must (a) notify the debtor in writing, (b) make all royalty payments, and (c) waive a right to set off and any administrative claims.

In order to avoid the continuing obligation to pay royalties, the drafter of the contract should consider separating all of the license obligations from the maintenance obligations. The license can be a relatively small payment compared to the amount to be paid in maintenance and consulting. It can also be paid early in the life of the project, so that it is "fully paid up" in the event of bankruptcy.

If the licensee retains its rights under a software license agreement in bankruptcy, the debtor/licensor has no affirmative obligations but continues to be obligated under its negative covenant not to sue for infringement. In the case of a software license, this absence of affirmative obligations means that the licensor is not obligated to continue to maintain the software. As a practical matter, the licensee will have to provide its own maintenance, support and continuing development or hire other parties to handle this work.

For this reason, it is important from the licensee's point of view that the license agreement provide for access to the source code by the licensee.

Here is an example of a simple clause in a license agreement covering bankruptcy. The clause essentially tracks the licensee's rights in the event of the licensor's bankruptcy.

> In the event that Licensor becomes the subject of any proceeding under Title 11 of the United States Code (the "Bankruptcy Code"), and the trustee rejects this Agreement or the escrow agreement referred to in Section __, Licensee may elect to retain its rights under this Agreement and the escrow agreement in accordance with Section 365(n) of the Bankruptcy Code. Upon written request of Licensee to Licensor or to the bankruptcy trustee, Licensor or the bankruptcy trustee shall not interfere with the rights of Licensee as provided in this Agreement and the escrow agreement, including the right to obtain the escrowed materials from the escrow agent.

It would be advantageous to the licensee to trigger the release of the escrow on pre-bankruptcy breaches or failures. Depending on the specific agreement, you may find pre-bankruptcy breaches that would be appropriate in a particular agreement.

In the event of bankruptcy, the licensee will have no rights other than those that are already in the agreement. Accordingly, the agreement should specifically allow the licensee the full right to modify and create derivative works from the software in the event of a release of the software to the licensee from escrow, and it should allow the licensee the full right to disclose the code to third party programmers engaged by the licensee to modify or update the code.

If the software licensor prefers not to place the source code in escrow, there are other ways to protect the interest of the licensee by contract. For one thing, the agreement can require the licensor to maintain a backup of the source code at a secure, off-site facility, at all times during the period that licensor is providing maintenance services to licensee. In addition, the agreement can require that the licensor deliver the source code to the customer upon the occurrence of the same events that would trigger such an obligation by an escrow agent, although it is not clear whether this would be enforceable in bankruptcy. The agreement might require, for example, that the licensor deliver the source code to the licensee upon the occurrence of any of the following events:

> (a) Licensor's failure to continue to do business in the ordinary course or its cessation of business operations; (b) initiation by Licensor of liquidation of its company; (c) discontinuance of Licensor's maintenance services to customers generally; or (d) any one or more of the following: (i) entry of an order for relief under Title 11 of the United States Code with respect to Licensor; (ii) the making by Licensor of a general assignment for the benefit of creditors; (iii) the appointment of a general receiver or trustee in bankruptcy of Licensor's business or property; or (iv) action by Licensor under any state or federal bankruptcy, insolvency or similar law for the purpose of its bankruptcy, reorganization, or liquidation.

This provision affords less protection than an escrow agreement, but in combination with the rights of the licensee under bankruptcy law and perhaps the addition of one or more pre-bankurptcy triggers, this provision may afford the licensee adequate protection.

The licensee can take the additional step of requiring the licensor to provide the names and contact information of the programmers of the software in question. In the event of bankruptcy, the licensee may be able to hire the programmers to continue their work on the software. If the licensor is reluctant to provide this information at the time the agreement is signed, the information can be placed in escrow or the licensor can be obligated to provide the information in the event of the licensor's bankruptcy.

