Number Three

Contractors know contracts – except when it comes to software contracts, otherwise known as license agreements. And, like many contracts written by one party for the protection and benefit of that party, license agreements can often be very one-sided.

However, as with other contracts, crafting a balanced agreement can lead to a balanced long-term relationship with your IT business partners.

Software License Agreements

BY CHRISTIAN R. BURGER

In today's IT environment, construction companies can become party to several software license agreements in any one year.

Some are for smaller systems, with less risk involved, and a faceless software company nestled in the confines of Silicon Valley. Others are for substantial software products that include larger investments and carry greater long-term risk. It is these agreements that should be read, understood, and negotiated for a more balanced outcome.

Although not an attorney, I will explore the important components of software license agreements and suggest some alternative language. If requested before the agreement is signed, such language can usually be accommodated as an amendment.

Types of Agreements

When you license a new piece of software, particularly larger applications like estimating, project management, or accounting/job cost systems, you typically become party to two-four primary agreements.

The first (and foremost) of these, is the *license agreement* itself. By licensing the software, you become an authorized user of the application(s). Instead of owning the software,

you have – in effect – been granted an indefinite lease.

The second type of agreement is an annual *support agreement* with the software company. This may cover both regular software upgrades and enhancements, as well as telephone support.

Upgrades are usually required to keep the system operational (operating system upgrades) or to keep the system current with compliance issues (tax table updates). Enhancements are typically annual improvements to the system and can include new functions, screen layouts, or data fields.

Finally, although not covered in this article, there may be a *professional services agreement* and/or a *data conversion agreement* relative to implementation. The professional services agreement covers the fees, terms, and conditions for the vendor's training and consulting staff. The data conversion agreement covers the services required to migrate your data from the legacy system to the new system.

If you sign a standard **LICENSE AGREEMENT,** your accountant, auditor, or other outside consultants **cannot ACCESS the system** without **being in breach** of the **AGREEMENT.**

These four agreements make up the bulk of the licensing process for most applications. Though your software vendor may initially present standardized versions of these agreements for your approval, there is more flexibility to the process than you may imagine – especially before the agreements are signed.

Preparing for Contingencies

After 17 years of watching construction companies enter into these agreements, I've witnessed the following events that could impact a contractor's license and support agreements:

1) The software company could go out of business.

2) The contractor could become disenchanted with its software vendor and not want to pay for support.

3) The contractor may inadvertently breach the license agreement, thereby jeopardizing the entire agreement, including the ability to continue using the software.

4) The software company or the construction company could be acquired by another company.

5) The contractor could acquire a company that would remain a separate wholly owned entity, but use the contractor's system.

6) The contractor could enter into a joint venture and want the new entity to process its books on the contractor's system.

7) The contractor's outside auditors or CPA may want to access the system during an audit.

8) The software developer may experience increased costs or declining sales, necessitating increased support fees.

These are just a few of the possibilities that need to be addressed in the license and support agreement review process. In construction, the name of the game is risk mitigation. Software licensing is no different.

License Agreement

To begin with, let's look at the license agreement from the software developers' point of view. Many software developers, particularly closely held private firms, have modest balance sheets. All of their "equity" is tied up in their software.

So, you can understand why license agreements are crafted to protect that all-important intellectual property. After all, it's the source of the software developers' income and the reason for their reluctance to release their source code (the programmer's instructions to the computer).

Price

Probably the most commonly negotiated component of any license agreement is price. Software companies are accustomed to providing a "list" price for their software when submitting a proposal. Rarely do you see a discount until you get closer to finalizing the contracts or closing the deal.

Don't Discount the Discounts

Discounts can be significant, depending on such factors as:

- The total number of users,

- The caché associated with the customer's name in the market place,
- How hungry the software company is, and (even)
- How near the company is to a quarter or year-end.

What You Need to Know

In the beginning, you do not have to license all the "seats" or "named users" you are ever going to need. In fact, from a cash perspective, it can be to your advantage to *only license what you need for the implementation, and then add to that once the system is ready for live processing.*

However, make sure that you include a clause in the agreement that allows you to *add licenses at the negotiated prices for a period of 18-24 months*. Also, if there are applications or modules you do not currently need (but may want to implement in a later phase), get that pricing locked in upfront.

Also, you will have further pricing leverage if an application or suite of applications is new to the market or relatively untested, and you are willing to assume that risk.

Defining "Customer"

Some software license agreements do not define the "customer." *The customer should be defined and, in each case, your company named.* Ideally, the definition should include the uppermost entity and all subsidiary entities, wholly owned subsidiaries, and entities with common ownership – the broader the better.

