by Jerry Askew of Askew Network Solutions

::Software Strategy with OSS

Of all the IT decisions firms make, those regarding software easily have the greatest effect, positive or negative, on the organization. While the portfolio of applications tends to be nearly identical among firms (*e.g.*, accounting, document management, word processor, etc.), the particular software packages used to provide those services vary. Perceived business needs dictate which particular pieces of software are needed. The software, in turn, dictates the kinds of systems and processes necessary to support it and, ultimately, the hardware, personnel, time and space required.

Despite the fundamental importance of software, many business executives do not have a good grasp on the nature of software, nor how it fits into the overall IT ecosystem. Similarly, IT executives frequently have only minimal visibility into the operational and administrative process that the applications are intended to serve. These factors combine to make it difficult indeed to make proper strategic decisions regarding software.

As an alternative to traditional software procurement models, open source software (OSS) offers a variety of advantages that help to minimize or avoid the pitfalls inherent in the life cycle of an application.

Overview of Open Source

As a concept, OSS is relatively simple to describe. The source code of the software is made available so that any interested party can inspect it and modify it. In contrast, the source code to commercial software is privately held by the company or entity that owns the software. Recently, a hybrid known as visible-source has appeared where a company allows the inspection of their source code but only under contract, and modifications to the source code are prohibited.

While the concept of OSS may be simple and straightforward, the implications are not. OSS is a radical departure from traditional software procurement models and warrants careful analysis to determine when and if it is the right choice.

Software in Business — A Comprehensive Look

Earlier in this article, I alluded to the foundational role software plays in a business organization. The overall impact is far greater than many people realize (or are willing to admit).

Circa 1990, desktop computers were making the transition from being a luxury item to a necessity. Attitudes at the time were that computers and software existed to serve current business processes. An executive once told me that the company would not alter its processes to accommodate computers — the computers must accommodate the business. Adapting business processes in order to incorporate technology was seen as the tail wagging the dog. What this executive did not realize was that the dog was not a necessary part of the business and would soon become obsolete . Only organizations that embraced and incorporated technology could realize the full business benefits therein.

The viewpoint that computers are somehow adjunct to business is still surprisingly prevalent, even as most business processes have yielded to automation out of sheer necessity. As a result, departments are staffed and business is conducted around the capabilities and limitations of the technology in use.

Traditional total cost of ownership (TCO) analysis fails to capture much of the impact software has on business. In order to make a proper strategic decision, it is necessary to contemplate all of the organizational consequences of a given course of action. The following sections explore various areas where OSS changes the rules of the software game and offer practical insight on the resulting implications.

Solution Research

After a business need has been identified, most likely by a management committee, the systems necessary to provide the solution must be selected. During the research phase, entire classes of solutions will be ruled out based primarily on their suitability to the task and a ballpark TCO. These two factors are interrelated — a less than perfect solution may be selected if the TCO is substantially lower.

The challenge is that both of these factors are difficult to quantify, and the only data available in most cases is either provided by the vendor (which is necessarily suspect) or from experiences at other firms (which is a limited sampling, especially with newer technologies). This tends to have a chilling effect on the implementation of newer technologies, which otherwise might be of substantial benefit to the organization.

Due to the initial investment generally required for commercial solutions, research must be complete and convincing. In addition to being very time-consuming, this process can foster the premature formation of opinions and may create an environment ripe with political risk. Initial expenditures may require approval from a committee, presenting the opportunity to rehash previously decided issues. This process is imperfect but necessary.

> If the support offered by one vendor does not suit your needs, you can simply find another vendor who is a better fit.

OSS presents distinct advantages at this stage of the game. While TCO is frequently on par with proprietary software, the initial acquisition cost is well within discretionary spending limits. This allows a solution to be brought in and tried at any scale with very little risk. Stakeholders are able to form opinions based on direct experience. Potential issues can be explored, demonstrated and discussed. Without the benefit of a tangible system, issues will be largely speculative — as will the solutions — making reconciliation difficult. Commercial software vendors may offer demo or trial periods, but these are frequently too limited or time-constrained to gain a full understanding of the system.

