Software Considerations: Use What You Have or Start All Over?

How to evaluate the positive aspects of your current system

Often, I've been called in to evaluate existing systems. Most of the time these are complex and have been developed over a long period of time. Also, too frequently, they have been built by multiple teams and are not well documented. Nevertheless, they usually manage to get the things done for which they were designed (i.e. processing orders, tracking inventory, etc.).

The process begins by asking the question: what are we trying to accomplish and what makes us want to get rid of the existing system? Sometimes there are fundamental reasons to change. One client said, "Anything would be better than our present system... the person who wrote it was *insane*!" "Everyone says that about their MIS people," I replied. "No, you don't understand, the person who wrote the code for our system is across town in the asylum!" That is probably a pretty good reason. Another client has an order processing system that has no customer numbers. Each order is an entity unto itself. As long as it ships correctly, everyone is happy... but just try to build sales history and you loose track of everything at the customer level. That is also a pretty big problem.

On the other hand, most clients will say things like: "We don't like the screens, they aren't user friendly (just green on black letters)." "The reports aren't flexible enough." Or, perhaps the best one, "MIS doesn't seem to understand what we want." These poorly articulated reasons often suggest that the problems haven't been defined well enough. Step one is then to list the problems with present systems. Often you will find that the system does in fact process orders... even if it isn't flashy.

Key advantages of staying with the same system.

Though the old system may not seem flashy, there are always many good reasons to stick with it. First is that you know what the problems are! This may sound trivial, but any system you are evaluating looks good on the outside. The fact is that you won't even suspect the real problems between the way it works and the way you are used to doing things until well into the installation process. Most problems won't even surface until you have been running it for a while... and then it may be too late.

The old system is understood by all. There are many aspects to software design which are not black and white. Though you probably don't realize it, the way your system works now is the only way most of your employees have handled situations. It is certain that, though the new system will handle the same problem, it will approach it differently. And with this difference comes vast potential for misunderstanding.

With any new system, employees will need massive training. Order processing will always understand this, their people will need to be comfortable with the (hopefully) minor modifications to the way orders are handled. But where the problems arise is in the higher level systems. People are used to interpreting monthly totals in a certain way, inventory turns have always been calculated like this, overhead allocation always weighted more heavily on the existing customers... why doesn't this new system do it the same way... With a new system, you change the way you measure many little things. Its like getting a new bathroom scale and thinking you've suddenly lost weight. Interpretation may be way off and you don't even realize it.

We tend to magnify the reasons why the existing solution is terrible. We also tend to minimize the problems which may arise with the new system. The fact is that we will not know the problems with the new until we jump in with both feet.

Reasons why change is always more difficult than it seems.

Once again, it is difficult to plan for the unexpected. Since we don't realize the hidden problems in the new system, we don't plan for them. Often, while the system evaluation is in progress, people either don't put their heart and soul into the process. Or cause just as big a problem by listing everything (including the kitchen sink) which the system must do. The long lists frequently lack perspective and priority and therefore are of little use in system modification and design. "They don't really know what they want." And, therefore, lets not worry about their requests. Both these problems come home to roost when we start trying to build the new systems.

On top of the design considerations, there are fundamental hardware and software challenges. The competitive climate in computing pushes developers to stretch their capabilities. Often we interpret their claims as fact instead of future. One client was expecting to integrate a $250,000 peripheral into their network, but just to be safe, they put a sentence in their lease from the manufacturer that if this capability wasn't working within three months, they didn't have to keep the machine. Well, it took almost two years for this system to really work and that clause saved their company from disaster. To be fair, it isn't that the promises are lies, but more often, with the complexity of today's systems, testing is limited to laboratory conditions. No one has ever really gotten this system running on a configuration exactly like yours!!!

Why your operations mainframe will not meet your marketing needs.

If you system is processing orders and the service agreements aren't prohibitively expensive, the fact that your system isn't pretty may actually be a plus. The world is littered with the bones of systems that looked great but didn't perform up to specs. One client brought in a team of consultants from their mainframe supplier. They were going to fix everything about the system. They spent two years in the redesign and programming. When they tested it, they found that it would take almost 48 hours to process one day's orders. As you can see, they would fall behind very fast.