Scope of License

The logical starting point of any software license agreement is the grant provision, which describes the scope of the grant. Here are a few of the basic grant considerations:

- Exclusivity – Most licenses are explicitly nonexclusive because the licensor is in the business of licensing the same software to others.
- Term – Normally, the grant of the license is perpetual, although it will terminate upon the occurrence of certain events. This does not mean that the licensor will support the software indefinitely or provide updates, fixes and new versions indefinitely, but only that the licensee may continue to use the licensed version.
- Assignability – Is the license assignable without the consent of the licensor? Can it be assigned to the licensee's affiliates? How are affiliates defined?
- Copies – Is the licensee permitted to make at least one (preferably more) copies for backup/archival purposes? How about copies for development and testing?
- Extent of use – Does the license cover any use at a specified site (a site license)? Does it cover a defined number of simultaneous users or CPUs (central processing units)? Does it specify one CPU? Can the licensee relocate the software without penalty? Can the licensee relocate the software without charge to another designated CPU, with the ability to use the software at both locations for a limited period of time (e.g., six weeks) in order to accommodate an orderly relocation. Does the license extend to use by affiliates of the licensee?

Microsoft recently imposed severe restrictions on the extent of use in its own license agreements. Although Microsoft's stated goal is to reduce piracy, the new "Product Activation" feature in Windows XP and Office XP places serious limits on the ability of users to move the software from one computer to another. Using the Internet, Microsoft takes a snapshot of your PC configuration and links it to your product registration number in its database. Most corporate users can avoid these restrictions with Microsoft's volume licensing agreements. Microsoft does not require "product activation" for volume-licensed customers.

Maintenance and Service Level Agreements

Maintenance agreements provide a customer with ongoing support. They are sometimes referred to as service level agreements.

- Type of support – Will the vendor provide support on site or by telephone, e-mail or internet?
- Hours of support – What hours will the vendor be available to be contacted?
- On-site response time – If the agreement calls for on-site support, how quickly will the vendor respond to a request? Within 24 hours? Within a specified number of hours?

- Fixes and updates – Maintenance agreements typically include the delivery of fixes and updates. Sometimes, they include new versions as well. Some agreements provide for guaranteed repair time or delivery of a work around solution.
- Fees – Better service will usually cost more.

Here is a sample clause dealing with maintenance:

> (a) *Commencement and Term of Support.* Vendor will provide telephone support and maintenance for Customer with respect to the Software beginning upon acceptance by Customer of the completed Software. Vendor will provide such support and maintenance on a year-to-year basis. Each such year of support and maintenance after the first year is hereinafter referred to as a "Renewal Term". After the first year of support and maintenance by Vendor, such support and maintenance will automatically renew from year to year unless either party notifies the other party of its intention not to renew at least sixty days before the end of the first year or of any Renewal Term or unless Customer fails to pay to Vendor the base Renewal Term fee for such support and maintenance before the start of such Renewal Term in accordance with Section __ below.
>
> (b) *Level of Support.* To the extent that Customer needs support from Vendor with respect to the functionality or operability of the Software, Vendor will provide telephone support, promptly responding to each telephone request from Customer, using its best efforts to solve each problem presented; provided, however, that Vendor shall provide such support directly to one of the liaison persons designated by Customer and not directly to Customer's other employees or customers. Vendor shall also provide help files online.
>
> (c) *Delivery of Updates.* During the Warranty Period and for as long as Vendor is providing support and maintenance for Customer with respect to the Software, Vendor shall provide such enhancements in and new versions of the Software as the parties may agree in writing. Unless otherwise agreed in writing, signed by both parties, such enhancements and new versions will be agreed in the form of work change notices signed by both parties that will be governed by the terms and conditions of this Agreement. At the end of any calendar quarter in which Vendor makes a fix or provides an enhancement or new version, Vendor shall deliver to Customer new copies of the Software containing such fixes or enhancements, or a copy of the new version.