In the ideal OSS scenario, committee involvement occurs after the solution has been proven and demonstrated, at the point that larger financial commitments are ready to be made. The reduction in speculation and hand-wringing will save the organization much time and frustration.

Acquisition Costs

The initial cost of enterprise-grade commercial software is high. This presents a significant risk since the firm is financially committed long before the software is known to provide an effective solution. The solution to this dilemma is a carefully negotiated contract with specific performance requirements and guarantees. This is not a perfect solution, however. Much of what is truly required will only be known after implementation has begun. Further, performance requirements must walk a fine line between being too subjective and being too restrictive.

It will come as no surprise that software pricing models are generally structured in a way that maximizes vendor profitability. In many cases, this prevents the apples-to-apples comparison between products. In some cases, necessary features may be unbundled in a way that makes the product seem less expensive. Only after you go into production will you discover that you do indeed need that “Advanced XML HyperInfo Enhancement Module” that had earlier appeared unnecessary.

OSS places a completely different spin on software acquisition costs. Instead of a large initial licensing fee followed by contracted maintenance, your investment grows as your use and reliance on the solution grows. Your commitment stays in sync with the utility and benefit you are receiving. If at any point you discover that the system is not meeting your needs, a large capital investment is not in the way of turning around and heading in a different direction.

An added benefit of avoiding a large initial investment is that you can likewise dispense with the complex contract. You may contract with a vendor for implementation services, but the amount of money at risk is in line with the services provided, not in an intangible and substantial license fee.

There are very few surprises with OSS software. Whatever it says it will do, it will do. There are no additional modules to buy, and there is no incentive to make the product seem to be more than it is. In fact, the incentive is to be conservative in stating the capabilities. If an issue exists — whether a bug or a limitation — it is likely to be well-known and discussed in publicly available forums.

Support

Software support services are typically negotiated at the time commercial software is purchased. Again, there are a number of risks, the solution to which is — you guessed it — another contract. The underlying risk is that the vendor is a single source provider of support. Specific items that must be addressed include: support cost increases, scope, response times and escalation, among others. If support is inadequate, you have little recourse other than to tolerate it or migrate to a competing product.

Support is an area where OSS has much to offer. The more popular packages, such as Linux and Apache, have big-name companies ready to stand behind them. Nearly all OSS packages have a community of smaller support companies built around them. The advantage is you have the flexibility to find the support vendor that works best for your company. For instance, you may find a support vendor that has expertise in the legal market or more specifically in a corporate legal practice. If the support offered by one vendor does not suit your needs, you can simply find another vendor who is a better fit. Since these support vendors tend to be smaller companies, it is likely that you will be treated as a very important customer — a status that would be difficult to achieve with a large commercial software vendor.

Utility

Software is acquired in order to fulfill a particular business need. Depending on the decisions made during the purchasing cycle, the software will fill that need to a greater or lesser degree. As the new system comes into heavier use, it is expected that some of the original concepts about the system will need to be revised. Often, similar issues have arisen with other users of the software, and the capability to satisfy these needs is available within the package. In the case of commercial software, this additional functionality may come at an added cost. In some cases, the software may not be able to accommodate the newly discovered need.

If a need develops that is not addressed by a piece of commercial software that you own, you have a few options. You can request an enhancement, negotiate for custom programming or implement a third-party add-on. A potential pitfall here is that the vendor may have business incentives that are in opposition to your needs. An example of this would be a vendor that has product tie-ins or strategic alliances with other vendors. If your need encompasses integration with a competing adjunct system, the vendor may either refuse or charge a disproportionate fee for the modifications.

The development of OSS is driven purely by need and utility. It is not uncommon to find edge-case functionality (*i.e.*, features that are important to a very small part of the overall community) in open source projects. This occurs because any part of the community, no matter how small, is able to contribute to the project. Because of this, most needs have been anticipated.