Operations is concerned with efficiency, stability and no down time. Your order processing system should be somewhat resistant to change, very fast and capable of handling your peak volume. The fact that it isn't the most flexible reporting tool, or doesn't allow marketing to turn on the dime should really not be a big issue. Consequently, it is no surprise that MIS is not as responsive to marketing... marketing after all is rarely mission critical. If the only thing wrong with your system is the reporting, it is very easy to fix that at the PC level.

Simple rules for migrating applications from obsolete technology.

First, try to keep as much the same as possible. One client was told that because their hardware company was in bankruptcy, they should scrap their application written in BASIC and move to client/server technology. They were quoted several million dollars and that didn't even include programming (much less training, documentation, etc.). Another client was sold new hardware to solve their capacity problem. They spend several hundred thousand dollars before their developer through in the towel. The software was never finished and the hardware was never turned on... but they could only get a few cents on the dollar for the equipment. The first rule is then to make sure that your systems are really obsolete. Even the most arcane manufacturers are now providing some way of escape. Even when they go bankrupt, someone usually arises to service the equipment.

The two clients mentioned above were both told that they were in serious trouble. Get a second opinion! In both cases, they were running on UNIX which is one of the most transportable systems. They both were able to buy compatible hardware which would solve their capacity needs. Though it was from different manufacturers, it could still run their software. It is worth almost exhaustive research to find a way to use your system on a new computer rather than starting from scratch... and there almost always is a way.

If you can't get out easily, there are more and less critical aspects of your system. You can migrate functions of your system one at a time without jeopardizing your whole company in the process. First, pull out the marketing functions (because if they crash, they can always wait a day or two). Create a mirror of your customer and order files on a marketing system with regular automatic updates. You can even have this done at a service bureau. Often the biggest complaints can be solved just by giving Marketing a more flexible environment for their reporting and query needs (their own sandbox to play in). Next you can move some of the accounting or inventory management. Only after you have the side pieces working, should you consider rewriting the core functions. Sometimes, removing these tangential systems will allow MIS to get rid of excess data and processing. This will give them the space and time to recode the core functions in an orderly way.

Only as a last resort should you consider junking your present system and moving to some other package. As we've discussed, it looks far better in demo's than it will when you get passed initial implementation.

Advantages of some of the emerging systems.

After throwing this scare into you, we should now discuss some of the new technology. Many Business-to-business direct marketers are now able to process on small LAN based systems. These offer considerable cost advantages over the more traditional platforms. We also expect to see more and more client/server systems in the next few years as the LAN systems are expanded to use the higher security and power of UNIX servers. These systems typically allow file structures to be expanded without any software modification. They allow users to access data with point and click query technology. They provide limited use of point and click technology at the TSR's desk (they look nice, but may not be dramatically more functional). The fact is that today's PC's are more than powerful enough to run most businesses. This area is very open (meaning you can move your software to an ever expanding array of hardware platforms). You can often get programming support from high school kids (this doesn't mean that you should, but there are many more people who know dBase than who know RPG).

Ways to assess the pro's and con's of new vs. old.

To assess the old, first be very clear about what is wrong with the old system. Realize that it is almost always easier to fix what you have than to install a new system. Challenge everyone to think about "work-arounds" ways of solving old problems with new ideas. Second, differentiate between thing which the old system does not do and things it cannot do. If it is a flat file system, it will be difficult to make it relational. If it is character based, then it will not look graphical. If there is no customer ID then it will take massive changes to create one. Next see if those things which cannot be done are really part of the core requirements (most often they are not). Finally, build realistic estimates of the time and money needed to fix what can be fixed (or move what cannot be fixed). Many companies have not expended nearly as much effort to fix systems as they will in putting in a new one... but they will happily take on the monumental task without serious consideration of the easier path.

The fix budget becomes the benchmark. See whether there are full featured alternate systems which are less expensive than your budget. Then multiply what ever number you come up with by about three (because that is probably closer to the real number). Talk to existing users of the new system. Talk especially across departments. MIS may love the new because their job is easier, but don't forget to talk to the same people who are now most critical of the new system. Look at the details, the file structures. Does it seem to match the types of things you are keeping now. If there are either far less or far more fields than your present system, it probably is not a tight fit. The more broad the installed base, the more features which will be included (which you probably don't need). The newer the technology (more open, more PC based) the few sites and the shorter the track record. Don't expect the wizbang technology to have been in place for more than two or three years (because that's about how long Windows has been generally accepted). Don't mistake graphical (slick interface) for quality and/or even high tech. Many systems which have crude core functionality have been spruced up to look dazzling.