Larger IT projects may call for various levels of support for various issues that arise, or different responses to different types of problems. This is commonly called a service level agreement. Such an agreement details the specific, quality, availability, performance levels and support services a customer can expect from its service provider. The agreement should address expected response times for computer applications, system capacity and interface compatibility. The customer's needs may be developed as part of the project.

What are the resolutions to each service-level failure? Describe the process in detail. How quickly must the vendor correct each failure? What measurement and enforcement tools and processes will be implemented to ensure that performance can be measured and enforced? Does the customer have the ability to get out of the contract without penalty if the vendor is not meeting its obligations? (See "Overwhelmed by Outsourcing? Here's Help!" Katherine F. Twiddy, Law Technology News, May 2002, www.kilpatrickstockton.com.)

Some agreements impose financial penalties for the vendor's failure to respond adequately. This provides a negative incentive to the vendor, but it may not adequately compensate the customer for the damage caused by the system failure. A better approach might be to specify the individuals who will be directly responsible for repairs, along with their contact information. If an issue is not resolved in a timely fashion to the customer's satisfaction, the agreement should identify a senior member of vendor's management who will address the issue. In other words, if the customer is sufficiently unhappy, the agreement should allow the customer to require an executive vice president of the vendor to fly out to the customer's site and explain why the network doesn't work. Most vendors will energetically avoid the necessity of such an outcome. ("Crafting Service-Level Agreements that Work", Johna Till Johnson, Network World, March 25, 2002, www.nwfusion.com)

Although a vendor may resist penalties for failure to respond adequately, the vendor may be willing to accept penalties together with bonuses for responses that exceed the contractual requirements.

One issue to consider in maintenance agreements is how long the licensor will support each version. At some point, a vendor may decide to stop supporting an outdated version of its software. The customer needs some assurance that the software will be supported for a reasonable period of time before the customer is required either to upgrade or change its software. Here is a sample clause that deals with this issue:

> Licensor shall make available to Licensee, within three months of availability to similarly situated clients, any new release of the Software, or any Enhancement(s). Licensee may within one year choose to implement any such Enhancement. Licensor shall maintain the previous Software release until Licensee's one year implementation period has expired. Should Licensee elect not to implement any Enhancement within the one year implementation period, and Licensor elects not to provide any maintenance on the previous Software to similarly situated clients, then maintenance for the previous release shall be on a time and materials basis at the rate specified on a price list attached to this Agreement, which rate shall not increase in any year by more than four percent (4%) over the previous year's rate.

The one-year period may be changed to two, three, four or five years, depending on the needs of the customer and the willingness of the vendor. The vendor may also want to limit the number of years it will support a particular version even on a time and materials basis.

Assignment

Software license agreements usually include a restriction against assignment by the licensee without the licensor's consent, which typically will not be unreasonably withheld. The agreement typically states that the assignment must effect a complete transfer. In some cases, it requires as a condition to any transfer that the assignee sign a new license agreement with the licensor. One reason for the restriction is that the licensor does not want proprietary information transferred to one of its competitors.

For the corporate licensee, it may be important to provide that the licensee may assign the agreement to an affiliate or to a buyer of the business. Here is a sample assignment provision in a software license agreement that deals with these questions and contains elements that give the licensee a reasonable degree of flexibility:

> Neither party may assign this Agreement to any third party without the prior written consent of the other party, which consent shall not be unreasonably withheld. Notwithstanding the foregoing, Licensee may, without liability for new license fees, upgrade fees or any other type of transfer charges, and upon notice to but without the consent of Licensor, assign all or any portion of its rights, duties or obligations pursuant to this Agreement to any entity which is an Affiliate; (i) into which Licensee may be merged or reorganized; or (ii) to which all or a portion of the capital stock or assets of Licensee may be sold, transferred, assigned or otherwise disposed.
>
> Should Licensee divest itself of an Affiliate, Licensee shall terminate use of the Software for the benefit of the divested Affiliate within one hundred eighty days of the effective date of Licensee's divestiture of such Affiliate.