This is not unfair to vendors because you will still need to add more licenses as additional users are added. It simply protects your company if new entities are added or acquisitions anticipated. It also means that your company can continue to use the software in the event it is acquired.

If joint ventures are routine or anticipated, ensure that the customer definition does not exclude such arrangements.

Source Code

In these days of failed or acquired software companies, it is not unusual or unwise to worry about source code.

Most software is compiled to prevent anyone outside the software company from making changes or accessing the actual programs. However, *having the software on deposit with a neutral third-party in the event something happens to the software company* (bankruptcy, acquisition, etc.) does provide a degree of security.

The Escrow Agreement

This is the purpose of the escrow agreement. It defines the escrow agent who holds the software, how often the source code has to be updated, and under what circumstances the source code is released to the beneficiary (customer).

This service is usually paid for by the software vendor, but the customer must be added to the escrow agreement as a beneficiary.

Breach of Contract

The definition of "breach of contract" is also very important since it can be grounds for revocation of your company's license. Legally, this means that the software company could revoke the license for any breach of agreement (no matter how small) and take back the software.

Make sure the license agreement *defines what constitutes "breach" in general and "material breach" in particular.* The agreement should also provide for *written notification of breach and a reasonable "cure" period (say, 30-45 days).*

Some vendors may resist a cure period if the breach is egregious, and that is generally acceptable. If your company is doing something that actually harms the vendor (such as selling the source code or providing access to competitive companies), 45 days to cure is not appropriate.

Jurisdiction

The jurisdiction for disputes is usually defined as the state of the software developer. Though some contractors have successfully changed jurisdiction to their own states, this is somewhat less important than agreeing that *disputes must be settled by arbitration*. (Also, the chosen arbiter should be familiar with intellectual property issues. This is not usually difficult to achieve.)

Number of Installations

Most software companies prefer that their software only be installed on one machine. They realize that this is somewhat risky and that a backup copy may be needed on a second machine in the event something happens to the primary server.

I prefer that *the customer be allowed a third installation for testing and training, which could exist on a separate server.* Even though this does not deprive the vendor of additional income, some are reluctant to authorize it in writing.

This third "training and test environment" allows your company to use the system, make configuration changes, and test enhancements – all without disturbing your actual production environment.

Including Outside Parties

If you sign a standard license agreement, your accountant, auditor, or other outside consultants cannot access the system without being in breach of the agreement.

Again, this is a rarely invoked clause, but I prefer something like: *"any and all parties, working on behalf of the customer, can have access to the software system provided they are subject to the same non-disclosures as the customer."*

Importantly, outside parties (particularly those who work with many companies and many systems) should not be allowed to take system documentation offsite. This protects the vendor's most valuable asset (information regarding the software) from piracy.

In short, the vendor is saying that **its SOFTWARE PRODUCT** will work according to **design and specification . . .** but not necessarily to your needs. That is your job: **caveat emptor.**

This is why **proper DUE DILIGENCE** is so important.

Warranty & Limits of Liability

Most software license agreements employ legalese that specifically limits or excludes any kind of warranty relative to the product. These limits are usually specific to damages, as well as to "fitness of purpose."

Fitness of Purpose

A fitness of purpose clause exists in the contracts of most significant software purchases. It stipulates that the customer is responsible for ensuring that the product meets its requirements or intended use.

In short, the vendor is saying that its software product will work according to design and specification (often documentation), but not necessarily to your needs. That is your job: caveat emptor.

This is why proper due diligence is so important. Using an RFP process, software demonstrations, and reference calling (possibly even a pilot process) helps ensure that the product performs all the functions you need it to.

Liability Limits

So, having done your due diligence, what if the software does not work? The software vendor's liability is nearly always limited to the amount of money paid for the license – in other words, there are no consequential damages.

The lack of consequential damages is rarely negotiable since it is fundamental to the protection of the vendor company. Instead, negotiate language that provides for: *"the full reimbursement of license fees paid in the event the software does not work according to system specifications or documentation within the first 90-120 days."*

Warranties for a shorter period do not provide enough protection since it is unlikely that the software can be adequately tested in less time.

The "New" Factor

If the software is relatively new and/or untested, negotiate a money-back guarantee for a period of up to one year. When new software is involved, most vendors are willing to consider such terms in order to get early adopters on board.

If there is a function critical to your organization (but not included or completely operational in the field-test), make sure you have an amendment to the primary agreement that states that the function will be included in the final version.

