

Managing Your ERP: How to avoid common pitfalls of implementation

Many organizations have undertaken enterprise-wide initiatives focused on replacing legacy systems, installing enterprise (ERP) products —such as SAP and PeopleSoft—and redesigning business processes. Some have been driven to these initiatives by Y2K concerns, others by a desire to enable growth and enhance their information management capabilities.

I have found 5 areas where ERP projects are most vulnerable, and that have been problem areas for every project I have worked on. These are:

- ◆ Understanding what integration means
- ◆ Managing communication
- ◆ Knowing your decision-making process
- ◆ Testing and managing your infrastructure
- ◆ Living with your ERP

There's a lot to be learned from past successes and failures that can make your project successful. What are the key insights about implementing and integrating an ERP into your environment? To act on some of these insights, you'll have to be a development manager or a project leader, but any ERP team member can benefit from knowing the lessons of the past. A sidebar shows how to apply them even when you have no sweeping powers.

Understanding What Integration Means

A major utilities company was implementing both the Finance modules and the Human Resources (HR) modules for an ERP product. They had divided into large teams, each with smaller sub-teams to handle the individual modules under each major category. Finance had a general ledger sub-team, an accounts payable sub-team and an accounts receivable sub-team. Human resources had a basic HR sub-team, and a benefits sub-team. There was also an infrastructure team to install and configure the technology underlying the product. It had divided into an implementation sub-team focusing on the development environments and configuration, and a deployment sub-team focusing on rolling out the technologies across the corporation.

Each team developed their own work plan, independently. After 4 weeks of planning, the team leaders for the Finance and HR teams submitted their plans to the project sponsors for approval. The infrastructure team had been working with what they could get from the other teams, and building their own plans based upon their available resources.

After the sponsor review, the infrastructure team discovered that all the teams had selected implementation dates within several weeks of each other. The sheer amount of work required to implement within those time frames was cost prohibitive, given the number of people they would need to build and manage the development, test, and

production environments needed. In addition, each of the two major functional teams were requesting to implement under a different product release.

The teams began a lot of lobbying and downright arguing. But at least they were communicating. The teams leaders began to meet as a team itself, and a single plan was created that incorporated the needs of all the teams and took into account the ability of the infrastructure team to implement within reasonable time and cost constraints.

What went wrong in the beginning? Each team was viewed as a single project. The infrastructure team was viewed as the backroom group. Planning occurred in relative isolation. This is a theme that has emerged on every single project I have seen. In an effort to simplify the management of the ERP, project teams have split themselves along functional and technical lines. But the ERP products themselves are intended to be used as one product, and so the isolation of teams results in different planning assumptions, unrelated timelines, and ultimately, an unworkable plan.

To avoid this scenario you should create a high-level planning team right from the beginning. This should include several team members from each functional area and the infrastructure team. This team should jointly lay out as many of the planning assumptions as possible such as the versions of the ERP product that will be implemented and their timelines, general development timelines, testing needs, and training timelines. The planning team does not, however, revert to diving into the details. Their roles is coordinate shared resources, monitor for consistency across teams, and manage cross-team issues as they arise.

Much of the planning will center around infrastructure needs. One example of an often overlooked issue is how long it will take to establish the development environment. Functional teams often expect to have a working version of the vanilla product in week one. It may take six weeks or more for the infrastructure team to implement a development environment, given the new hardware and software requirements that often accompany an ERP.

So, begin your planning from the top down. View the project as one project not several. And be prepared for an iterative planning process as you learn more about the product.

Managing Communication

Your project will most likely be one of the largest projects you have worked on. It may include anywhere from 100 to 300 or more team members, including functional leaders and user test personnel. When you are working with that many people, a key area to focus on is how you are going to communicate across all the teams and to all these people.

If you are going to succeed, then someone, perhaps you, must take the leadership role in providing tools and processes to help manage communication. These are communication

processes that span multiple teams—across technical and functional areas—to improve overall productivity and effectiveness. From vision to requirements planning and through the functional implementation and rollout, communication is the key to success.

I have found that it is especially effective to use a central repository structure, which is designed to be accessible by all project members—from senior management to outside vendors. It can be web-based, a Lotus Notes application or an application built in MS Exchange and Visual Basic. It typically includes vision and goals, project timelines, issue management, change control management, requirements documentation with revisions, and project tracking and status reporting. Indexing and the ability to view information in different ways make finding information easy for everyone. Using workflow features further enhances the management processes by alerting key individuals of changes or the need for approvals.

One key to the success of a repository like this is that you must be prepared to assign someone to build and maintain it. As the repository is used, people will think of better ways to organize it and new sections to add. Be sure that the repository is updated and modified to meet the needs of those you expect to use it. It will be well worth the development cost.