Finally, exercise a high degree of caution and skepticism. If you think your business is pretty complicated (i.e. you do both manufacturing and distribution, have multiple divisions or have pushed the organizational envelope in any particular direction) it probably is. Don't be swayed by claims that, "we can fix it up here and there, your business is just like everyone else." On the other hand, if you are one of many similar businesses in an industry, there probably is an off the shelf package that can work for you. In some cases, restructuring your business to fit the system may actually be a plus (you may not have that great a structure to begin with). In any case, you should be aware of the differences between the way you do business and the way the system expects business to be done.

Ways in which new is not better.

New is not better if it doesn't save time, money or make you a lot smarter. New is not better because the box is smaller and quieter. New is better if it lets you tailor your business to the needs of your customers... but be careful... most customers would rather you saved your money and passed it along to them. They would rather get it fast than have fifty delivery options. Keep it simple... new may add 400 extra reports, but maybe a few changes to the old would give you one or two good reports.

Pitfalls of "Object Oriented" and "Client/Server" technologies.

It took us almost 18 months to get comfortable with the new OO development tools. Another year before we were really productive. That was in an environment with Ph.D. systems people who had been programming since they could type. Your shop may not have the time to reinvent all the wheels. Granted, the tools are getting easier, but they are also getting different... and some of your best people may not be very comfortable with the change. There is little merit in conversion for its own sake. I don't have the time to detail all the reasons why the higher level language is more difficult, suffice it to say that you still have to type code and that code still has to work. I'm not sure whether the productivity promises will ever be delivered.

Client/Server is a conceptual dream... letting the PC work along side the powerful mainframe type server. But in real life, whenever the PC gets involved, things slow down fast. We are now taking more and more of the pure C/S applications and rewriting them into mainframe type solutions which then talk to the PC at the end of the process. As we said above, it is easier to "pretty them up" than to recode everything. Don't be overly concerned with the elegance of the structure, concentrate on the function.

How to say "No" to your MIS and/or Marketing System Requests.

Working in both environments, I see two reactions to change. Either we want the latest and the greatest "fully relational, object oriented, client/server, massively parallel, open system, case tool driven, graphical interface." Or we want to stick with the old until it is absolutely unsalvageable. Both paths are wrong. I feel that Marketing is much more often guilty of the "silver bullet" syndrome. Thinking that this one extra field in the file, or neural analysis or some other buzz word will give them instant insight and exceed all expectations. I do occasionally, however, run into MIS departments who are introducing change for change's sake. However, many times, MIS will not embrace change because it may threaten existing skill sets, reduced budgets, etc.

Here are some questions to ask as you evaluate the next great breakthrough. First, is there a plan for implementation? Second, are there alternatives which may allow us to use what we now have more effectively? Third, do we need a 1/4" drill or are we really looking for a 1/4" hole? Fourth, how critical is the need for a solution? Fifth, what will it take to get it done (in time and money)? Sixth, how realistic is this estimate? Seventh, what do we think the real improvement will yield in bottom line profits? What is the payback period? Eighth, do we really believe that these are good estimates?

Why people blame technology for their own lack of imagination.

It is always easier to blame the computer than to try to come up with a solution. Computers are difficult to understand and complex. Since everyone has problems with theirs, they are easy targets for blame. On the other hand, they do seem to do what we tell them (though usually not what we want). In that sense they are a reflextion of ourselves. It is therefore hard to look objectively at our computer systems without some sense of guilt. Someone thought THIS system was going to be the answer to our problems too.

I ordered from a client recently and there was no attempt at cross-sell or up-sell. When I mentioned this, they said that though the software was pretty well completed, it hadn't been implemented yet. I said, "perhaps you didn't understand my question, why was there no cross-sell?" "Because the computer wasn't ready." they replied again. In the old days... we used to take a piece of paper and a red marker and write THIS WEEK'S SPECIAL, ONE CARTON ONLY $17.95! You can hang it over each terminal, changing it requires no programing and with a little creativity, it is even graphical!

Lets not lose the forrest for the trees. Keep focused and you can successfully blend the new technology with your old worn out work horse system.