While software license agreements typically restrict transfer by the licensee, they also typically place no restrictions upon transfer by the Licensor. For most licensees, this poses no problem. However, licensees may also be concerned about transfer to competitors.

Termination

When a license is fully paid, the license term is usually indefinite. However, many license agreements call for termination upon the occurrence of certain events. For example, the licensor may retain the right to terminate the agreement upon the occurrence of a breach by the licensee; or the agreement might allow either party to terminate upon notice to the other if the other party defaults. "Default" might be defined to mean:

> (i) failure by a party to comply with or to perform any material provision of this Agreement and continuance of such failure for thirty (30) days after receipt of notice thereof by such defaulting party, unless the defaulting party has commenced and is diligently endeavoring to remedy any such failure that cannot be cured within such thirty-day period, and unless such failure is caused by the delay

> or failure by the other party or failure by a necessary Third Party Licensor to cooperate or to provide Licensor with necessary information in a timely manner; or
>
> (ii) any representation or warranty made by a party to this Agreement is false or misleading in any material respect.

The agreement should also state what happens in the event of termination. Here is one clause along these lines:

> In the event Licensee is the defaulting party, Licensee shall forthwith return to Licensor or destroy the Software and destroy all copies thereof on Licensee's computers, disks and other digital storage devices, and certify such destruction in writing to Licensor, and Licensor shall have no obligation to refund any amount to Licensee.
>
> In the event Licensor is the defaulting party, Licensee may elect to return to Licensor or destroy the Software and all copies thereof on Licensee's computers, disks and other digital storage devices and certify such destruction, whereupon Licensor shall refund to Licensee as follows: (a) during the first year after receipt by Licensee of the Software, Licensor shall refund seventy percent of the License Fee and (b) for every year thereafter the applicable pro rata portion of seventy percent of the License Fees and Maintenance Fees, and the license for such Software shall terminate. The pro rata amount for the License Fees shall be calculated based on a five-year life expectancy for the Software. The pro rata amount for the Maintenance Fees is based on the portion remaining in the then current year's Maintenance term.

The licensee should consider softening the above clause by allowing for a transition period during which the licensee may continue using the software, even when the license is terminated as a result of the licensee's breach. The software may be mission-critical to the licensee, and it may take the licensee six months or a year or more to transition to a new vendor. During that time, the parties should seek to cooperate, even if they are involved in litigation.

UCITA

What is UCITA?

The Uniform Computer Information Transactions Act (UCITA) is a proposed uniform law for software licensing, online access and other transactions in computer information. Originally a proposed revision to UCC, Article 2B, UCITA was co-developed by the National Conference of Commissioners on Uniform State Laws (NCCUSL) and the American Law Institute over a 10-year period. The ALI co-authors all revisions to the UCC. In 1999, the ALI withdrew from effort. Without ALI's support, NCCUSL renamed Article 2B the "Uniform Computer Information Transactions Act" and has been promoting its adoption by the states.

Why should I care?

When drafting information technology agreements or selling goods with embedded software, UCITA implications should be considered:

- UCITA has already been adopted in Virginia and Maryland, and may be considered for adoption in other states.

- UCITA, where applicable, generally provides "default" terms which apply unless modified by contract.

- UCITA is intended to both restate the existing law as well as define how the law should develop. The NCCUSL's recommendations have already been given weight by courts even where UCITA has not been adopted. The comments to UCITA alone may be persuasive.

- There are new warranties and suggested disclaimer language. Art. 4. For example, an "as is" disclaimer does not affect the warranty of noninfringement and noninterference. Sec. 406(c). This warranty must be disclaimed by specific language.

What transactions does UCITA cover?

"Computer information transaction" means "an agreement or the performance of it to create, modify, transfer, or license computer information or informational rights in computer information. The term includes a support contract under Section 612…." Sec. 102(a)(11).