If required functionality is missing from an OSS solution, it can be easily added. This is generally done by "sponsoring" the necessary development. Sponsoring is simply contracting a programmer to add the improvements that you need. The cost will be competitive since you can contract any programmer to do the job (the lead author is generally the first choice).

While an OSS product may have preferential support for certain adjunct products, it is never at the exclusion of other products. In fact, any preference expressed in the project is simply a reflection of the needs of the community, not the result of a vendor's strategic business relationships. Due to the importance of interoperability, OSS systems leverage and adhere to published industry standards to a far greater degree than commercial software.

Opportunity Cost

An often overlooked part of the software picture is opportunity cost. Opportunity cost is basically the amount of money the firm either fails to realize or outright loses as a result of the software not doing exactly what is needed. Some familiar manifestations of this would be:

> Additional staffing requirements due to inadequate reporting or process inefficiency
>
> Improper (or complete lack of) profiling due to ease of use or feature availability issues
>
> Delayed or lost billing
>
> Failure to obtain new business due to an inability to meet client requirements or outclass a competing firm
>
> Loss of existing business due to shortcomings in a conflicts system

While hard to quantify, opportunity cost can be of great significance to a firm.

Many of the factors that contribute to the opportunity cost are given attention during the process of selecting a software solution. The challenge is that it is not possible to accurately predict people's behaviors or the developing needs of the firm or its clients. As a result, the opportunity "gap" tends to widen over time.

With commercial software, this is simply par for the course. Regardless of which vendor's solution you use, there will be trade-offs, and the solution will never be perfect. For this reason, larger companies frequently turn to custom development, which, unfortunately, is not an option for most law firms. OSS offers an alternative in allowing what you might call "targeted" custom development. In other words, if you later find you need to make a small change to one aspect of the software, you are free to do it. This kind of freedom does not exist with commercial software and can become a significant, yet well-hidden liability.

Upgrades

Another aspect of support is upgrades to the software. Support contracts will typically include patches, updates and new versions of the software. Several years ago some vendors introduced the concept of a "new product" into the mix. When supporting technologies advanced, such as the next generation of Windows or the availability of server-side SQL databases, these vendors repackaged their product under a slightly modified name and expected their customers to buy the product all over again.

Commercial software vendors are constantly searching for new and improved sources of revenue. In the commodity software market, vendors have begun to leverage their installed base in pursuit of new revenue streams. It is increasingly common to find that software upgrades contain subtle compatibility changes or unwanted bundling in an attempt to influence end-user behaviors. While this is a highly effective business strategy for the vendor, it generally does nothing to help you meet your business objectives and may even be a hindrance.

Open Source software is free of these annoyances since the profit motive lies with the supporting services, not with the software itself. Furthermore, open source software cannot effectively be manipulated to create benefit for one party at the expense of another. A fundamental aspect of OSS is that it cannot be controlled or manipulated by the copyright holder or anyone else. If such control is attempted, the disaffected portion of the community simply creates a new copy of the software and continues development according to their own needs. This process is known as a "fork" and can be thought of as the OSS version of competition.

