The Case Against Modifying Your Enterprise Software

Jeff Kugler - 11/22/2006

The Case Against Modifying Your Enterprise Software

Consider the case of two hypothetical companies—Company A and Company Z—each using an identical enterprise software package delivering identical core functionality. Company A has the same number of users as Company Z, but the *total cost of ownership* (TCO) experienced by Company Z is about twice the TCO enjoyed by Company A.

How could this be the case? The answer is simple. Company A chose to implement its software as is—without modifications. Now let us define our terms. Modification is not the same as customization. A modern, flexible enterprise application suite should be easily customizable to cater to most business needs and tailor the user experience—without altering the underlying code or business logic. Modification implies actually altering the code and business logic running behind the application.

- Modifications increase the cost of implementation, as custom programming becomes necessary to meet the specific demands. Modifications also increase the cost of implementation by lengthening the project timeline.
- Modification increases the cost of technical support because a software company's personnel must maintain the modifications in its code management system.
- Modification increases the cost of upgrades, as the modification must in most cases be "uplifted" each time a new version of the software is implemented. And this process once again lengthens the project timeline for the upgrade, increasing both hard costs and the soft costs that can be felt across an organization.

But wait! Doesn't Company Z obtain significant additional functionality or benefit from the modifications? In most situations the answer to this question is, sadly, no.

Increased Cost, Little Gain

I strongly caution anyone planning an enterprise software project against modifications, and have found that organizations that run the software as a stock product are happier with their investment. They have fewer problems, and find that their investment in enterprise software pays for itself—and starts yielding measurable financial rewards much more quickly than for companies which run modified software.

Almost universally, the decision to purchase and implement an enterprise software package is driven by legitimate and specific business needs. Companies need to better coordinate projects and processes across departments, or streamline financial reporting and analysis. They might invest in a software package to allow closer collaboration with customers and suppliers, or to enable efficiencies that make the company more competitive.

Enterprise software vendors appreciate these business drivers, and have developed software and support services to help companies realize their goals. But once a software product is selected for sound business reasons, the same situation unfolds time and time again. Through a desire to be inclusive and respectful, and to ease the process of change, corporate leaders seek input on the implementation process from employees throughout the corporate structure. Employees suggest modifications that might do little to meet business goals, driving up the cost of the software without delivering any improved functionality or performance.

Although those on the front lines of any business deserve every respect, once you get to a certain level in most corporate structures, individual employees will only have knowledge or understanding of their own roles in the organization, rather than a broad understanding of the business as a whole. They may not be privy to the business reasoning for changing the tools they work with every day. Many software modifications that employees insist are necessary for them to continue their current level of productivity are designed to preserve the status quo, and would do little to help the company reach its goals. In fact these modifications could be counter-productive.

Many requested modifications turn out to be unnecessary, and simply duplicate features already inherent in a new enterprise suite. In some instances of implementation, I have seen companies ask to deploy modifications numbering well over sixty, only to abandon most of those modifications shortly after going live, in favor of functionality already resident in the software. But when organizations can be convinced to run the software "as is" for ninety days, a fascinating thing happens. One at a time, requested modifications drop off the list. This is because employees' paradigms shift from the old to the new system. They find that the new software system can in fact do what the old one could do—and sometimes more. The new software environment is no longer a threat and becomes a familiar, normal part of life.

Keep in mind that with a new enterprise software system, some employees might find they have to go through additional steps to accomplish the outcomes obtained with just one click in the legacy system. But they may not know how much work, time, and expense is saved elsewhere in the organization—generally in accounting—because of the additional steps they are suddenly faced with. Employees may not be aware that the alteration they are requesting will cost tens of thousands of dollars up front, and cost even more to maintain down the road.

In some cases, department leaders and employees may have legitimate concerns about adopting new software. Perhaps their performance is measured in a way that would be impacted by either the learning curve, or by altered practices inherent in a new enterprise software package. *Information technology* (IT) staff in particular can have concerns about the adoption of a software package without modifications. After all, an IT department that historically has devoted significant resources to modifying and maintaining modifications to the legacy system could perceive a stock software package as a threat to job security. Sensitivity to these concerns will help ease the transition from a legacy system, and build acceptance to the new software environment.