Embedded software: The NCCUSL recognizes that there is no "bright line" rule to determine when UCITA or the UCC Articles 2 and 2A should apply to the sale of goods with embedded software. NCCUSL has recently approved an amendment to Article 2 of the UCC to exclude "information" from the definition of "goods." The Preliminary Comments to amended Art. 2 and the Official Comments to UCITA state that chips (and software) embedded in goods in most cases will be governed by Art. 2. It is left to the courts to determine whether a particular transaction is an information transaction subject to UCITA or a goods transaction subject to the UCC.

What are some of the more controversial provisions?

E-mail Receipt

UCITA defines "Receipt" to mean "in the case of an electronic notice, coming into existence in an information processing system or at an address in that system in a form capable of being processed by or perceived from a system of that type by a recipient, if the recipient uses, or otherwise has designated or holds out, that place or system for receipt of notices of the kind to be given and the sender does not know that the notice cannot be accessed from that place." Sec. 102(a)(52) (II). Under Section 215(a) "Receipt of an electronic message is effective when received even if no individual is aware of its receipt." "This Act confirms the enforceability of automated contracting involving 'electronic agents,' but in some cases automation may produce unexpected, potentially oppressive results…. Common law concepts of mistake may apply…. In addition, in appropriate cases, unconscionably doctrine may invalidate a term….” Comment 3, Sec. 111.

Known Defects

One of the fundamental assumptions of UCITA is that “The complexity of software products makes them inherently imperfect” and “Minor flaws (‘bugs’) are common in virtually all software.” The warranty of merchantability does not require a defect-free product; as under UCC Sec. 2-314, the concept of “fair average” applies, with some products better and some worse. Sec. 403. In addition, failure to disclose a known defect should be dealt with under the law of fraud, which is not affected by UCITA. Sometimes, such a failure might be fraudulent—as when the seller knowingly makes a false statement about the product. But the failure to mention a known defect, even a material defect, may not give rise to fraud liability. Unless otherwise specified in a contract, UCITA does not hold software publishers accountable for *breach of contract* if they knowingly delivered a defective product without revealing the defect.

Transfer Restrictions

Under UCITA, almost all software-related transactions will be licensing transactions. A transaction can be a license even if the licensee is given title to the transferred copy. Sec. 102(a)(42). Under UCITA, “a term prohibiting transfer of a party’s interest is enforceable, and a transfer made in violation of that term is a breach of contract and is ineffective….” Sec. 503(2). According to the UCITA drafting committee, “UCITA is intended neither to avoid nor to contradict the large body of existing federal intellectual property law.” Others disagree. For example, the American Intellectual Property Law Association protested to NCCUSL that UCITA “eliminates the ‘first sale’ doctrine”, which allows the owner of a copy to sell it or give it away.

Under a recent amendment, a mass-market license may be transferred as part of a gift of a computer containing an authorized copy to a public elementary or secondary school, a public library or from a consumer to a consumer. Sec. 503(2)(C).

Magnuson-Moss Act

UCITA takes consumer software transactions out of the scope of the Magnuson-Moss Warranty Improvement Act and out of other consumer protection statutes that cover the sale of goods.

Who has supported UCITA?

Business Software Alliance
Digital Commerce Coalition
Information Technology Association of America
Equipment Leasing Association of America
Software and Information Industry Association
Computer Software Industry Association
Silicon Valley Software Industry Coalition

Who has opposed UCITA as drafted?

Association for Computing Machinery
American Committee for Interoperable Systems
American Library Association
American Society for Quality
Computer Professionals for Social Responsibility
Consumer Project on Technology
Consumers Union
Digital Future Coalition
Electronic Frontier Foundation
Free Software Foundation
Institute for Electrical and Electronics Engineers
National Writers Union
SHARE (IT users' group)
Society for Information Management
Software Engineering Institute

What's happening in Connecticut and New York?

