8 ways to improve your Enterprise Systems

The pros and cons to updating your packaged business software

Executive Summary

IT management faces many challenges in keeping enterprise system functionality up to date. Often enterprise software systems require additional features pertaining to Business Intelligence, customer relationship management, forecasting, budgeting, customer and vendor portals, and e-commerce capabilities. This can even be true of homegrown systems custom-built in years past that now require new functions and features to match modern business requirements.

This white paper explores all of the options available to bring that essential functionality to your Enterprise systems, and the pros and cons of each method. It also provides common pitfalls be aware of and a checklist of features and topics that you should consider before making your final decision.

Introduction

Enterprise software such as Enterprise Resource Planning (ERPs), Manufacturing Resource Planning (MRPs), Supply Chain Management (SCMs), Business Intelligence, Warehouse and Distribution Management systems are powerful resources and generally sold and implemented as packaged solutions. Unfortunately, like anything you buy in a box off the shelf, one size does not fit all. This can become increasingly apparent as the package ages. Though it may still have all of the necessary power on the backend, it may not have the necessary front end capabilities that users expect and modern-day businesses require.

Many factors contribute to this trend. First, every business is different with varying business models, industry conditions, and individual company concerns. Secondly, these myriad requirements are further complicated by the diversity of locations (whether it's city, state/province, or country) and each of these governmental levels can require different reporting laws, tax rates, and industrial mandates.

Thirdly, when compounding all of these factors with corporate idiosyncrasies and individual preferences, it becomes increasingly clear how any enterprise software package can fall short of needs or expectations. Finally, there's the modernization factor. Even homegrown systems that were tailor-made and implemented for individual companies in years gone by now face demands for modern interfaces and technological capabilities.

So, the question becomes, what is the best way to improve your enterprise system(s) to provide the modern features and capabilities that your business requires? You be the judge.

Here is a thorough look at eight approaches to improving your enterprise systems.

Eight ways to improve your Enterprise Systems

IT managers face many challenges in keeping their businesses technologically up-to-date, and that is especially true when enterprise software falls short of expectations and available functionality. This expectation of functionality includes better Business Intelligence, forecasting, new product samples, budgeting, online ordering, customer Portals, vendor Portals, and the like.

As an IT Manager, here are the options available to bring new levels of functionality to your users. For our purposes, we'll refer to the systems as ERP, but these approaches apply to any packaged software systems that require analysis, retrieval, and maintenance of business information.

Approach 1: Replace your old ERP with a new ERP that has the features you need.

Pro:
You have a powerful ERP that is time-tested, but your users want a package that can provide the modern features they need? If you can find a new ERP package with these features and buy it, it will surely give you the features you are looking for, and perhaps even a few others.

Con:
High Cost: Purchasing and implementing a new ERP is incredibly expensive, particularly when you factor in the price of the lost productivity, added support requirements, training, consulting, and general IT fire-fighting that goes into ERP implementation to make it a time-consuming, painful, and costly process.

High Risk: There is a high probability of failure when you enter into ERP implementation, and it requires company-wide participation, in planning, roll-out, testing, and training from the end-user to support.

Band-aid Fix: This is not an effectual long term solution to your problem, but instead it is simply a stop-gap measure. You might get the new features you want, but you will inevitably be missing something that you need. And, the list of new features and applications that users require will grow, so that even if everything runs like clockwork, you'll end up back in the same spot in short order. It's a short-term plan, disguised by a lot of effort.

Approach 2: Do nothing.

Pro:
Nothing ventured, nothing gained…but nothing is broken either. Budgets aren't blown, essential business applications don't get broken, and no one gets fired or has to spend long hours trying to integrate new third-party software into the current ERP.

Con:
Nothing gained: Believe it or not, most companies follow this option for a fair amount of time when faced with this problem. Rather than risk throwing good money (and time, and people) after bad, they do nothing to address the need for new features, and try to make do with what they have.

Competitors gain: If you're not growing, you're dying. Your competitors are rolling right along with productive solutions while your people are spinning their wheels, and your customers tolerate the status quo. This is not a good long-term plan unless you plan to retire before it gets worse, or your competitor buys you and you stay on board long enough to help convert your data into their more efficient systems.

Approach 3: Demand the new features you want from your current ERP vendor.

Pro:
This is a valiant attempt to try and get the applications that you need without any development or commitment on your part. Maybe they can work with you.

Con:
You are at their mercy.

Long Timeframe: Unless you are Methuselah, you probably don't have time to wait this one out. The likelihood that your ERP vendor is going to listen to your appeals for the exact changes you'd like to see, out of all of the other appeals being made to them daily basis, is pretty unlikely unless you are a behemoth and their number one customer.

Ill-fitting: Even if they make the basic changes you desire, they are very likely applying a generic change to fix a multitude of sins. This is still the "pound it to fit, and paint it to match" approach, so their new solutions are not going to be exactly what you need…whether it's usability, full functionality, language choices, and the like.