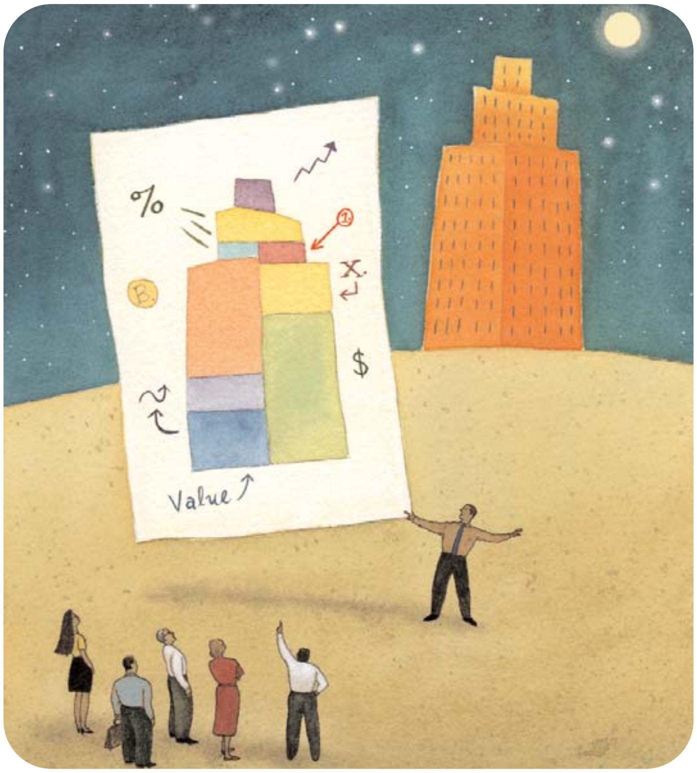
Generally, a fork occurs as a result of differing views on the future direction of the project. In some cases, a fork occurs because the original project is no longer viable, and the fork will give new life to the project. In other cases, both branches of the fork are viable but intended to service different needs or different industries. In any case, the result will tend to increase your options as opposed to limiting them.

Maintainability

In selecting and implementing a software solution, the firm takes on a significant commitment. The investment in implementation and training often dwarfs the cost of the software itself. As people become proficient with the software and new processes develop around it, the costs of switching to another solution quickly become prohibitive.

A readily identifiable risk with commercial software is that the vendor may cease to maintain the software. This situation can result from any number of causes including bankruptcy, casualty, sale of the company or simply a strategic change in direction. This risk is present regardless of the size of the vendor. Consequently, purchase agreements for high-cost and/or strategic software will often include provisions guaranteeing a certain support lifetime as well as source code escrow.

OSS has distinct advantages in this area. Since the source code is published, escrow arrangements are obviously unnecessary. More importantly, multiple companies and individuals are invested in and actively working with the code base at any given time. If any one company or author, even the lead author, abandons a project, someone else will likely take the author’s place. At the very least, there is a community of developers who are familiar with the project and can perform work at a moment’s notice. By contrast, it would take a significant amount of time for an independent programmer to come up to speed on proprietary source code that has been released from escrow.



Ease of Migration

It has been said that “the only constant is change,” and this is certainly true of business. At some point, the software that was purchased will no longer meet the firm’s needs, and the time will come to migrate to a new solution. This can occur with OSS as well as with commercial software, although the flexibility of OSS offers a possible option to migration. By sponsoring a modest set of improvements, you may be able to mitigate any identified shortcomings and save your firm the significant expense of migration to another solution.

There are times, however, when migration is unavoidable even with OSS. Typically, the reasons for this migration are shared with other members of the project community. To that extent, tools and expertise will be available to help with the migration. In some cases, the project itself may include tools for migrating to other systems. With OSS, there is no incentive to keep you captive to a solution that doesn’t fit your needs. In addition, the focus on the use of standards will ease migration among OSS solutions.

By contrast, commercial software vendors are unlikely to offer any help in migrating away from their product. In many cases, the system will be designed in a way that discourages migration. Making it hard to leave can be an effective retention strategy, and often requires less investment than making you want to stay.

The Right Decision

The OSS model offers a number of advantages over commercial software at various stages throughout the solution life cycle. Some of these advantages are difficult to quantify, but they directly address organizationally significant issues that typically arise in high-risk and high-cost projects. Depending on the application, OSS may be an available option. While we have discussed the general differentiators of OSS and commercial software, the crux of your decision will lie in the specific capabilities of each solution. In those cases where OSS solutions meet your requirements, understanding the rules of the game will help in making the right decision for your firm.

This article was first published in ILTA's June, 2006 white paper titled “Open Source Software — The Door Is Open” *and is reprinted here with permission. For more information about ILTA, visit their website at www.iltanet.org.*