In a letter dated November 13, 2001, signed by the Attorney Generals of Connecticut, New York and half of the other states, the National Association of Attorneys General expressed opposition to UCITA, arguing, among other things, that UCITA displaces more protective state consumer protection laws.

In March 2001, A07902 was introduced and supported by Attorney General Elliot Spitzer. It would amend Sec. 5-1401 of the General Obligations Law to provide that computer information transaction contracts shall be interpreted according to New York law where at least one party to the contract is a New York resident or has its principal place of business in New York. Any contract subject to UCITA may be voidable by any party and unenforceable as against public policy. This bill has not been acted on. Similar "bomb shelter" legislation is being considered in other states.

What have been the most recent developments?

In January 2002, an ABA Working Group issued a report highly critical of UCITA. The report outlined 10 specific areas of concern. At its Annual Meeting in early August, NCCUSL responded to most of the ABA's concerns by adopting substantive amendments in the following areas:

- Electronic Self-Help Banned. Vendors of "digital information," including software, may not disable the use of the information by electronic means if there is a breach of an information contract. Vendors have a remedy for a material breach of contract. Sec. 816. Licensor can exercise self-help "by taking possession of a tangible copy" without a breach of the peace. Sec. 815(a).

- State Consumer Protection Law Trumps UCITA. An information contract is subject to and may not waive any consumer protection provided in state or federal law, including requirements for conspicuous disclosure, unfair or deceptive trade practices, and laws related to electronic signatures and records. Sec. 104.

- Right to Criticize. Information contract terms that prohibit criticism of an information product are unenforceable. Parties may contract in a manner consistent with other law such as the law of trade secret or commercial disparagement. Sec. 105(c).

- Remedies for Known Material Defects. Remedies for a known material defect of a product are expressly made available as fully as for defective goods or services. Sec. 116.

- Reverse Engineering. An information contract may not prohibit reverse engineering that is done for the purposes of making an information product work together with other information products ("interoperability"). Sec. 118.

- Open-Source Provisions. Open-source is expressly not covered if only copyright permission (*i.e.,* to use, modify or copy) is given and is not part of a contract. If there is a contract, there are no implied warranties if there is no intent for profit or commercial gain from the transfer of the copy or from controlling use or distribution of the copy. This is intended to exclude from UCITA voluntary contributions and developments in the open source community. Sec. 410.

- Some changes were made for clarity and ease of understanding. The ABA Working Group's principal concern with UCITA was that it is "extremely difficult to understand." As a result, the desired result of a uniform law establishing a high level of clarity and certainty might not be achieved.

Other changes adopted by NCCUSL

- Reaffirmed that public policy may invalidate a contract or term. Sec. 105.

- Clarified that the enforceability of a choice of forum is a question for judicial determination. Sec. 110(c).

- Clarified that a licensee must have an opportunity to review license terms before being bound by a license agreement. Sec. 112 and 113. "A party is bound by the terms of a record only if it agrees to it, by manifesting assent or otherwise. Assent can be by authenticating the record or by other conduct indicating assent. However, a party cannot assent unless it had an opportunity to review the record before reacting." Comment to Sec. 112. If a license is paid for before the terms are disclosed, such as when terms are included inside a shrink-wrapped package, the licensee has a right of return, including compensation for reasonable and foreseeable costs. Sec. 209.

- Clarified that the law of fraud, misrepresentation and unfair and deceptive practices, as they deal with disclosure of defects, are not changed by UCITA. Sec. 116.

What is likely to happen to UCITA?

- Despite the recent changes, AFFECT (Americans for Fair Electronic Commerce Transactions) and other consumer groups continue to oppose UCITA. They maintain that the proposed law is unclear and too complex and favors the software and information industries. Specifically, they object to UCITA covering embedded software and not requiring disclosure of terms prior to payment, even if there is a right of return.