Approach 4: Use desktop software (like Excel/Access) to copy data and create solutions separate from ERP database.

Pro:
This is a classic end-user "fix" to ERP problems because it's easy to implement, they can get around IT, and IT is willing to turn a blind eye, generally, because it means that nothing is done to "break" any of the behind the scenes applications.

Con:
Lack of universal access: Without centralized access (easiest via the Web), using desktop software for separate business solutions often means dealing with multiple versions in various locations opened and edited by a variety of users using different versions of the software. It would also mean that only one person would be able to maintain the data safely. This is a fine stop-gap solution, but it is not a good long-term one.

Duplication errors/Data inaccuracy: Core ERP data in this solution needs to be duplicated into whichever new "database' you choose, so the data is not live, and therefore is immediately outdated, defeating at least some of the purpose of establishing a centralized ERP in the first place.

No scalability: It will quickly outgrow its intended purpose. Excel isn't meant to handle hundreds of thousands or millions of records.

Approach 5: Purchase a bolt-on third-party solution to get the new features you need.

Pro:
Great features in the demo, everyone is excited that finally they are going to have the functionality they need, management gets on board.

Con:
Duplication errors/Data inaccuracy: Again, you'll wind up having to duplicate key parts of your database into the third-party software's database such as the customer master, product master, and the like. If you've been around long enough, you know how this turns out. Portions of your staff wind up mitigating the differences between the disparate databases, and much productivity (and practicality) is lost.

Expensive: The licensing and annual maintenance of such products are generally quite costly, and when adding in the time and effort of managing additional vendor support, training, contracts, and the like, it can become truly cost-prohibitive.

Approach 6: Customize the ERP software code itself through vendor consultants.

Pro:
It solves the problem, for now. Customizing the applications, or having consultants create the features that you require and write them directly into your ERP package absolutely fixes the issues you face today, and your users will be quite happy for the foreseeable future.

Con:
High Cost: Customizing your ERP can get very expensive. And getting the functionality you need may be a two-part venture. First, you'll need to upgrade your ERP first so you'll be customizing the most current version and have the most relevant technology at your fingertips. Secondly, the reason you'll want to upgrade first is that these customizations are going to have to last you until the next time you find yourself in the same situation you find yourself in now.

Temporary: The keyword on the pro side over there is *foreseeable*. If a few months or years down the road you find that your users need other capabilities that you couldn't or didn't foresee, or you require an upgrade to your ERP, all of those expensive customizations and features that you paid big money for will be completely overwritten, and will need to be re-done. That is doubly expensive.

Approach 7: Extend your applications by hand-coding custom applications and features you need (or hiring a consultant to do so) and tie them back to your ERP.

Pro:
This is very close to being the best option. Very close. No data needs to be duplicated into a separated database. No one's job becomes managing a third-party-software and its vendor.

To be clear, these are not changes that have to be made to your ERP package itself. These are custom applications, hand-coded, and integrated with your ERP. If you go this route, all of the changes you need are made, and they have the longevity to last as long as you need them. This solution won't be "overwritten" when you need an upgrade.

Con:
Hand-coding takes up to too much time and costs too much money.

Long Timeframe: If you want to customize these applications yourself, it is probably going to take way too long. You could have 6 projects to do, and the first 2 could take four years to complete. Whether you use your internal staff, or outside consultants, that long timeframe means large dollars.

High Cost: If you want it done sooner, you'll likely have to hire consultants, and then you are spending a lot of money on high-priced outsourcing…and they still may not get it done in the timeframe you desire.

Approach 8: Create custom applications integrated with your ERP by using an application development tool.

Pro:

Easy customization: Here, there is no duplicated data, and no need to purchase 6 separate canned software packages. You can keep data where it belongs, but add the few new tables you need, and provide a slick, modern Web interface for users. It is as simple to use as Excel/Access, but gives you the power to create well-architected custom Web applications.

Long-term and cost-effective: Your newly-built applications are accessible by everyone, able to execute over any database, and can be served from any operating system. You would complete all 6 projects in 1 year with very little overhead investment. It also enables you to continue to take advantage of your ERP vendor's upgrades, and it costs very little time and money compared with any of the alternatives.

Con:

Knowing how to choose the right application development tool can be tricky. There are many different kinds of development tools, so before selecting a tool, create a checklist of your needs.

Solution Checklist: Best Practices for extending Enterprise applications with a tool.

If you do decide to opt for an application development tool, remember to keep the following checklist by your side:

- ✓ **Web-based applications:** Find a tool that generates Web-based applications that are served from centralized servers and only require a Web browser for your users to run them. Remember, any special client-side installs should be avoided.

- ✓ **Platform requirements:** Look for platform-independence. This places your hardware management and future software decisions squarely in your hands. The Web applications you build with the tool should run on any platform you desire (Linux, Windows, UNIX, OS/400, etc.) For maximum flexibility the best tool to provide this should be able to run on any platform itself.