Support Agreements

Support costs are almost always based on the price of the license fee. This annual fee covers maintenance,

enhancements, and telephone support and is usually expressed as a percentage (15-20%) of the software "price."

There are three important things to know relative to the support agreement:

1) ***Know what it includes.*** Some support agreements are for enhancements and support only, and do not include telephone support. This is a significant difference that, over time, could make one system much more expensive than another.

2) ***Know when the support agreements commence and are billable.*** Most software maintenance agreements begin immediately when you license the software; however, the date of the first payment can vary. Some support agreements are priced to include the first year's support once the software is purchased, while others begin 90 days after the installation.

3) ***Find out the basis of the fee.*** Most are based on the list price of the software at the time of the billing, sometimes written as "the then current list price." This means that as software prices go up, so does the amount you pay for maintenance. I prefer that ***the support percentage be applied against the price paid, usually a discounted amount.***

Support Increases

Finally, know that the percentage charged is frequently not capped. So it can feasibly go from 15% to 18% to 20% (and even 25%) in a short period of time. Admittedly, software companies don't do this unless they have to, but you can protect yourself with a simple clause that:

- Caps the increases for a period of three or five years, or
- Establishes the amount that can never be exceeded, or
- Indexes the amount to an economic indicator, such as the CPI.

From the software developer's perspective, increases are not completely without merit. Remember the high-tech boom of 1999 and 2000, when the inflation rate for technical resources and services was increasing at a much higher rate than the rest of the economy?

Keep in mind that your software vendor's economic health is critical to its ongoing success and, therefore, to your ability to keep using an ever-improving software system.

Third-Party Issues

If there are many third-party software programs involved, make sure that the language of the primary support agreement ***obligates the vendor to maintain the interface and/or interoperability with the other third-party programs.*** Otherwise, if a new release comes out, the interface may no longer work.

Further, it would be best if the software company was obligated to maintain that relationship and interface with the third-party program until such time that they cannot, and then they must find a replacement product for the original application. This is especially important if the third-party application is a critical part of the overall system working or meeting your requirements.

Example

For example, some heavy/highway system vendors have products that support the interface of scale ticketing applications to the billing, inventory, and job cost applications. Without that interface, the office's workload increases exponentially.

The accounting software vendor must be responsible for maintaining a compatible interface with the scale ticket vendor. A long-standing relationship between the two companies is usually a good sign that they can, and will, continue to work together.

Terms

Payment terms are also negotiable. There is usually a request for a deposit at the time the agreement is signed. Some software developers ask for all monies upfront before the software is delivered or, slightly better, when the software is delivered and installed.

I prefer a ***graduated payment schedule, with percentages paid out upon completion of certain milestones.*** For example:

- 20% deposit on signing the agreement (good faith),
- 30% upon delivery and installation,
- 40% upon substantial completion of training, and
- the final 10% at some significant operational transaction (first estimate, first payroll, first live, etc.).

This latter clause usually needs to be accompanied with a phrase that suggests ***"or six months, whichever is sooner"*** to protect the software vendor from being delayed in payment due to circumstances beyond its control.

Negotiating

As with any negotiating, there are a few key points:

1) Begin by knowing what you want, what your risks are, and what you are trying to protect.

2) Also, remember to look at this as a long-term relationship with a valued business partner.

3) Have an understanding and appreciation for your software vendor's interests as well.

4) Decide which items are critical and which you are willing to concede.

5) Also, know your degree of leverage. This is not an entirely one-sided transaction. You, as a contractor-customer, offer something to the software company in the form of market share, customer reference, and an ongoing revenue stream from your support fees. So, use that power when deciding how best to get what you want.

Conclusion

Ninety-five percent of all software license agreements are never referred to once they are signed and put in place. It is only occasionally that you need to dig back into the document and remind the vendor (or your own company) what was promised. Taking the time upfront to cover your bases will ensure that you are not unpleasantly surprised after the fact and sets the course for a favorable long-term relationship. BP

CHRISTIAN R. BURGER is President of Burger Consulting Group in Chicago, IL. Before establishing his own consulting firm, Christian was a management consultant for FMI for eight years and a client manager for J.D. Edwards for one.

He received a BS in Accounting from Ball State University in Muncie, IN and a MA in Liberal Studies from Northwestern University in Evanston, IL.

Co-Chair of CFMA's Technology Committee, Christian is a member of CFMA's Chicago Chapter, a previous author for this magazine, and a frequent speaker at industry events across the country.

Phone: 630-510-1875
E-Mail: crburger@burgerconsulting.com
Web Site: www.burgerconsulting.com