- Some have commented that the software industry may be losing interest because the most recent amendments either eliminated or watered-down the industry's favorite provisions.

- Since NCCUSL has responded to most of the ABA's concerns, the House of Delegates may consider backing UCITA at its midyear meeting in February.

- The President of NCCUSL has stated that if more states do not adopt it and the ABA does not approve, "a different approach to the subject matter will have to be taken." At some point UCITA may be downgraded to a model act, which does not require ABA approval, rather than a uniform law. NCCUSL also approved a change in the scope of UCC Article 2 to redefine "goods" to exclude "information," leaving it up to the courts to determine when to apply Art. 2 by analogy to computer programs and digital content. The ALI has yet to approve this amendment.

Where can I learn more?

Supporting sites:

The Official NCCUSL site: http://www.nccusl.org/nccusl/default.asp
and http://www.law.upenn.edu/bll/ulc/ulc_frame.htm
Carol A. Kunze's site and state by state tracking: http://www.ucitaonline.com.

Opposition sites:

Americans for Fair Electronic Commerce Transactions: www.affect.ucita.com
American Library Association: www.ala.org/washoff/ucita
Inforworld: www.inforworld.com/ucita
Cem Kaner's site: www.badsoftware.com

THE SPEAKERS

Thomas M. Pitegoff, the founder of Pitegoff Law Office, drafted his first software development agreement in the late 1970s for a business unit of American Express. In the 1980s, at his request, he was the first lawyer in the New York office of Hunton & Williams to get a personal computer on his desk; he was active in the CompuServe LawSig (before the advent of e-mail); and he was a member of the Computer Law Committee of the Association of the Bar of the City of New York. At one point in the 1990s, he handled all of the computer contracts for Philip Morris International at their Rye Brook headquarters. (The company has since moved to Switzerland.) During the dot com boom years, he did work for several computer and internet companies. He continues to do a substantial amount of computer contract work today. Tom is an ICANN domain name arbitrator for the CPR Institute for Dispute Resolution. In addition, he works with IT companies in the area in his capacity as a director of the Westchester Information Technology Cluster, a trade association of computer companies. He is also a member of the Computer Law Association and the Cyberspace Law Committee of the American Bar Association's Section of Business Law (Co-Chair of the International Subcommittee, 1998-2000). He writes and speaks frequently on computer law, franchising and international business law. His articles include: *Open Source, Open World: New Possibilities for Computer Software* (Business Law Today, American Bar Association, September/October 2001); *Conflicts of Law in Cyberspace* (Pike & Fischer, Jan. 2000); *The Inadvertent CyberFranchisor*, CYBERSPACE LAWYER (April 1998); *Negotiating a Computer Software License*, THE FRANCHISE UPDATE REPORT (April 1998). He is admitted to the New York bar. Tom's practice covers corporate and commercial law, including franchising, licensing, distribution, international business and computer law.

Leonard D. Andrew, counsel to Pitegoff Law Office, has more than 30 years of experience at IBM, retiring in 1999 as Associate General Counsel, Worldwide Industry Solutions Units. In that role, he was the senior legal executive for all legal issues related to IBM's worldwide sales of products and services to such industries as health, telecommunications, media, education, government, manufacturing and insurance. He was also counsel for IBM's large-scale computer division and personal computer division. From 1990 to 1994, Len was the General Counsel of IBM World Trade Asia Corp., in Tokyo, Japan, where he was IBM's senior legal executive for 21 countries and territories in Asia and the Pacific. His responsibilities included international competition law, intellectual property licensing and litigation, and commercial agreements. During that period, he negotiated joint ventures in India and China. Len acts as an arbitrator of complex IT disputes as a member of the Commercial, International and E-Commerce panels of the American Arbitration Association. He is admitted to the New York bar and was formerly admitted to the bars in the District of Columbia and California and as a foreign law practitioner in Japan.