- ✓ **Database requirements:** First, you want a tool that can create applications that can access multiple (disparate) databases, for maximum flexibility and longevity. Secondly, **pay attention to how each tool does this.** For example, a tool that gives you the ability to create an application, compile it once, and AFTER it is compiled to just "flip a switch" to point it to any database is very different than a tool that requires you to go through the full build and compile process in order to access an individual database. It may seem like a small distinction, but having to rebuild and recompile each time adds greatly to the long term Total Cost of Ownership (TCO.)

- ✓ **Application range:** Only consider solutions that can create the widest variety of applications: analyzing, extracting and maintaining data. For example, some solutions may be great at just extracting and reporting, and that might be good enough for the BI app you need today, but what if the next edict down the pike from management is an order-entry system, or a way for users to edit their information? The best tool will be able to accommodate your wide range of Web and business needs, and will give you the ability to create other solutions going forward.

- ✓ **Architecture:** Only consider solutions with the very best (and most flexible) architecture.

  Java provides the most powerful enterprise-level applications, and also provides a systematic structure for maintaining or updating applications going forward. You might not require Java if you just wanted a quick one-off contact form on your Web site, but you most definitely want Java if you are creating Enterprise-level data-driven applications for your business. It's important to choose a tool that will provide the best possible speed and scalability.

- ✓ **Security:** When building applications, you need to make sure the tool provides security capabilities down to the record level to protect your business data.

- ✓ **Ease of use/Learning Curve:** You should consider solutions that can be used by the widest audience—developers, power users, end users, business-analysts, DBAs, and the like. The most powerful tool is one that is simple enough for the Excel/Access crowd, but powerful enough for developers who can't stand having their hands tied by vendors, and want to be more effective and productive in their creations.

- ✓ **Creativity:** It's not a word often found in technology white papers, but it should be. When looking at tools, you want to find a tool that will provide the exact bones of what you need, but still allow you the flexibility to alter your creations to fit your need or whim, whether it is changing the look through style sheets, hand-coded HTML, or a WYSIWYG painter…or whether it's altering the underlying source code to suit a specific need. You don't want to be locked in to a world where you have no freedom to be innovative. Instead, you want the best of both, so require that your vendors show you how you can have both.

Choosing an application development tool

Here are some of the common tool risks you may want to look out for.

Avoid tools that:

- Require you to learn a language or proprietary "meta-language" to be productive.
- Have a long-learning curve, or require teams of consultants.
- Have a minimal upfront cost, and are good for the initial task at hand, but require expensive modules to do any other things you may want to do.
- Require heavy outside support.
- Can only run on one platform.
- Can only access one database.
- Cannot handle the complexity of integrating legacy systems.

Conclusion

In summary, there are eight approaches before you, to get what you need from your Enterprise systems. You can replace your ERP entirely, but that is high-risk, expensive, slows productivity, and long-term you'll have the same problems. If you are privately-held, you can do nothing and make do until the problem is out of hand. You can ask your ERP vendor for specific features or changes, but any package changes are, again, not likely specific to your needs.

You can copy data to an outside database such as Access or Excel to create quick and dirty fixes. You can purchase a third-party package solution for specific features, but again you are duplicating data, have a new vendor to manage. You can have a consultant customize the ERP package itself, but these expensive changes will be overwritten when the package gets upgraded, and must be redone.

You can hand code or hire a consultant to hand code custom applications with features you need tie them directly to your ERP. Finally, you can use a tool, an application development tool, to develop your own custom applications that work seamlessly with your ERP, but won't be overwritten, and can be used in the future to continually provide you with the ability to create solutions as you need them.

Regardless of which method you choose, ultimately, it is most important to not only look at the features themselves, but the timeframe involved in implementing them, and the long term outlook. It's impossible to know where technology will be ten years from now, or where this solution might find you, so how flexible is the solution you choose, long term?

Is there platform flexibility, database flexibility, is it proprietary or is it written in a universal language that can be supported outside the vendor? The more you clarify your own needs, the more software vendors and consultants can present you with the clearest solutions for your business.

Reliable reporting and data integrity together are the basis for sound decision making, and delivering trusted information is key to remaining solvent and competitive. For operational purposes, it's necessary to have real-time access to data in your operational systems, and while some organizations choose to develop in-house systems, while still others use packaged software solutions, the important focus needs to be on using those systems to their fullest potential, and making these systems work toward your success.

About mrc:

mrc offers a comprehensive software tool suite called m-Power for developing custom enterprise-level Web applications as well as custom BI, dashboards, portal applications, report-writing, and e-commerce applications. They can be used to customize any Enterprise system like an ERP, MRP, SCM, CRM, or Warehouse management.

Applications can integrate data between multiple databases, and run on any platform.

mrc's m-Power is used by both developers and end-users to create Web applications quickly and cost-effectively.

For more information on m-Power and other mrc solutions, visit http://www.mrc-productivity.com or call 630-916-0662.

mrc

555 Waters Edge, Ste 120
Lombard, IL 60148
630-916-0662
mrc@mrc-productivity.com