Decision-Making

Another area of significant problems in implementing an ERP is in getting decisions made and having them stick. When integrated products are being implemented, decisions made in one functional area can impact the design of another functional area or the plans of the technical team. In order to ensure success and eliminate rework, decisions need to be made and understood across teams.

In most of the teams I have worked with, this is the single hardest cultural change to make. It requires communicating to the right people and a formal decision-making process.

The first thing to consider is "Who can make what kind of decisions?". If the decision involves only programming issues or a change in the development process, a development manager and stakeholders such as a test leads and application leads might be sufficient. But if the decision involves changing a policy or a business process, you need to know who can make this change and make it stick. Is it the manager of the functional area, like accounting, or does it need to be made by the VP of Finance? Who else is affected and does this decision maker know that? It helps to have this worked out in advance, with a list that might say, for example, that all policy decisions for Finance must be made by the VP, while all operational decisions for finance can be made by the manager of Finance.

The decision-making process must provide the key decision-makers and other stakeholders with enough information and the right kind to make an informed decision.

All of this information should be provided to them in writing, so they can look at it, review it, and ask further questions. The documentation should include:

- ◆ a brief description of the decision that needs to be made and the options available
- ◆ a list potential impacts for each option
- ◆ any costs associated with each option
- ◆ any time delays to the project associated with each option.

The final piece of information that needs to be provided to a decision-maker is when the decision needs to be made in order not to impact the project schedule or produce further impacts. Many times a decision-maker has all the information, but just doesn't realize the impact a delay may make, and so they may hold a decision, pondering it, checking with others on their opinion. Given a deadline, most people will turn around a decision in the time necessary. Without that knowledge, there will appear to be no urgency to make the decision at all.

Having the right information in the hands of the right people makes decision-making simpler and streamlined, and reduces rumor and speculation regarding potential changes to the project or system.

Testing and Managing your Infrastructure

IT can greatly improve the chances for ERP success by following a rigorous, industry-best practice testing process. People often think of testing only in terms of the business process. "Can I run transactions and get back information like I want to?" In the days of the mainframe or in single server or departmental environments, that was quite often enough. Systems were "one box, one connection".

Today, with client/server technology, we have an enormously complex technical environment to manage. So for the ERP testing group, testing takes on a new meaning. Every support process (like backup and restore), interface, server, desktop and network component must be tested individually and then, end-to-end. You've got to test for stability and reliability, and you have to test the ability to deploy across your enterprise, on any and all platforms you choose for your servers and desktop.

Often, your environment will already consist of mixes of platforms, and variability will be found most often on the desktop. Different applications used by different end users will have established a mix of configurations and setups in your environment. To streamline the process and have a chance to make your ERP configurations work consistently, standards should be developed and adhered to for each product, especially the desktop.

The desktop arena can be the most difficult to manage. Users are used to having their workstation set up their own way, and often this can be a huge challenge to the ERP project. Establishing desktop standards may make sense to you, but it won't when you

tell your users. Be prepared to help them understand this change, why it is necessary and how you will try to support their needs, but without adding cost to the project bottom line.

Because client/server is a highly configurable arena, the ERP teams will need to send small teams of technologists to each geographic location to assess the environment, make recommendations and establish technical training plans for local support. They must later return to re-inspect for adequate compliance. This may require a change not only in your process, but in your culture.

The costs of non-compliance, and the instability of a non-standard, very complex technical environment can make testing your worst nightmare. Not managing this complexity can be too much for the project to bear, and will likely reduce the overall benefit of implementing an integrated enterprise system.

Living with your ERP

"I can't wait until this project is over and we can get back to the way things were".

This is a comment often heard on large ERP projects. But the truth is, things will not go back to the way they were after your ERP goes live. One profound learning that has come up for clients again and again is that this is a permanent and fundamental change in how they do business and in how they support the business through information technology. Many of the team structures put in place to deliver the ERP will need to remain in place to support it, although on a smaller scale.

You will always need to manage the integration of your ERP, so your planning team will still be a necessary part of ongoing operations. Communication will still need to occur across a large part of your organization, so your repository will still need to be maintained and your communication processes will need to remain in place. Decisions will always have to be made with every significant change and every upgrade of your ERP. And your infrastructure testing will be challenged with each new release of your ERP as technologies are added to enhance the capabilities of the product.

But you will have attained a new level of process maturity after this implementation. You may find that other projects seem to go more smoothly as you've learned to manage communication and decision-making. You will have learned to plan with integration in mind, and surprises will occur less often. Your development methods and processes will be more effective, and it will make sense to spread this value to other teams and projects in the company.

So if you are in the beginning stages of your ERP implementation, know that this will be a difficult journey. But this may also be an opportunity to implement the processes you've wished you could implement before. It may be the chance to develop some of the rigor you knew you needed on other smaller projects, but for which you couldn't muster

the management support. You can turn this into an opportunity for growth, and you may even have fun doing it.