Modifications to Watch Out For

- **Duplicating the Legacy System**
  "In the old system "—those are four words to watch out for when employees request modifications. It is only natural that users of a system become attached to that system, and would request modifications to make a new software package look and behave like the old one. In cases where real benefit can be derived from hanging onto elements of a legacy system, it is much less expensive across the life cycle of a software package to integrate elements of the old system with a new enterprise software package than it is to undertake custom programming—assuming that the new package is flexible and granular enough to accomplish this.

- **Data Entry Aids**
  Another type of "convenience mod" is the modification that formats or manipulates data as it is entered in the system so users can truncate information, leave out special characters, or in other ways shave time off data entry. An example might be a request to modify the system to recognize suffixes and prefixes on part numbers, and automatically fill out the remaining characters. Fully-featured systems can offer streamlined ways to enter standard data like dates. But excessive modifications to streamline data entry tend to be "penny-wise but pound-foolish."

- **The "Do My Job" Button**
  Many requests for modifications involve stringing various elements of functionality together, with the intended result of reducing the amount of work for the user. Replacement of several steps within the application with an "add parts" button, an "add purchase orders," or an "add customers" button are examples of "do my job" buttons. As a result of the prevalence of consumer software packages like **Microsoft Excel**, many users will refer to these modifications as *Wizards*. It is natural for users to desire a single-screen way to do things. For the uninitiated user, it makes sense that the new software package would require less thought, effort, and time with these Wizard modifications. But there is a steep, slippery slope towards automating all of a user's tasks so they can simply push one button that does their job so they can go home! And after using a new enterprise software package for a period of

time, these expensive Wizards quickly become obsolete as employees master the learning curve.

Common Workarounds to Avoid Modifications

The above-mentioned modifications are requested not from a desire to achieve business goals, but in an attempt to make the new software environment more familiar and predictable to the hands-on user. A consumer product like **Microsoft Excel** may be a familiar benchmark for many employees, but when you open Excel, you are looking at a blank spreadsheet. An enterprise software package, however, is not a blank slate. It is loaded with parameters and safeguards that prevent users from entering incorrect data, violating accounting rules, and otherwise running afoul of established business practices. It is natural that an individual software user would want all of the freedom of Excel, with none of the ramifications that come from all of that unbounded freedom.

Some enterprise applications will allow data to be exported into Excel, manipulated, and loaded right back into the application. Indeed, a robust enterprise suite will allow for this type of interoperability—not only with Excel, but with other proprietary and custom software. But exercise caution, because not every enterprise software system on the market provides these safeguards when pushing exported data into a software package. Take care that your shortcut to save $100 worth of effort does not cost you $20,000 in consulting time to fix corrupted data!

When Does It Make Sense to Modify Enterprise Software?

The case against modifying enterprise software is a strong one. Your standard software offering is the product of millions of dollars of research and development and extensive testing for reliable, consistent performance. The business logic and functionality is based on best practices developed over many decades by successful industries, and taught by professional organizations like the **Association for Operations Management** (**APICS**).

Used as developed, your enterprise software system will help you improve your business processes, allowing you to engineer your business as opposed to simply letting it evolve. Sometimes, a business may have developed practices that are unique, and that do help them compete in the market. In those situations, when there is a defensible business reason to modify the software and diverge from established best practices, it may be possible to cost-justify the added expense.

But be on guard against software modifications that are not driven by legitimate, measurable business needs. Justification of one such modification may open the flood gates, and you will be hard-pressed to turn down other requests for modification!

About the Author

Jeff Kugler is a solutions consultant with the Milwaukee office of **IFS North America**. Before entering the software industry, Kugler worked in manufacturing and operations. He has managed implementations both as a customer and as a consultant, and is active in lean manufacturing advocacy both inside and outside IFS. He holds a BA in computer science from Lakeland College, Sheboygan, Wisconsin (US).